Probabilistic Biological Network Alignment

Andrei Todor, Alin Dobra, and Tamer Kahveci

Abstract—Interactions between molecules are probabilistic events. An interaction may or may not happen with some probability, depending on a variety of factors such as the size, abundance, or proximity of the interacting molecules. In this paper, we consider the problem of aligning two biological networks. Unlike existing methods, we allow one of the two networks to contain probabilistic interactions. Allowing interaction probabilities makes the alignment more biologically relevant at the expense of explosive growth in the number of alternative topologies that may arise from different subsets of interactions that take place. We develop a novel method that efficiently and precisely characterizes this massive search space. We represent the topological similarity between pairs of aligned molecules (i.e., proteins) with the help of random variables and compute their expected values. We validate our method showing that, without sacrificing the running time performance, it can produce novel alignments. Our results also demonstrate that our method identifies biologically meaningful mappings under a comprehensive set of criteria used in the literature as well as the statistical coherence measure that we developed to analyze the statistical significance of the similarity of the functions of the aligned protein pairs.

Index Terms—Probabilistic biological networks, network alignment, neighborhood topology, random graphs

1 INTRODUCTION

BIOLOGICAL networks, such as protein-protein interaction, metabolic and gene regulatory networks, are essential in describing the complex mechanisms by which cells carry out numerous functions. Studying those networks has been very effective in tackling many problems such as understanding the genetic factors that impact various diseases [18], [13], drug discovery [32], [36], and investigating the relationships among organisms and species [23], [12], [13].

Biological networks characterize the interactions between biological molecules within the cell, such as genes, proteins, and enzymes. Like many processes in the biological realm, interactions are probabilistic events. An interaction may or may not happen with some probability, depending on a variety of factors such as the size, abundance, or proximity of the interacting molecules [3]. Thus, we have less than 100 percent confidence in such interactions [4].

In the remainder of this paper, if a biological network contains at least one probabilistic interaction, we call it a probabilistic network. Should all interactions be certain, we name it a deterministic network. We represent probabilistic networks using graphs with proteins as nodes and interactions as the edges. Fig. 1 (top) depicts a hypothetical probabilistic network that has three proteins and two interactions. The number on each edge is the probability of the corresponding interaction. We consider that interaction probabilities are independent of each other. An important observation is that a probabilistic network is actually a summary of all possible deterministic networks that are determined by the subset of interactions that take place. This means that a probabilistic network represented as a graph with $|E|$ edges will in fact describe the $2^{|E|}$ deterministic networks that could arise as instances of the probabilistic network, each with some probability. Fig. 1 illustrates how two probabilistic interactions lead to four deterministic networks.

Interaction probability data are becoming increasingly available in popular biological databases, such as MINT [4] or STRING [33]. The probabilities of interactions are crucial in understanding biological networks. They cannot be neglected in any research of biological networks that aspires to capture the true nature of the underlying biological phenomenon and incorporate all the information available on network topology. However, our survey revealed that they are often ignored in the computational analysis of biological networks. Certainly one deterring factor is that these networks introduce exponential number of alternative deterministic topologies (see Fig. 1).

Important insights into the structure and function of biological networks are gained by comparing them via network alignment. This approach has already been used to infer new functions for genes [6] or to delineate new networks in less studied organisms [8]. An increasing amount of research is dedicated to the network alignment problems [15], [28], [34], [22], [29], [31], [30], [25], [13], [5], [20], [26], [27]. All these methods, to the best of our knowledge, however, ignore the fact that interactions are probabilistic events. By assuming the networks to be deterministic, these methods implicitly enforce the topology of a specific instantiation of the probabilistic network over the alignment. This has a big risk of yielding biased and thus inaccurate alignments. For instance, in Fig. 1, ignoring the interaction probabilities will enforce the network topology at the bottom right which is misleading with 82 percent probability (i.e., $(1 - 0.18) \times 100$). Several methods in the literature allow for edge weights while aligning networks [29], [31], [30], [19]. These methods often view these numbers as the relative importance of the interactions...
rather than interaction probabilities. They suggest creating a bias based on these numbers through simple formulations. Singh et al. [29], [30], [31], for instance, scale the contribution of each edge to the alignment by a value that is proportional to its weight. As a result, such methods have the same drawbacks as the other existing methods discussed above: They enforce a single network topology although many alternative topologies are possible. Thus, accurate mathematical modeling of the network topology that incorporates interaction probabilities is needed. This task, however, is nontrivial, as the number of deterministic instances of the probabilistic network will lead to solutions exponential in the number of probabilistic interactions. For instance, the data set we use in our experiments (Section 3), which is downloaded from MINT [4] and KEGG [14], contains networks with up to a few thousands of probabilistic interactions. Even a medium sized network with 300 probabilistic interactions leads to $2^{300}$ (i.e., more than $10^{90}$) alternative topologies. Studying so many topologies one by one is clearly impossible even for simple computational problems on biological networks.

**Contributions of this paper.** We consider the problem of aligning two networks. Particularly, we focus on protein-protein interaction networks in our experiments. However, all of our technical discussion in this paper applies to other types of biological networks. Unlike existing network alignment methods, we allow one of the networks to be probabilistic. We also present the mathematical extension for allowing both networks to be probabilistic. Our method combines the pairwise similarity of the aligned proteins with their topologies in their networks with the help of the support matrix, a structure that describes the similarities of the topologies of pairs of nodes one from each network. This strategy has been successfully used for deterministic networks by a number of methods including IsoRank [21], [25], [29], [30], [31] and SubMap [2]. However, there is no method in the literature that does that for probabilistic networks. To tackle this challenge, we compute the expected support matrix that is derived from all possible deterministic networks induced by the given probabilistic network.

**Statistical Coherence.**

We formulate the statistical coherence score that characterizes the biological significance of a network alignment.

The remainder of the paper is organized as follows: In Section 2 we describe our method in detail. In Section 3, we present our experimental results. Finally, we conclude the paper with a summary in Section 4.

2 Technical Details

We start by briefly presenting a previously proposed solution to the alignment problem for deterministic networks, namely the IsoRank algorithm, which is essential in understanding our method (Section 2.1). We then formulate our probabilistic alignment problem and present the mathematical framework that will allow us to solve the probabilistic problem efficiently (Section 2.2). We finally focus on the mathematical and algorithmic aspects of our method that addresses the probabilistic network alignment problem (Section 2.3).

