A Linear-time Algorithm for Predicting Functional Annotations from Protein Protein Interaction Networks*

Yonghui Wu
Department of Computer Science & Engineering
University of California
Riverside, CA 92507
yonghui@cs.ucr.edu

Stefano Lonardi*
Department of Computer Science & Engineering
University of California
Riverside, CA 92507
stelo@cs.ucr.edu

ABSTRACT

Recent proteome-wide screening efforts have made available genome-wide, high-throughput protein-protein interaction (PPI) maps for several model organisms. This has enabled the systematic analysis of PPI networks, which has become one of the primary challenges for the system biology community. Here we address the problem of predicting the functional classes of proteins (i.e., GO annotations) based solely on the structure of the PPI network. We present a maximum likelihood formulation of the problem and the corresponding learning and inference algorithms. The time complexity of both algorithms is linear in the size of the PPI network and experimental results show that their accuracy in the functional prediction outperforms current existing methods.

1. INTRODUCTION

High-throughput protein-protein interaction (PPI) networks with various levels of proteome coverage are currently available for several model organisms, namely *S. cerevisiae* [19], *D. melanogaster* [7, 6], *C.elegans* [12], *H. sapiens* [15] and *H. pylori* [14]. PPI data can be obtained through a variety of sophisticated assays, like co-immunoprecipitation, yeast two hybrid, tandem affinity purification and mass spectrometry. A PPI network is usually represented by a node-labeled undirected graph where vertices correspond to proteins and edges denote physical interactions.

Since the main mechanism by which cells are able to process information is through protein-protein interactions, PPI data has been essential to obtain new knowledge and insights in a wide spectrum of biological processes. In this paper, we focus on the problem of predicting the functional category of proteins *solely* based on the topological structure of the PPI network. The rationale of this approach is based on the observation that a protein is much more likely to interact with another protein in the same functional class than with a protein with a different function (see, e.g., [10, 21, 18, 13]). The prediction of functional classes can be useful either for proteins for which there is little or non-existing functional information (e.g., for predicting the involvement of a protein in specific pathway), or to confirm existing annotations provided by other methods. Motivated by the expectation that in the near future massive PPI networks will be available, here we propose a *computationally efficient* method that accurately determines the functional categories and will be capable to scale gracefully with the size of the network.

A variety of algorithmic techniques have been proposed in the literature to solve the problem of functional prediction with a wide range of computational complexity. Perhaps the most computationally efficient algorithm is based on the *majority rule* where the function of an unknown protein is simply determined by the most common function among its interacting partners [17]. A slightly more sophisticated majority-based method is the χ^2-method proposed in [8]. At the other end of the computational complexity spectrum, the authors of [21, 10] propose to assign proteins to functional classes so that the number of protein interactions among different functional categories is minimized. The optimization problem, known as *generalized multicut*, is NP complete.

The *functional flow* algorithm introduced in [13] lays somewhere in the middle of the complexity spectrum. The idea is to treat proteins with known function as infinite sources of (functional) flow. The flow is propagated through the network in a series of discrete steps. At the end, the function of unknown proteins is assigned based on the largest amount of flow received. The authors of [13] show that functional flow algorithm outperforms the generalized multicut algorithm, the majority rule-based algorithm and also its generalization to more distant neighbors [13]. The authors of [2] show that functional flow also outperforms the χ^2-method. Because of this, the performance of functional flow is the reference for our algorithm. Experimental results will show that our method achieves a better prediction accuracy than functional flow.

Perhaps the most similar method to the one we propose here is described in [4, 5], where the authors propose a probabilistic model based on the theory of Markov random fields. In their follow-up papers [3], Deng *et al* show how to integrate in their Markov random field additional information, namely gene expression data, protein complex information, domain structures to increase the prediction accuracy. The relationship between this work and [4, 5] will be discussed in greater detail later in paper. Here, however, we want to emphasize that the method presented in this manuscript is

*This project was supported in part by NSF CAREER IIS-0447773, and NSF DBI-0321756.
*Corresponding author

BIOKDD'07 , August 12, 2007, San Jose, California, USA

computationally more efficient than Deng *et al.* Unfortunately, the accuracy of their prediction cannot be directly compared with ours because these methods predict multiple functional classes for each protein. The approach in [11] is essentially similar to [5].

More recent papers tackle slightly different albeit related problems. In [18] the authors predict functional linkages between proteins based on the integration of four kinds of evidence, namely gene co-expression, gene co-inheritance, gene co-location and gene co-evolution. In [9], the authors predict protein interactions based on the cellular localization of proteins.