### 2.1 Deterministic Alignment Using Network Topology

IsoRank is a popular algorithm for aligning deterministic networks [2], [21], [29], [31]. This section summarizes IsoRank and serves as the basis for our method.

Biological networks are naturally represented mathematically as graphs. The graph $G = (V, E)$ has a node $v \in V$ for each protein and an edge $e \in E$ for each protein-protein interaction.
interaction. Suppose that we would like to align two networks represented by graphs \( G_1 = (V_1, E_1) \) and \( G_2 = (V_2, E_2) \). Let us denote the degree of node \( u \) by \( n_u \) and also \( |V_1| = m, |V_2| = n \). The IsoRank algorithm mimics the power method for computing the principal eigenvector of a matrix [11]. The 2D, doubly indexed support matrix is defined for \( i, u \in \{1, \ldots, m\}, j, v \in \{1, \ldots, n\} \) as

\[
A[i,j][u,v] = \begin{cases} 
\frac{1}{n_u n_v}, & \text{if } (i, u) \in E_1 \text{ and } (j, v) \in E_2, \\
\frac{1}{mn}, & \text{if } n_u n_v = 0, \\
0, & \text{otherwise}.
\end{cases} 
\]  

(1)

The support matrix captures the local topological similarity of edges \( (i, u) \in E_1 \) and \( (j, v) \in E_2 \). The term \( \frac{1}{mn} \) for \( n_u n_v = 0 \) appears here for the situation when node \( u \) or \( v \) is isolated. This does not happen often in existing data sets, but it has to be added to make the matrix stochastic if there are isolated nodes in the networks. Even if the data set we use does not have isolated nodes, this case becomes important in our probabilistic method, as we will see in the following section.

Once the matrix \( A \) is computed, the IsoRank algorithm uses the power method iterations to find the benefit of matching any pair of proteins, one from \( G_1 \) and the other from \( G_2 \) as a vector \( H \). Each entry of \( H \) maps to such a pair of proteins. To compute the final protein-protein similarity, IsoRank first builds an initial vector \( H_0 \) such that each entry of \( H_0 \) contains the pairwise sequence similarities of two proteins. It then iterates the following update equation:

\[
H \leftarrow \alpha A H + (1 - \alpha) H_0. 
\]  

(2)

Here, \( \alpha \) is a constant which balances the relative contributions of the network data and the sequence data to the pairwise node similarity. Essentially, \( \alpha \) indicates how much of the information comes from the a-priori node similarity \( (H_0) \) and how much from the local edge similarity \( (A) \). Determining an appropriate value for \( \alpha \) was investigated in [29] and [2]. Here, we adopt these previous findings.

The IsoRank method has a version that takes edge weights into consideration. The edge weights here are considered as the importance of the corresponding edges. IsoRank integrates these weights by scaling the contribution of each edge to an entry in \( H \) by its weight. We refer the readers to the IsoRank paper for details on the computation of \( A \). Such scaling creates a positive bias toward pairs of edges with large edge weights. However, it is important to note that it does not consider edges as probabilistic interactions and thus ignores all the alternative network topologies that may be observed as a result of the probabilistic interactions.

### 2.2 Probabilistic Network Alignment

We now generalize the IsoRank algorithm to the situation in which network \( G_1 = (V_1, E_1) \) is deterministic and network \( G_2 = (V_2, E_2) \) is probabilistic. Later, we will show how our method can be generalized to the situation in which both networks are probabilistic.

The main challenge in aligning probabilistic networks is to use the iterations defined in (2) for computing \( H \). This is because the probabilistic nature of \( G_2 \) makes it impossible to pick a specific deterministic topology and its corresponding support matrix \( A \). There are totally \( 2^{|E_2|} \) alternative support matrices each appearing with some probability. We model all of these support matrices with a matrix of random variables and replace \( A \) with the expected value of \( A, E[A] \).

We formally define a probabilistic graph as \( G = (V, E, P(\cdot)) \), where \( V \) is the set of nodes, \( E \) is the set of possible edges, and \( P(\cdot) \) is a function that indicates the probability of existence for edges, i.e., \( P(e) = p_e \) is the probability that edge \( e \in E \) exists. We denote the set of nodes connected to \( v \) that exist in \( G \) by \( N_v \). The random variable \( N_v = |N_v| \) models the degree of \( v \) for \( v \in V \). Let \( d_v \) be the largest possible degree for vertex \( v \). The sequence \( P(N_v = k) \), for \( k = 0, 1, \ldots, d_v \), is the degree distribution of \( v \) in the probabilistic graph \( G \).

The support matrix is now a random matrix with entries that depend on \( N_v \), i.e.,

\[
A[i,j][u,v] = \begin{cases} 
\frac{1}{n_u n_v}, & \text{if } (i, u) \in E_1 \text{ and } (j, v) \in E_2, \\
\frac{1}{mn}, & \text{if } n_u n_v = 0, \\
0, & \text{otherwise}.
\end{cases} 
\]  

(3)

We investigate now how to compute the expectation of this matrix. We focus on a specific entry of \( A \), namely \( A[i,j][u,v] \). We note that \( n_u \) is a deterministic quantity, so if \( n_u = 0 \) the result is a constant, \( \frac{1}{mn} \). If \( n_u \neq 0 \) and \( (i, u) \not\in E_1 \), the result is also constant, 0. For the rest of the cases, we have to apply the definition of expectation of a discrete random variable

\[
E[A[i,j][u,v]] = \sum_{\kappa} \kappa P(A[i,j][u,v] = \kappa), 
\]  

(4)

where the summation is taken over the possible values of \( A[i,j][u,v] \). These values are 0, \( \frac{1}{mn} \), or \( \frac{1}{n_u n_v} \), for \( N_v \in \{1, 2, \ldots, d_v\} \). The term with a value of 0 is of no consequence in the summation, so we need to find the probabilities of each of the two remaining cases. The value \( \frac{1}{mn} \) arises if \( N_v = 0 \). The values \( \frac{1}{n_u n_v} \) arise if \( (j, v) \in E_2 \). This introduces a computationally interesting case. If the probabilistic edge \( (j, v) \) is guaranteed to appear in \( E_2 \), it implies that the \( v \) and \( j \) are definitely neighbors of each other. This provides us prior information about the number of neighbors of \( v \). Thus, the probability distribution of the random variable \( N_v \) must take this information into consideration. We can express this mathematically as a conditional probability \( P(N_v = k | (j, v) \in E_2) \) for \( k = 1, 2, \ldots, d_v \). After some algebraic manipulations we get

\[
E[A[i,j][u,v]] = 
\begin{cases} 
\frac{1}{mn} \sum_{k=1}^{d_v} P(N_v = k | (j, v) \in E_2), & \text{if } (i, u) \in E_1, \\
\frac{1}{mn}, & \text{if } n_u = 0, \\
0, & \text{otherwise}.
\end{cases} 
\]  

(5)
The same line of reasoning can be applied to address the situation in which both networks are probabilistic, leading to the following formula:

\[ E[A[i, j][u, v]] = \frac{1}{mn} P(N_e = 0) P(N_v = 0) + P((i, u) \in E_1) P((j, v) \in E_2) \]

\[ \times \sum_{k=1}^{d_u} \sum_{l=1}^{d_v} \frac{1}{kl} P(N_u = k | (i, u) \in E_1) P(N_v = l | (j, v) \in E_2). \]

(6)

We note that allowing both networks to be probabilistic adds another nesting level to the summation in this formula. In general, the cost of computing a single entry in the matrix will grow exponentially with the number of probabilistic networks. Our key contribution here is to show that we can compute the distribution for a probabilistic network in polynomial time. Scaling this to multiple probabilistic networks is an interesting future work on top of this work.

2.3 Computing Node Degree Distributions

The key to computing the terms in \( E[A] \) is to compute the degree distribution of each node of the probabilistic graph efficiently. More explicitly, we need to compute \( P(N_e = k) \) for all \( v \in V \) and all \( k \in \{1, 2, \ldots, d_v\} \). One way to find the distribution of \( N_v \) is to use a direct method that enumerates all possibilities of getting \( k \) neighbors for any given \( k \):

\[ P(N_v = k) = \sum_{S \subseteq N_v, |S| = k} \prod_{e \in S} p_e \prod_{e \in N_v - S} (1 - p_e). \]

(7)

A quick examination of this formula reveals that the total number of operations is \( 2 \times d_v \times 2^{|S|} \). The exponential term arises because we need to produce all subsets of \( N_v \) of a given size. Next, we show how we can reduce this complexity to \( O(d_v^2) \). To achieve this, we adopt the probability generating function, as it captures discrete distributions in a compact manner.

Definition 1. Let \( X \) be a discrete random variable taking the values \( 0, \ldots, N \). The PGF of \( X \) is defined as

\[ Q_X(z) = E[z^X] = \sum_{k=0}^{N} P(X = k) z^k. \]

(8)

Example 1. Let \( X \) be a discrete random variable taking values from \( \{0, 1, 2, 3\} \), with the distribution given by the sequence \( a = [0.3, 0.05, 0.15, 0.5] \). In other words, \( P(X = 0) = 0.3, P(X = 1) = 0.05, P(X = 2) = 0.15 \), and \( P(X = 3) = 0.5 \). The PGF of \( X \) is the polynomial

\[ Q_X(z) = 0.3 + 0.05z + 0.15z^2 + 0.5z^3. \]

(9)

By simply listing the coefficients of this polynomial, we get the distribution of \( X_v \), that is the sequence \( a \).

In what follows, we will show that computing the PGF of the random variable \( N_v \) \( (v \in V) \) is much easier than applying (7) to compute the probability of a specific instantiation of \( N_v \). More specifically, we present a theorem that provides us with the means to compute the PGF of \( N_v \) for any \( v \in V \). As we discuss later, this theorem is essential in computing the expected value \( E[A] \) in (5).

Theorem 1. Let \( G = (V, E, P) \) be a probabilistic graph and \( v \in V \) a node in this graph. Let \( E_v \) be the set of edges incident on \( v \). The PGF of the degree distribution of \( v \) is

\[ Q_{N_v}(z) = \prod_{e \in E_v} (1 - p_e + p_e z). \]

(10)

Proof. We define \( X_v \) to be the indicator random variable that has value 1 if \( e \) exists and 0 otherwise. It follows that \( N_v = \sum_{e \in E_v} X_e \). The PGF of \( X_v \) is \( E[z^{X_v}] = P(X_v = 0) z^0 + P(X_v = 1) z^1 = 1 - p_e + p_e z \). Using the independence of \( X_e \), we have

\[ Q_{N_v}(z) = E[z^{N_v}] = E[z^{\sum_{e \in E_v} X_e}] = E \left[ \prod_{e \in E_v} z^{X_e} \right] = \prod_{e \in E_v} E[z^{X_e}] = \prod_{e \in E_v} (1 - p_e + p_e z). \]

(11)

Example 2. The PGF of the distribution for node \( a \) from the probabilistic graph in Fig. 1 is simply the multiplication of the polynomials corresponding to each incident edge: \( (0.7 + 0.3z)(0.4 + 0.6z) = 0.28 + 0.54z + 0.18z^2 \). Thus, \( P(N_v = 0) = 0.28, P(N_v = 1) = 0.54 \), and \( P(N_v = 2) = 0.18 \).

We have presented a powerful result and proved it in a general and succinct manner. In the Supplementary Materials, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2012.142, Section 5.3, we provide a more detailed discussion with insights on the relevance of this result to our specific problem domain.

We have already noted that our goal in (5) is to compute conditional distributions. We are not far from our goal, since Theorem 1 allows us to compute conditional distributions as easily. In particular, to condition on a particular edge \( e \in E_v \) being present, we simply divide the PGF of \( N_v \) by 1 - \( p_e + p_e z \), the PGF of \( N_v \) when the graph is restricted to edge \( e \) only. This produces the PGF of \( N_v^{e} \), the number of neighbors of \( v \) excluding edge \( e \). When we shift this distribution by 1, we get \( P(N_v = k | e \in E_v) = P(N_v^{e} = k - 1) \). Intuitively, this holds because the probability to observe \( k \) neighbors of \( v \) in a probabilistic graph in which we know that an incident edge is present is the same as the probability of observing \( k - 1 \) neighbors in a graph that
does not have that edge (see also the online Supplementary Materials Section S.3).

2.4 Algorithm

Algorithm 1 describes how we compute $E[A]$ in detail. Briefly, it first computes the PGF of the degree distribution for each node of the probabilistic graph. Then, for each node, it considers each incident edge and computes the PGF of the degree of the node conditioned on the edge. This further allows the computation of each entry of the expected support matrix.