2. PROBLEM DEFINITION AND MODEL FORMULATION

We denote by $G(V, E)$ the PPI network under analysis, where V represents the set of proteins and E is the set of edges (interactions). For reason that will be clear later in the paper, we assume G to be directed (i.e., each undirected edge in the original PPI is represented by two directed edges, except for self-loops). We denote the set of k given functional classes as $\mathcal{F} = \{C_1, C_2, \ldots, C_k\}$. Each functional class can be thought as one of k possible colors that can be used to color the graph. Function $f : V \rightarrow \mathcal{F}$ captures the notion of functional class for all the proteins in V. When the function of a protein $v \in V$ is known, say C_i, then we will have $f(v) = C_i$. If the function of v is unknown, then $f(v) = \emptyset$. We define $W = \{v \in V : f(v) \in \mathcal{F}\}$ to be the set of proteins whose function is known and $U = V \setminus W$ to be the set of the proteins whose function is unknown. The functional annotation problem can be informally stated as follows. Given a PPI network $G(W \cup U, E)$ where W is annotated with functional classes, find the correct functional classes for the vertices in U.

The model used here to tackle the problem is entirely probabilistic and it is based on two simple observations. First, a simple statistical analysis on the available PPI data [16] and the associated GO functional annotations [1] reveals that the distribution associated with the functional classes is highly skewed. For example, in the *S. cerevisiae* network, the function "catalytic activity" is assigned to 1,514 proteins, whereas the function "protein tag" is only assigned to 5 proteins. This observation constitutes our prior knowledge on the probability of a randomly chosen protein to perform a certain function and can be captured by the notion of *prior distribution*. We denote the prior distribution by $\mathcal{P} : \mathcal{F} \rightarrow [0, 1]$, where $\mathcal{P}(C_i)$ is the probability of a randomly chosen protein to have function C_i.

Second, our model has to incorporate the connectivity structure of the PPI networks. It is well-known that a protein is more likely to interact with another protein performing the same function [10, 21, 18, 13]. We model this preference using conditional probability distributions. If protein $t \in W$ has function C_i and protein $s \in U$ interacts with t, then the probability that s performs function C_j is given by $\mathbf{P}(C_j|C_i)$. We expect $\mathbf{P}(C_i|C_i)$ to be higher than $\mathbf{P}(C_j|C_i), \forall j \neq i$, because s is more likely to perform the same function of t. This can be easily generalized to multiple interacting partners. Suppose we want to predict the function of protein $s \in U$ and that we know that $t_1, t_2, t_3, \ldots, t_m \in W$ interact with s, as well as their functions $f(t_1), f(t_2), f(t_3), \ldots, f(t_m)$. If we assume that $f(t_1), f(t_2), f(t_3), \ldots, f(t_m)$ are independent and distributed according to the conditional multinomial distribution $[\mathbf{P}(C_1|f(s)), \mathbf{P}(C_2|f(s)), \mathbf{P}(C_3|f(s)), \ldots, \mathbf{P}(C_K|f(s))]$, then the most likely function for s is the one that maximizes

$$\begin{aligned} L(s) &= \mathcal{P}(f(s)) \prod_{t \in \{t_1, t_2, \ldots, t_m\}} \mathbf{P}(f(t)|f(s)) \\ &= \mathcal{P}(f(s)) \prod_{t \in V : (s,t) \in E} \mathbf{P}(f(t)|f(s)) \end{aligned}$$

We call $L(s)$ the *local likelihood* of protein s.

Note that a necessary condition to predict the functional class for $s \in U$ is to know the functional classes of the neighbors of s. Very often, however, the functions of the neighbors turns out to be unknown. Clearly, the assignment of a function to protein s may affect the prediction of the functions for the neighbors of s, and vice versa. Because of this, a purely local strategy is insufficient. To address this problem, we need to introduce the concept *global likelihood* of a PPI Network as $L(G) = \prod_{v \in V} L(v)$.