Algorithm 1. Computation of expectation of support matrix $E[A]$

Require: Graph $G_1 = (V_1, E_1)$
Require: Probabilistic graph $G_2 = (V_2, E_2, P)$
Ensure: Return $E[A]$, expected value of support matrix

1: for all $v \in V_2$ do
2: \hspace{1em} $Q_{Nv} = 1$
3: end for
4: for all $(j, v) \in E_2$ do
5: \hspace{1em} $Q_N = Q_{N} \times (1 - P((j, v)) + zP((j, v)))$
6: \hspace{1em} $Q_{Nj} = Q_{Nj} \times (1 - P((j, v)) + zP((j, v)))$
7: end for
8: for all $u \in V_1$ do
9: \hspace{1em} if $n_u = 0$ then
10: \hspace{2em} for all $i \in V_1, v \in V_2, j \in V_2$ do
11: \hspace{3em} $E[A[i, j][u, v]] = \frac{1}{mn}$
12: \hspace{2em} end for
13: \hspace{1em} else
14: \hspace{2em} for all $i \in V_1$ do
15: \hspace{3em} if $(i, u) \not\in E_1$ then
16: \hspace{4em} for all $v \in V_2, j \in V_2$ do
17: \hspace{5em} $E[A[i, j][u, v]] = 0$
18: \hspace{4em} end for
19: \hspace{2em} end for
20: \hspace{2em} else
21: \hspace{3em} for all $j \in V_2$ do
22: \hspace{4em} $Q_N^j = \frac{Q_{Nj}}{1 - P((j, v)) + zP((j, v))}$
23: \hspace{4em} $S = 0$
24: \hspace{4em} for $k = 1 \rightarrow d_v$ do
25: \hspace{5em} $S = S + \frac{1}{k} \times Q_N^j ||_{k-1}$
26: \hspace{4em} end for
27: \hspace{4em} $E[A[i, j][u, v]] = \frac{S \times P((j, v))}{n_u} + \frac{Q_{Nj}}{mn}$
28: \hspace{4em} end for
29: end if
30: end for
31: end if
32: end if
33: end for

To simplify the algorithm, we will introduce two notations. First, we will denote the $k$th coefficient of the polynomial $Q$, i.e., the coefficient of $z^k$, by $Q||_k$. Second, given a probabilistic edge $(v, j)$, we will denote the conditional PGF of a node $v$ with the condition that the edge $(v, j)$ exists with $Q_N^v$. The algorithm takes a deterministic and a probabilistic graph as input. It works in two phases. In the first phase (lines 1-7), it computes the PGF for each node of the probabilistic graph. In the second phase (lines 8-31) it computes the expected value of the support matrix, $E[A]$. We explain each phase in more detail next. We use $z$ for the free variable of the polynomials.

- **First phase.** The algorithm initializes all the polynomials to 1 (lines 1-3). It then incrementally builds them by considering one probabilistic edge at a time (lines 4-7). Each edge corresponds to a first degree polynomial. It multiplies that polynomial with the PGF of the two nodes incident to that edge and updates those PGFs.

- **Second phase.** The algorithm first deals with the two deterministic cases in (5), which are $n_u = 0$ (lines 9-12) or $(i, u) \not\in E_2$ (lines 14-18). For the probabilistic cases, we examine each edge $(v, j)$ of the probabilistic graph one by one (lines 20-21). Recall from Section 2.2 that when a probabilistic edge is present, we need to compute the conditional distribution for the number of neighbors with the prior information that that edge exists. In line 22, our algorithm computes the conditional distribution of the nodes at the endpoints of each probabilistic edge conditioned on that edge. It does that by dividing the PGF of the node by the first degree polynomial corresponding to that edge. Conceptually, this can be considered as removing the contribution of that probabilistic edge from the PGF. Next, we compute the expected value of one entry of the matrix $A$ by summing all of the possible values for $1/N_v$ multiplied by their probabilities (lines 23-27).

The number of operations to compute the PGF of each node $v$ is $3 \times (d_v - 1)^2/2$. An additional $O(d_v)$ cost is incurred to compute conditional probabilities. For clarity, Algorithm 1 is presented in an unoptimized form. The loops in lines 8 and 14 combined are actually an iteration over the edges of one graph. Similarly, lines 20 and 21 seem to be a double nested loop over the nodes of the other graph, but they are actually an iteration over the edges, because $P((j, v)) \neq 0$ only if $(j, v)$ is an edge. For simplicity, in the analysis, we have assumed that both input graphs have similar sizes and densities. Assuming that, on average, the number of probabilistic edges incident on each node is $M$, the overall effort required for the entire graph is $O(|E|^2 M + |V| M^2)$.

Not only the theoretical complexity of the algorithm is small, but, as we will see in the next section, the implementation of the algorithm is fast as well. Without the use of this formalism, the analysis of the distribution of $N_v$ thus the computation of the matrix $E[A]$ would be much more expensive.

3 Results

This section presents the experimental evaluation of our method.

**Implementation details.** We implemented our algorithm in C++. We also implemented IsoRank [29] in C++ as the state-of-the-art method for comparing deterministic networks. We implemented both the unweighted and the weighted versions of IsoRank. We set the IsoRank parameter $\alpha$ (see Section 2.1), the relative contribution of topology and pairwise similarity, to 0.6. This is because the literature suggests that this value provides best results
for IsoRank [2], [29]. We computed the similarity between protein pairs (i.e., the $H_0$ vector in Section 2.1) using BLAST [1]. We ran our experiments on a standard desktop computer with 4 GB of RAM and a 2-GHz processor.

**Data set.** Our primary data source was the MINT database [4], which contains protein-protein interactions in 30 main organisms. MINT keeps a single network per organism, each network having tens of thousands of interactions. In addition to this data set, we extracted smaller yet biologically coherent subnetworks of different sizes with the help of the KEGG database [14]. KEGG decomposes the networks hierarchically according to their underlying functions. We decomposed the MINT networks into the networks defined by KEGG, by partitioning the protein sets according to the leaf level of KEGG, but adopting the edges defined by MINT, including the interaction confidence. To do this, we scanned the interactions from MINT and if both participating proteins belong to the same leaf level KEGG network, we assign the interaction to the appropriate network. We removed the networks that had fewer than 10 nodes. The final data set contains 198 networks from 10 organisms. We will call this the partitioned data set.