The free variables in the global likelihood function $L(\cdot)$ are $f(u_i)$, for all proteins $u_i \in U$ with unknown function. We seek the assignment to $f(u_i)$ such that the global likelihood $L(G)$ is maximized, which is equivalent to maximizing

$$l(G) = \sum_{v \in V} \log(\mathcal{P}(f(v))) + \sum_{(v,w) \in E} \log(\mathbf{P}(f(w)|f(v)))$$

Now we are ready to give a formal summary of the optimization problem associated with our model. We are given a directed PPI network $G(W \cup U, E)$ where U is the set of proteins with unknown functions and W is the set of proteins with known functions, a set of functions $\mathcal{F}$, a prior distribution $\mathcal{P}$ with $\sum_{C_i \in \mathcal{F}} \mathcal{P}(C_i) = 1$, and the conditional distributions $\mathbf{P}(C_i|C_j)$ such that $\sum_{C_i \in \mathcal{F}} \mathbf{P}(C_i|C_j) = 1, \forall C_j \in \mathcal{F}$. The problem is to predict the functional class $f(u)$ for each protein in set U, such that the global log likelihood $l(G)$ is maximized.

3. RELATION TO PREVIOUS WORK

Our model implicitly defines a Markov random field (MRF), a probabilistic model which is also used in [4, 5]. In Deng *et al.*'s works [4, 5], a distinct MRF is built for each functional class in $\mathcal{F}$. Each protein in the PPI network is associated to an indicator random variable for that function of interest. More specifically, each protein is associated with a unary potential $e^{\phi(X_i)}$, which has value $e^{\phi(1)}$ if the protein has that function and $e^{\phi(0)}$ otherwise. Each edge of the PPI graph is associated with a binary potential $e^{\psi(X_i, X_j)}$, which can take three possible values, namely $e^{\psi(1,1)}$ if both of the proteins have the function, $e^{\psi(0,1)}$ if one of the proteins has the function, and $e^{\psi(0,0)}$ if neither of the proteins has the function. Given the parameters $\theta = \{\phi(0), \phi(1), \psi(1,1), \psi(0,1), \psi(0,0)\}$, the global Gibbs distribution of the entire network is simply the product of the unary potentials and the binary potentials normalized by a constant factor depending on the parameters, as follows.

$$P\{X_1, X_2, X_3, \ldots, X_n|\theta\} = e^{\sum_{i=1}^{n} \phi(X_i) + \sum_{(i,j) \in E} \psi(X_i, X_j)} / Z(\theta)$$

Note that in our model, the prior probability $\mathcal{P}(f(v_i))$ corresponds to the unary potential in Deng's model, whereas the product $\mathbf{P}(f(v_i)|f(v_j))\mathbf{P}(f(v_j)|f(v_i))$ corresponds to the binary potential.

Despite the similarities, there are significant differences between Deng *et al.*'s model and ours. First, instead of building a distinct MRF for each function, we only have one unified probabilistic model for all the functions in $\mathcal{F}$ which allows us to capture the correlations between the functions. Second, the use of conditional distributions dramatically simplifies the process of estimating the parameters, which boils down to a simple count of relevant statistics (details to be explained in Section 4). The semantics of the conditional distributions also naturally gives rise to the efficient iterative algorithm that we will develop later. Finally, since we are modeling from the conditional distributions, the normalization factor of the global Gibbs distribution in our model is always one irrespective of the parameters we use.

A less obvious connection can be established between our model and the generalized multi cut approach by Vazquez *et al.* [21]. Recall that in this latter approach, the objective is to assign functional annotations to unknown proteins in such a way that one minimizes the number of times neighboring proteins have different annotations. A formal description of the generalized multi cut problem follows. Let I be the standard indicator function which is equal to 1 if the boolean expression is true and 0 otherwise. Given a PPI network $G(U \cup W, E)$ we seek annotations to the proteins in U such that $\sum_{(u,v)\in E} I(f(u) \neq f(v))$ is minimized.

FACT 1. *The generalized multi cut problem is a special case of our optimization problem when the prior distribution is uniform and most of the mass of the conditional probabilities is concentrated around* $\boldsymbol{P}(C_i|C_i)$.

Proof. Let us consider the following prior distribution and conditional distributions.