In Table 1, we summarize the number of networks and average and maximum number of nodes and edges for each organism of the data set. We plot the distribution of the network size in terms of number of proteins and the number of interactions in all networks in the online Supplementary Materials (see Figs. SM-I and SM-II).

While using unweighted IsoRank, we ignored all the interaction probabilities, as this method cannot deal with probabilistic interactions. While using weighted IsoRank, we used the confidence values as the edge weights. The MINT confidence values are a weighted average of several factors that impact the reliability of the reported interactions, such as size and type of the experiment by which the interaction was observed, whether the interaction is unequivocally direct or not, the number of interaction partners detected in a single purification, the sequence similarity and the number of publications supporting the interaction. The final values reported are in the range [0, 1] [4]. The data, the code, and the results we obtained are available at http://bioinformatics.cise.ufl.edu/probAlign.html.

### 3.1 Agreement with the Deterministic Methods

Before we start evaluating the biological significance of our method, we need to answer the following fundamental question: Does the probabilistic model yield any alignment that we do not get using the deterministic model or the model that uses interaction probabilities as edge weights? In other words, is there any need for our probabilistic model? Our first experiment answers this question.

In order to do this reliably, we define a metric named *agreement* as the ratio of the number of nodes that are mapped identically by both alignments to the size of the alignment (i.e., the size of the smaller network). This is a number between 0 and 1. A smaller value indicates that the two alignments differ significantly, whereas a value of 1 indicates that the two alignments are identical. We computed the agreement between the alignment of our method and of both the weighted and unweighted IsoRank methods. We used the partitioned data set in this experiment. We aligned each network from one organism with the corresponding network in another organism for all organism pairs, which yielded 244 experiments. We present the results in Fig. 2. The graph shows that the results obtained by our method are significantly different than those obtained by both the weighted and unweighted versions of IsoRank. The probabilistic model completely agrees with the weighted and unweighted deterministic methods in less than 6 percent of the experiments. In many experiments, they produce noticeably different alignments. Weighted and unweighted IsoRank produce identical results in around 42 percent of the experiments. However, they disagree with each other significantly in more than half of the experiments. Notice that these results do not indicate which of the three methods compared is producing more significant results. They, however, suggest that our method produces novel alignments most of the time. Thus, ignoring the interaction probabilities or using them as edge weights can affect the accuracy of the final result. Our next three sets of experiments take a very close look into the alignments and evaluate whether these novel alignments are significant.

### 3.2 Functional Coherence of the Alignments

From Section 3.1, we know that the probabilistic model yields novel alignments. Here, we evaluate whether these results are biologically significant. Singh et al. [31] defined a measure, named *functional coherence*, to demonstrate the significance of the IsoRank method. This measure takes into account the individual pairwise matches between one node in one network and the corresponding node in the other network. Given an alignment, this measure considers each pair of aligned proteins, one by one. For each such pair, it considers the GO terms associated with them. The GO

<table>
<thead>
<tr>
<th>Table 1: Network Statistics per Organism</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organism</td>
</tr>
<tr>
<td>------------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>C. elegans</td>
</tr>
<tr>
<td>D. melanog.</td>
</tr>
<tr>
<td>E. coli</td>
</tr>
<tr>
<td>H. pylori</td>
</tr>
<tr>
<td>H. sapiens</td>
</tr>
<tr>
<td>M. musculus</td>
</tr>
<tr>
<td>R. norvegicus</td>
</tr>
<tr>
<td>S. cerevisiae</td>
</tr>
<tr>
<td>S. pombe</td>
</tr>
<tr>
<td>T. pallidum</td>
</tr>
<tr>
<td>Entire dataset</td>
</tr>
</tbody>
</table>

![Fig. 2. Agreement of alignments obtained using the unweighted IsoRank, weighted IsoRank, and our probabilistic method between pairs of methods. The results are sorted in increasing agreement order for each method pair.](http://bioinformatics.cise.ufl.edu/probAlign.html)
values mean better coherence. The diagonal line is the mean of the functional coherence obtained using the probabilistic method and the weighted IsoRank method. Higher values mean better coherence. The diagonal line is the $x = y$ line.

database organizes the GO terms in a hierarchical manner through “is-a” and “part-of” relationships. Thus, they are not necessarily independent of each other. Functional coherence chooses an independent subset of these GO terms by considering only the annotations that are at the fifth level in the GO hierarchy. The rationale for choosing the fifth level is that lower levels will not have sufficient specificity. For every pair of GO terms in the resulting set, it computes the similarity of those terms as a function of the number of the proteins in which they appear together. It defines the coherence of a pair of proteins as the median of the similarity of the terms. Finally, it computes the overall functional coherence of the alignment as the average of the coherence of all mapped protein pairs in the alignment. We refer the interested reader to Singh et al. [31] for further details on computation of this function.

Although proteins are annotated with GO terms based on sequence similarity on a large extent, the same information used by IsoRank for computing the initial homology vector $H$, the homology vector is subsequently altered using the topology matrix to obtain a topologically meaningful alignment that is also functionally coherent.

We computed the functional coherence of our method as well as that of the weighted and unweighted IsoRank methods for all the 244 queries in the previous section. The functional coherence of the two IsoRank versions was identical in almost all the experiments. Therefore, we report only the weighted IsoRank and our method in Fig. 3. The results demonstrate that the two methods have very similar functional coherence values with only a few minor differences. This is very interesting given that the two methods disagree in their alignments greatly (see Fig. 2). The reason for this mainly lies in the definition of the functional coherence function. This function computes the similarity of the GO terms rather than that of the aligned proteins directly. Also, since it reports the average of the medians, it cannot tell if a mapping has many highly similar terms. The median of a distribution is not a accurate representation of the entire distribution and thus the result it returns is not sensitive enough to tell the difference between different alignments. The fact that weighted and unweighted IsoRank had identical functional coherence values in our experiments supports this.

A second observation following from our results is that almost all the network pairs had functional coherence values in a very narrow band, between 0.15 and 0.25. This is consistent with the results reported at Singh et al. [31] as they observed values in the 0.2 to 0.22 range on their data sets. The narrow band also supports our earlier observation that the functional coherence value is not very sensitive. Moreover, the functional coherence is not calibrated w.r.t. random alignments (i.e., it is not a $p$-value), thus large values do not necessarily indicate a good alignment. We present a detailed case study analysis of this issue in Section S.4, which is available in the online supplemental material. In summary, there is no clear difference between our method and IsoRank based on this measure. Both methods are equally good.