$$\begin{aligned}
\mathcal{P}(C_i) &= 1/|\mathcal{F}| && \forall C_i \in \mathcal{F} \\
\mathbf{P}(C_j|C_i) &= \epsilon && \forall C_i, C_j \in \mathcal{F}, C_i \neq C_j \\
\mathbf{P}(C_i|C_i) &= 1-(|\mathcal{F}|-1)\epsilon && \forall C_i \in \mathcal{F}
\end{aligned}$$

where $0 < \epsilon < 1$ is an arbitrarily small number. Then, the global log likelihood for the graph can be written as

$$\begin{aligned}
& l(G(V,E)) \\
&= \sum_{v\in V} \log(\mathcal{P}(f(v))) + \sum_{(v,w)\in E} \log(\mathbf{P}(f(w)|f(v))) \\
&= \sum_{v\in V} \log(1/|\mathcal{F}|) + \sum_{(v,w)\in E f(w)\neq f(v)} \log(\mathbf{P}(f(w)|f(v))) \\
&\quad + \sum_{(v,w)\in E f(w)=f(v)} \log(\mathbf{P}(f(w)|f(v))) \\
&= |V|\log(1/|\mathcal{F}|) + \sum_{(v,w)\in E f(w)\neq f(v)} \log(\epsilon) \\
&\quad + \sum_{(v,w)\in E f(w)=f(v)} \log(1-(|\mathcal{F}|-1)\epsilon) \\
&= |V|\log(1/|\mathcal{F}|) + |E|\log(1-(|\mathcal{F}|-1)\epsilon) \qquad (1) \\
&\quad +(\log(\epsilon) - \log(1-(|\mathcal{F}|-1)\epsilon)) \sum_{(v,w)\in E} I(f(v)\neq f(w))
\end{aligned}$$

Note that the first two terms of (2) are constant and that the third term increases as the quantity $\sum_{(v,w)\in E} I(f(v) \neq f(w))$ decreases because $\log(\epsilon) - \log(1-(|\mathcal{F}|-1)\epsilon)$ is negative for a sufficiently small ϵ. Therefore, under this particular prior distribution and conditional distributions, maximizing the global log likelihood in our problem is equivalent to minimizing the objective function in the generalized multicut problem. ∎

The generalized multicut problem is NP complete [13] because it is a generalization of the multi-way cut problem [20], which is known to be NP complete. Since our problem is a generalization of the generalized multicut problem, it is NP complete as well.

4. PARAMETER LEARNING

The prior distribution and the conditional distributions are multinomial distributions whose parameters can be learned from the structure of the given PPI network and the functional annotations on W. We need to determine $k-1$ parameters for the prior and $k(k-1)$ parameters for the k conditional distributions. We obtain these parameters using the maximum likelihood estimation method.

Let $F(W, E')$ be the subgraph of $G(V, E)$ induced by the set W of known functions, where $E' = \{(u,v)|(u,v) \in E, u \in W, v \in W\}$. The global likelihood for the subgraph $F(W, E')$ is defined as follows.

$$\begin{aligned}
& L(F(W,E')) \\
&= \prod_{v\in W} \mathcal{P}(f(v)) \prod_{(u,v)\in E'} \mathbf{P}(f(v)|f(u)) \\
&= \prod_{C_i\in\mathcal{F}} \mathcal{P}(C_i)^{\sum_{v\in W} I(f(v)=C_i)} \qquad (2) \\
&\quad \prod_{C_i\in\mathcal{F}} \prod_{C_j\in\mathcal{F}} \mathbf{P}(C_j|C_i)^{\sum_{(v_i,v_j)\in E'} I(f(v_i)=C_i, f(v_j)=C_j)}
\end{aligned}$$

The first term in (3) is maximized when $\mathcal{P}(C_i) = \sum_{v\in W} I(f(v) = C_i)/|W|$ for all $C_i \in \mathcal{F}$. The second term in equation (3) is maximized when $\mathbf{P}(C_j|C_i) = \frac{\sum_{(v_i,v_j)\in E'} I(f(v_i)=C_i, f(v_j)=C_j)}{\sum_{(v_i,v_j)\in E'} I(f(v_i)=C_i)}$ for all $C_j \in \mathcal{F}$. Therefore, the maximum likelihood estimates for the parameters are

$$\mathcal{P}(C_i) = \sum_{v\in W} I(f(v)=C_i)/|W| \quad C_i \in \mathcal{F}$$

$$\mathbf{P}(C_j|C_i) = \frac{\sum_{(v_i,v_j)\in E'} I(f(v_i)=C_i, f(v_j)=C_j)}{\sum_{(v_i,v_j)\in E'} I(f(v_i)=C_i)} \quad C_i, C_j \in \mathcal{F}$$

As a common practice in Bayesian statistics, we apply (uniform) Dirichlet priors to our estimators. This prevents the problem of handling zero probabilities. The time complexity of the learning phase is $O(|E| + |W|)$, whereas the space complexity is $O(k^2)$.

5. INFERENCE OF FUNCTIONAL CLASSES

Since we determined that our problem is NP complete, it is rather unlikely that we will find a polynomial time algorithm that can solve the problem optimally. To this end, we designed a statistically based iterative algorithm (SBIA for short), which turns out to perform well in practice. Our algorithm consists of two phases, namely the initialization

phase and the iterative phase. The initialization phase consists of two steps. In the first step, we estimate the parameters for the prior distribution and the conditional distributions as described in Section 4. In the second step, we assign an initial functional class to each protein in V, as follows.

For each unknown protein $v \in U$, we assign

$$f^0(v) = argmax_{C_i \in \mathcal{F}} \ \mathcal{P}(C_i) \prod_{(v,t) \in E, t \in W} \mathbf{P}(f^0(t)|C_i).$$

In other words, we predict the initial function for v to be the one that maximizes the local likelihood of v (ignoring neighbors with unknown functions). If $v \in W$, then we set $f^0(v)$ to be the function corresponding to annotation in the original data.

In the second phase, we iteratively re-evaluate our predictions. For clarity of exposition we use superscripts to denote the iteration number, i.e., $f^n(v)$ denotes the predicted functional class for v made in the n^{th} iteration. For each unknown protein $v \in U$, we set

$$f^n(v) = argmax_{C_i \in \mathcal{F}} \ \mathcal{P}(C_i) \prod_{(v,t) \in E} \mathbf{P}(f^{n-1}(t)|C_i).$$

That is, we adjust our prediction for protein v to be the function that maximizes the local likelihood with respect to the functions predicted for its neighbors in the previous step. Again, if $v \in W$, then $f^n(v) = f^{n-1}(v)$.

We stop the iterative process as soon as the difference between the value of the global likelihood in two consecutive steps drops below a given threshold. The pseudo-code in Figure 1 summarizes the algorithm. The time complexity of the algorithm is $O(d|E|)$, where d represents the number of iterations (usually $d \leq 5$ in our experiments).

6. EXPERIMENTAL RESULTS

The dataset used in our experimental studies is the most well-characterized PPI network available at the time of writing, namely the network for *S. cerevisiae*, which is composed of 4,959 proteins and 17,511 interactions. The network was obtained from the DIP database [16]. We also extracted a *high confidence* yeast PPI network, which is a subset of the yeast PPI network in which interactions that are confirmed by only a single experiment have been removed. This latter network has 1,735 proteins and 2354 interactions. The functional annotations were obtained from the Gene Ontology (GO) hierarchy [1].

We used cross validation to quantitatively evaluate the prediction accuracy of our algorithm and to compare its performance with other methods. In each experiment, we randomly removed the functional annotation to a percentage p of known proteins, where p ranges from 5% to 95%. This new set of "unknown" proteins served as the test set, called hereafter T. We use $W \setminus T$ to denote the set of known proteins after p% of them have been "un-labelled" and U to denote the set of the remaining unknown proteins. Clearly, the SBIA's learning phase (i.e., the computation of the prior and the conditional probabilities) is carried out only on the proteins in $W \setminus T$. Learning on the original set W would constitute "cheating".

So far, in our model we assumed that each protein can perform only one function. This is, however, not true for some proteins. A protein may participate in multiple biological processes and as a result, it will carry out multiple functions. In the yeast network, 488 proteins out of 3,022 are annotated with two or more top level functions. To handle this issue, the nodes in $W \setminus T$ that are associated with multiple functions are replicated, so that each copy carries out exactly one of the annotated functions. Each copy has the same interaction partners of the original protein.

As said, the goal is to predict a function for each of the proteins in set $T \cup U$, based on the functional classes in $W \setminus T$ and the topology of the graph. For each protein in T, we declare a prediction to be correct if the predicted function is one of the functions the protein was originally assigned. The prediction accuracy is calculated as the ratio between the number of correct predictions and the total number of proteins in the set T. Since the prediction accuracy varies slightly every time we randomly select T, we replicate the same experiments ten times and compute the average accuracy. We also record the standard deviation, represented by the error bars in the figures.

We compared the accuracy of our method against that of functional flow [13] and against that of the *naive* approach. We chose to compare SBIA against the functional flow method because papers [2, 13] report that functional flow outperforms both majority-rule based methods [17, 8] as well as methods based on the generalized multicut [21, 10]. As said, a direct comparison between our method and MRF-based methods [4, 5, 11] is not feasible because these latter approaches predict more than one functional class for each protein. The naive method simply predicts the function of a protein to be the most probable functional class according to the prior, i.e., $argmax_{C_i \in \mathcal{F}} \mathcal{P}(C_i)$. Clearly, the expected prediction accuracy of the naive approach is equal to the ratio between the number of proteins annotated with the most probable function and the total number $|W|$ of known proteins.