### 3.3 Evaluation of the Functional Enrichment

The functional coherence measure discussed in Section 3.2 focused on individual mappings to evaluate the significance of the results. Another criterion for understanding the biological relevance of the alignment is the uniformity of the functions of the proteins in the entire subnetwork found by the alignment. Uniformity, here, indicates that the resulting subnetwork collectively performs the same biological function. One measure commonly used to evaluate this is the functional enrichment of the result network [28]. Smaller values of this measure mean better enrichment.

In this experiment, we computed the functional enrichment for our method as well as the weighted and unweighted IsoRank methods for all the 244 homologous network pairs used in the previous sections. All the three methods produced very similar enrichment results, so we plot the results for our method only in Fig. 4.

Results demonstrate that all the methods identify functionally enriched results. For instance, in 81 percent of the experiments the functional enrichment scores are less than $10^{-10}$ indicating that the probability of observing such enriched subset of proteins by chance is less than $10^{-10}$, which is nearly impossible. The fact that all three methods are highly enriched can be explained by two reasons. First, the KEGG organizes the subnetworks into functional groups. Thus, it is not surprising to observe uniformity of the GO annotations in the alignment. Second, the enrichment score only focuses on the most enriched GO term and ignores the rest of the terms. Thus, it is an optimistic measure. That said, it is encouraging to observe that the disagreement (see Fig. 2) between our method and the IsoRank does not diminish its ability to return functionally enriched subnetworks. We conclude that our method can identify subnetworks that are as functionally enriched as both
weighted and unweighted IsoRank. Since neither functional enrichment nor functional coherence explain the disagreement between the two methods, we investigate this matter further in the next experiment.

### 3.4 Statistical Coherence of the Alignments

One drawback of the functional coherence measure is that it does not consider the distribution of all the GO terms in the aligned networks, but only the term pair with median similarity. This makes it very hard to understand the significance of this measure. Here we define a new measure, the statistical coherence, that addresses this problem. Similar to the functional coherence measure defined in Section 3.2, this measure considers each pair of aligned proteins, one by one. For each such pair, it considers the same set of independent GO terms associated with them as those chosen by the functional coherence measure (see Section 3.2 on how the terms are chosen). The statistical coherence measure differs from the functional coherence in how it uses these terms. We first discuss how the statistical coherence is computed for one pair of aligned proteins. Then we describe how the score is computed for the entire alignment.

Briefly, we compute the coherence of the mapping based on the following two conjectures: 1) The more common GO terms an aligned protein pair has, the more coherent the mapping is. 2) It is more significant to map two proteins with common terms that are rare in the two input networks. The first conjecture makes sense because, as the GO terms considered are independent, a larger number of common terms implies more evidences denoting that the two aligned proteins serve similar roles. The second conjecture takes the background distribution of the terms into consideration. For instance, if a term exists in all the proteins of the input networks, even a random alignment will map that term. Consequently, the second conjecture is important. We choose this factor to be $\log(p_i)$, so $X$ becomes $X = \sum_{t_i \in T} A_i B_j (\log(p_j))$, where we have denoted $P(A_i = 1) = P(B_j = 1) = p_j$. Then our probability becomes

$$P \left( \sum_{t_i \in T} (A_i B_j (\log(p_j))) \geq \sum_{t_j \in A' \cap B'} \log(p_j) \sum_{t_i \in T} A_i = m, \sum_{t_i \in T} B_i = n \right).$$

We approximate this probability by computing the mean and variance. The variance is computed from the second moment, given by

$$E[X^2|M = m, N = n] = \sum_{t \in T} \sum_{t_j \in T, i \neq j} \frac{P(M = m|A_i = 1, A_j = 1) P(N = n|B_i = 1, B_j = 1)}{P(N = n)} + \sum_{t \in T} P(M = m|A_i = 1) P(N = n|B_i = 1) \frac{p_j^2 (\log(p_j))^2}{P(N = n)}.$$

We discuss how we derive this equation in the online Supplementary Materials (see Section S2).

Let $T = \{t_1, t_2, \ldots, t_r\}$ be the set of all terms observed in the proteins of the two given networks to be aligned. For each term $t_i \in T$, we define an indicator random variable $A_i$ which takes the value 1 if the term is in $A'$ and 0 otherwise. Similarly, we define indicator random variables $B_i$ which take the value 1 if $t_i \in B'$ and 0 otherwise. These definitions allow us to compute $X$ as $X = \sum_{t_i \in T} A_i B_j$. This quantity does not discriminate the terms that appear very frequently in the two given networks from those that appear less frequently. Thus, a term which is very frequent in GO and has a high chance to be associated to two proteins will have the same contribution in the final score as a rare term. In our score, we want to give a greater emphasis to rare terms that are common to the two matched proteins, so we add a weight factor that is large when the term is rare. We choose this factor to be $-\log(p_j)$, so $X$ becomes $X = \sum_{t_i \in T} A_i B_j (-\log(p_j))$, where we have denoted $P(A_i = 1) = P(B_j = 1) = p_j$. Then our probability becomes

$$P \left( \sum_{t_i \in T} (A_i B_j (-\log(p_j))) \geq \sum_{t_j \in A' \cap B'} (-\log(p_j)) \sum_{t_i \in T} A_i = m, \sum_{t_i \in T} B_i = n \right).$$

We approximate this probability by computing the mean and variance. The variance is computed from the second moment, given by

$$E[X^2|M = m, N = n] = \sum_{t \in T} \sum_{t_j \in T, i \neq j} \frac{P(M = m|A_i = 1, A_j = 1) P(N = n|B_i = 1, B_j = 1)}{P(N = n)} + \sum_{t \in T} P(M = m|A_i = 1) P(N = n|B_i = 1) \frac{p_j^2 (-\log(p_j))^2}{P(N = n)}.$$
We report the geometric mean of the accuracy and fast convergence using the Lanczos approximation [17]. These values can be computed numerically with the incomplete gamma function, which is the complement of the cumulative distribution function of this distribution, in logarithmic scale. The experiments are sorted by the ratio value.

From these quantities, we can easily compute the mean and variance and then further the parameters of the gamma distribution with the same expected value and variance. To obtain the probability we are interested in, we need to evaluate the complement of the cumulative distribution function of this distribution, which is the incomplete gamma function. These values can be computed numerically with accuracy and fast convergence using the Lanczos approximation [17]. We report the geometric mean of the p-values for all pairs.