We carried out two sets of experiments. In the first set, we considered the seventeen top level molecular functions defined in GO. In the yeast PPI network, 3,022 proteins out of 4,959 are annotated with one or more top level functions. The most frequent function is "catalytic activity", which occurs 1,514 times. Thus, the expected prediction accuracy for the naive approach is 0.501 or 50%. In the high confidence yeast PPI network 1,325 proteins are annotated. The most frequent function in this network is again "catalytic activity", which is assigned to 568 proteins. The statistics of the networks constituting the dataset are summarized in Table 1.

Figure 2-left and 3-left summarize the results of the first set of experiments on the seventeen functional classes in the top level of the GO hierarchy. The figures show that SBIA always outperforms functional flow, especially when p is large. In the yeast network, the prediction accuracies of the functional flow algorithm even falls below that of the naive approach when p is greater than 55%. SBIA, however, still retains good prediction accuracy until p becomes higher than 70%, and then asymptotically converges to that of the naive approach. Notice that the initialization phase of SBIA already achieves a good prediction accuracy. When p is less than 80%, the iterative phase improves the prediction accuracy even more, along with the global likelihood of the graph. The number of iterations executed is usually rather small, less than 5. When p is greater than 80%, the information left in the network is highly incomplete, and as expected the performance of our algorithm falls back to

SBIA:

- Input:
  1. $G(V, E)$, where $V = U \cup W$. W is the set of known problems and U is the set of unknown proteins.
  2. $\mathcal{F}$, the set of functions.
  3. $f : W \rightarrow \mathcal{F}$, the annotations on the proteins in W.
- Output:
  1. $f : U \rightarrow \mathcal{F}$, the predicted function for the proteins in U.
- Initialization phase
  1. Estimate $Pri(C), P(C_i|C_j), C, C_i, C_j \in \mathcal{F}$ as suggested in section 4.
  2. **For** v in V:
     IF $(v \in U)$ $f(v) = argmax_{f(v) \in \mathcal{F}} Pri(f(v)) \prod_{(v,t) \in E, t \in W} P(f(t)|f(v))$;
- Iterative phase
  1. **DO**:
     FOR v in W: $f'(v) = f(v)$
     FOR v in U: $f'(v) = argmax_{f'(v) \in \mathcal{F}} Pri(f'(v)) \prod_{(v,t) \in E} P(f'(t)|f'(v))$
     $L(G) = (\prod_{v \in V} Pri(f(v))) \cdot (\prod_{(v,w) \in E} P(f(w)|f(v)))$
     $L'(G) = (\prod_{v \in V} Pri(f'(v))) \cdot (\prod_{(v,w) \in E} P(f'(w)|f'(v)))$
     IF $L'(G) >= L(G)$:
     FOR v in V: $f(v) = f'(v)$
     WHILE $(L'(G) > L(G))$
  2. **RETERN** $f : U \rightarrow \mathcal{F}$

Figure 1: Pseudo code of our Statistically Based Iterative Algorithm(SBIA)

Table 1: The statistics of the PPI networks used in the experiments. $|V|$ is the number of proteins in the network, $|E|$ is the number of interactions, $|W|$ is the number of known proteins, and *naive expected* is the expected prediction accuracy of the naive approach (see text).

			17 functional classes		190 functional classes	
organism	$\|V\|$	$\|E\|$	$\|W\|$	*naive expected*	$\|W\|$	*naive expected*
yeast	4,959	17,511	3,022	0.5010	2930	0.1939
yeast high confidence	1,735	2,354	1,325	0.4286	1278	0.1979