We computed the statistical coherence score for our method as well as for the weighted and unweighted IsoRank methods for all the 244 homologous network pairs used in the previous two sections. A large statistical coherence means a high probability that an alignment is coherent by assigning GO terms randomly to proteins. For 155 queries, the statistical coherences of all the methods were larger than 0.01, so we ignored those queries, since there was a 0.01 probability that the alignments are coherent by chance. This threshold separates the samples in the data set for which a strong conclusion can be made that good alignments have been found from the samples about which we are less confident. Weighted and unweighted IsoRank produced very similar results, which is not surprising as their results completely agree in around 40 percent of the queries (see Fig. 2). Overall, weighted IsoRank was slightly better than the unweighted one, so we report the comparison between our method and weighted IsoRank in Fig. 5.

The results demonstrate that the probabilistic method outperforms the IsoRank method (both weighted and unweighted) in identifying meaningful alignments. Overall, in 82 percent of the experiments, the probabilistic method leads to statistically more coherent alignments than IsoRank. We observe that the gap between the two methods tends to increase in favor of our method as the functional coherence increases. Notice that the numbers in the figure are in logarithmic scale (base 10). Thus, in 37 percent of the experiments the probabilistic method returns results that are statistically at least two times more significant than those of IsoRank. In 8 percent of the experiments, the significance of the results of our method was more than 10 times that of IsoRank. These extremely good values of the statistical coherence obtained using the probabilistic method in comparison with the deterministic methods, with or without weights, show that network topology is essential but not sufficient to map the functionally similar proteins with each other.

These results coupled with those in Fig. 2 present strong evidence that considering interaction probabilities leads to novel and significant alignments, and thus there is great value in using these probabilities that are readily available.

Unlike the functional coherency measure, the statistical measure not only indicates which method is better but also the quality of the result. This is because it estimates the p-value of the test under the assumption that the alignment is random. By computing the p-value of the functional coherency, the same correction for the background random alignment would happen—the measure would indicate quality as well as relative performance. Unfortunately, due to the fact that the formula for functional coherence contains a median computation, the approximation of the p-value seems very hard. One of the motivations in selecting our statistical coherence measure was to be able to compute the p-value. The functional enrichment also has this property. We elaborate on this and provide experimental evidence in the online Supplementary Materials (see Section S.4).

### 3.5 Alignment of Entire Networks

Our previous experiments in this paper were made with the partitioned data set to observe the performance of our method. Most of these networks were small in terms of the number of edges and nodes. Also, each partition was determined by KEGG based on the main function of the proteins in that network. In this experiment, we take another step and evaluate our method on alignments of entire PPI networks of different organisms, with the goal of observing how well it identifies novel alignments. Table 2 lists the alignments we performed in this experiment on data sets taken from MINT. We aligned these networks using our method as well as the weighted IsoRank method.

We observed noticeable disagreement between our method and IsoRank. The two methods agreed the least on the largest two networks, namely *M. musculus* and *R. norvegicus*. This is interesting, given that these two species are phylogenetically very close; they belong to the same subfamily. Thus, they are expected to have very similar sets of proteins and interactions. Focusing on the mappings that the two alignments disagree (i.e., novel mappings found by our method) reveals numerous significant mappings that are missed by IsoRank. Here, we present a few interesting cases.

Table 2 shows the gene names of the proteins aligned with the proteins in this subnetwork using weighted IsoRank and our method. We use this example for mainly two reasons. First, the two alignment methods disagree on almost all
Nodes of this connected subnetwork. Second, the proteins in this network serving in many pathways, including signaling networks such as MAPK, TGF-beta signaling pathways; metabolic networks such as glycolysis and glycogenesis pathways; disease networks such as Alzheimer's and Parkinson's disease. Thus, they are important in regulating many critical functions. Our method successfully maps the proteins corresponding to the same gene in the two species for almost all the proteins, whereas IsoRank failed in doing that for four out of six proteins. Both methods aligned the proteins corresponding to the same gene in the two species for almost all the proteins, whereas IsoRank failed in doing that for four out of six proteins. Both methods aligned the proteins for Gapdh with a protein for a different gene. In that for four out of six proteins. Both methods aligned the proteins for Gapdh with a protein for a different gene. In that for four out of six proteins.

Our method disagreed with the weighted IsoRank greatly on the alignment of the PPI networks of M. musculus and S. pombe as well (see Table 2). We present the gene names of the proteins of a small subnetwork from E. coli O157 in Table 4 and the network topology in Fig. 6. The protein for gene rpoA interacts with all the other proteins in this subnetwork. In the same table, we also show the gene names of the proteins from the S. pombe PPI network that is aligned with those proteins using our method and using weighted IsoRank. The S. pombe proteins in this example have the same network topology as those of E. coli O157 for both our method and weighted IsoRank.

**TABLE 3**

<table>
<thead>
<tr>
<th>Organism</th>
<th>Method</th>
<th>Genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>E. coli O157</td>
<td>Mapk1</td>
<td>P63086, P21708, P0A854V4</td>
</tr>
<tr>
<td>S. pombe</td>
<td>Mapk3</td>
<td>P21708, P0A7Z6, P0A94B</td>
</tr>
<tr>
<td>M. musculus</td>
<td>Sla2a4</td>
<td>P19357, P4797, P0A854V4</td>
</tr>
<tr>
<td>R. norvegicus</td>
<td>Gapdh</td>
<td>P47196, P0A854</td>
</tr>
<tr>
<td>H. sapiens</td>
<td>Akt1</td>
<td>O70437, P47196</td>
</tr>
</tbody>
</table>

The mappings that the two methods disagree are in **bold**.

We make two observations from Table 4. First, both methods align proteins for different genes from the two organism. Thus, we need to understand whether this mapping has any significance. The proteins for rpb1, rpb2, and rpb3 are all DNA-directed RNA polymerase II subunits. They participate in Purine and Pyrimidine metabolisms and in RNA polymerase. They form the enzyme complex EC:2.7.7.6. The proteins for rpoA, rpoB, and rpoC belong to DNA-directed RNA polymerase subunits. They form the same enzyme complex and participate in the same reaction of the same pathways as rpb1, rpb2, and rpb3. We make similar observations for tdh1 and gapA. Both proteins are glyceraldehyde-3-phosphate dehydrogenase and both execute the same reactions in the same pathways, namely, glycolysis/gluconeogenesis, biosynthesis of secondary metabolites, and microbial metabolism pathways. Thus, the alignments found by the two methods map functionally identical proteins with each other almost all the time. The two methods disagree in two out of five proteins. Among these, rpb1 (found by our method) is the relevant mapping. Weighted IsoRank aligned rad1, which is a cell cycle checkpoint, and has a significantly different set of functions than rpoA. Both rpb6 and cdc42 have different functions than ciaA, and this mapping needs closer investigation.