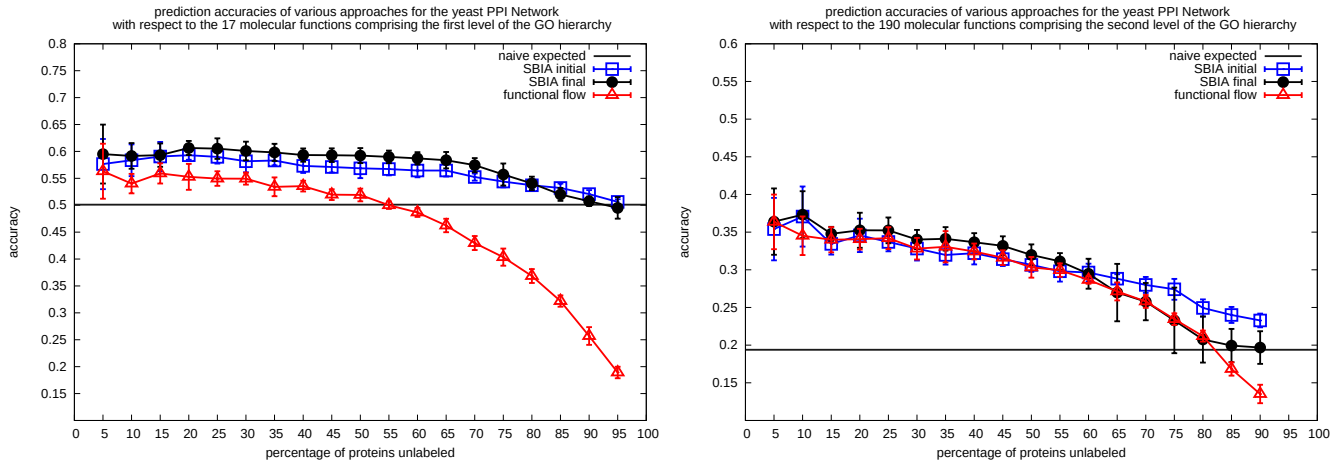

Figure 2: Prediction accuracies on the yeast PPI network with respect to the 17 functional classes at the first level of the GO hierarchy (right) and 190 functional classes at the second level of the GO hierarchy (left). The x-axis represents the percentage of known proteins on which the algorithms are tested. The "naive expected" line indicates the expected prediction accuracy of the naive approach. "SBIA initial" refers to the accuracy of SBIA after the initialization phase, whereas "SBIA final" shows the final accuracy of SBIA. "Functional flow" denotes the prediction accuracy of the functional flow algorithm

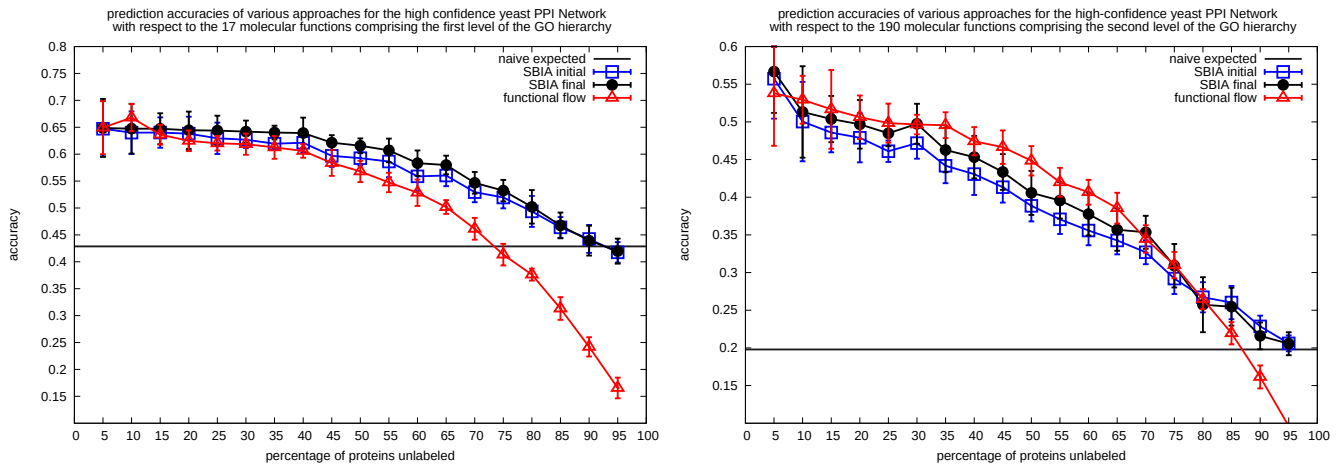

Figure 3: Prediction accuracy on the yeast high confidence PPI network (see caption of Figure 2 for more details). LEFT: 17 functional classes, RIGHT: 190 functional classes.

that of the naive approach. Due to the higher quality of the data in the yeast high confidence network, the improvement in accuracy of our algorithm and functional flow relative to the naive approach is almost doubled.