The result discussed above is a small example among many. These results suggest that our method is considerably powerful in mapping highly relevant proteins. Detailed and further results are available at http://bioinformatics.cise.ufl.edu/probAlign.html.

In terms of quantitative evaluation of alignments of entire networks, Table 5 presents comparative results obtained on some of the larger networks. The evaluation is based on the three metrics: functional enrichment, functional coherence, and statistical coherence, applied to the alignments obtained by the two methods: probabilistic alignment (P) and weighted IsoRank (W). We observe that as the size of the networks increase, the difference between the results evaluated by these metrics becomes smaller. This means that we gain more benefits from applying our method to smaller networks.

We have also experimented by aligning a pathway of one organism with the entire network of another organism. In Fig. 7, we report results obtained by aligning the seven biggest deterministic pathways of H. sapiens with the entire probabilistic network of M. musculus. The graph shows for each pathway the percentage of proteins from the H. sapiens pathway that were matched with proteins of M. musculus belonging to the homologous pathway, when using both our method and IsoRank. We see that in most cases our method obtains a higher percentage.
3.6 Running Time Analysis

So far, we have evaluated our method qualitatively and demonstrated that it can find novel alignments that are biologically significant. This, however, comes at the expense of computing the expected support matrix over all possible network topologies. A biologically relevant alignment method is preferable only if it can be used in practical running time. This leads us to the last question that needs to be answered: How fast (or slow) is our method as compared to the deterministic approaches?

3.6.1 Results on the Partitioned Data Set

We evaluate the running time of our method as well as the weighted and unweighted IsoRank methods. The running times of both weighted and unweighted IsoRank were similar. The unweighted IsoRank was slightly faster, as ignoring weights simplifies the computation of the support matrix. So, we compare our method to the faster of the two unweighted IsoRank.

In this experiment, we performed alignments between all pairs of networks in our partitioned data set without excluding the small networks and measured the running time for each method and each query. We eliminated the pairs of networks that lead to less than 1 millisecond running time, as these times cannot be measured accurately. Thus, we obtained about 31,000 alignments. Fig. 8 shows the results for all of these queries. The plot demonstrates that our method runs in practical time for all queries. It completes alignment in 20 seconds or less in all experiments. In more than 99 percent of the experiments, our method took less than half a second to align two networks. Furthermore, the running time of our method is comparable to that of the deterministic one. On average, our method took only 47 percent longer time than the deterministic method. In 66 and 82 percent of all the experiments, our method took less than twice and three times more time as compared to the deterministic method, respectively. This is a very important result, as it shows that our formulation in Sections 2.2 and 2.3 allows us to precisely compute the expected support matrix over an exponential number of alternative deterministic topologies in a small amount of time.

3.6.2 Results on Entire Networks

In this experiment, we evaluate the running time of our method for aligning entire networks for an organism. The purpose is to observe how our method scales with increased network size. To observe this in a systematic way, we aligned the entire network for an organism obtained from MINT with itself. This way, we know that both input networks to be aligned have exactly the same number of nodes and edges. We repeated this experiment for organisms of different sizes. These organisms were chosen from a broad spectrum of network sizes to demonstrate how the performance changes with increasing network size on real data sets.

Table 6 reports the results. We have split the computation into four phases: computing the topological matrix (its expected value for the probabilistic method), reading precomputed pairwise protein alignment scores, performing alignment iterations and maximum bipartite graph matching. As expected, our method is slower in the first phase, the times are equal in the second phase and there is no rule for which method is faster in the last two phases, since both convergence of the power method and the number of iterations in graph matching depend on the output of the previous methods in an unpredictable way. Overall, the results show that our method can run for
entire networks with more than a thousand probabilistic interactions in less than 4 minutes on a single processor. This is very promising, especially given the fact that we are precisely computing the expected support matrix for a number topologies that is exponential in the number of probabilistic interactions. This leads to a conclusion beyond the goal of this paper: Our PGF strategy can efficiently measure the degree distribution of a large number of probabilistic interactions. Thus, any computational analysis problem on probabilistic networks that require degree distribution can use our PGF strategy efficiently. We also observe that the running time for *E. coli* is more than that for *H. pylori*, although the latter has more nodes. This is because both the number of edges and the number of nodes contribute to the running time.

4 Conclusion and Further Discussion

In this work, we presented a method to solve an important problem yet unaddressed by the literature: How can we incorporate interaction probabilities in the analysis of biological networks? More specifically, we developed a network alignment algorithm that uses interaction probabilities to discover more meaningful alignments. Our method makes use of the powerful mathematical concept of probability generating functions to efficiently compute node degree distributions in complex networks. These distributions are subsequently used as topological data in the alignment process. The small computational price is entirely justified by the improvement in the quality of the alignments. This improvement becomes evident when evaluated under our statistical coherence measure, which takes into account individual pairwise matches between proteins in the two networks. Our conclusion is that the claims for the need and usefulness of treating biological interactions in a probabilistic setting, laid out in the preliminary discussion, are entirely validated by our experimental results.

Alternative models for a probabilistic network could be considered. For example, we consider all interactions independent of each other. A more general treatment would not make this assumption, but it is common in the literature [24], [10], [37]. Moreover, typical models that deal with the presence of dependency among interactions are the stoichiometric model, the S-model and GMA-models (see [35]), but they are not suitable for our purposes because

1. They do not describe the topology of the network. They rather describe the state of the network. In the steady state, since the fluxes do not change anymore, it becomes safe to assume independence among interactions.
2. They require additional parameters that require costly wet-lab experimentation and, thus, are not available for many real biological networks.

Future work will address the case were both graphs are probabilistic, not only one of them. A further extension will be multiple probabilistic network alignment, which will face the challenge of increasing complexity, because the loop nesting level increases by one with each added network to be aligned.

Acknowledgments

This work was partially supported by the US National Science Foundation under grants CCF-0829867 and IIS-0845439.

References