In the second set of experiments, we considered all the 190 molecular functions comprising the second level of the GO hierarchy. In the yeast network, 2,930 proteins out of 4,959 yeast proteins are annotated with one or more second level molecular functions. The most prevalent function is "hydrolase activity", which appears 568 times. Hence the expected prediction accuracy for the naive approach is 0.1939. In the high confidence yeast network, 1,278 out of 1,735 proteins are annotated. The most prevalent function is "protein binding", which is annotated to 253 proteins. The statistics are summarized in Table 1.

Figure 2-right and 3-right summarize the second set of experimental results. In Figure 3-right, the functional flow algorithm outperforms SBIA by 2-3% on average. We suspect that this is due to the relatively small size of the network (containing about 1,300 characterized proteins) under consideration and the large number of functions ($k = 190$). Recall that the number of parameters of our model is $\Theta(k^2)$. In this case, we believe that there is not enough data for the accurate estimation of the parameters for the prior distribution and the conditional distributions. For the yeast PPI network, the result is similar to that in the previous set of experiments. SBIA still outperforms functional flow, but the difference between the two approaches is not as strong as in the previous case.

7. CONCLUSIONS

We developed an efficient algorithm to assign functional GO terms to uncharacterized proteins on a PPI network based solely on the topology of the graph and the functional labels of known proteins. The statistical model proposed in this paper is a generalization of the GenMultiCut model and resemble the MRF-based model by Deng *et.al.* The similarity with the work of Deng *et.al.* is, however, superficial as we discussed in details in the paper. In particular, the structure of our model allows one to obtain easily and efficiently the maximum likelihood estimation of the underlying parameters, which is tipically not possible for a general MRF. Based on our statistical model, we presented efficient learning and inference algorithms. Our inference algorithm is an iterative algorithm, where each iteration runs in time linear in the size of the input. According to our experimental results, our algorithm converges very quickly to a local optimum. More importantly, our method gives consistently better predictions when compared with previous known algorithms.

8. REFERENCES

[1] Ashburner, M., Ball, C. A., and *et al*. Gene ontology: tool for the unification of biology. *Nature Genetics 25* (2000), 25–29.

[2] Chua, H. N., Sung, W.-K., and Wong, L. Exploiting indirect neighbours and topological weight to predict protein function from protein-protein interactions. *Bioinformatics 22* (2006), 1623 – 1630.

[3] Deng, M., Chen, T., and Sun, F. An integrated probabilistic model for functional prediction of proteins. *Journal of Computational Biology 11*, 2/3 (2004), 463–475.

[4] Deng, M., Tu, Z., Sun, F., and Chen, T. Mapping gene ontology to proteins based on protein-protein interaction data. *Bioinformatics 20*, 6 (2004), 895–902.

[5] Deng, M., Zhang, K., Mehta, S., Chen, T., and Sun, F. Prediction of protein function using protein-protein interaction data. *Journal of Computational Biology 10*, 6 (2003), 947–960.

[6] Formstecher, E., Aresta, S., and *et al*. Protein interaction mapping: A drosophila case study. *Genome Res. 15*, 3 (2005), 376–384.

[7] Giot, L., Bader, J. S., and *et al*. A protein interaction map of *Drosophila melanogaster*. *Science 302*, 5651 (2003), 1727–1736.

[8] Hishigaki, H., Nakai, K., Ono, T., Tanigami, A., and Takagi, T. Assessment of prediction accuracy of protein function from protein–protein interaction data. *Yeast 18*, 6 (2001), 523–531.

[9] Jaimovich, A., Elidan, G., Margalit, H., and Friedman, N. Towards an integrated protein-protein interaction network. In *Proceedings of ACM RECOMB* (2005), pp. 14–30.

[10] Karaoz, U., Murali, T. M., Letovsky, S., Zheng, Y., Ding, C., Cantor, C. R., and Kasif, S. Whole-genome annotation by using evidence integration in functional-linkage networks. *Proc Natl Acad Sci U S A 101*, 9 (2004), 2888–2893.

[11] Letovsky, S., and Kasif, S. Predicting protein function from protein/protein interaction data: a probabilistic approach. *Bioinformatics 19*, 1 (2003), i197–i204.

[12] Li, S., Armstrong, C., and *et al.* A map of the interactome network of the metazoan *C. elegans*. *Science 303* (2004), 540–543.

[13] Nabieva, E., Jim, K., Agarwal, A., Chazelle, B., and Singh, M. Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. In *Proceedings of ISMB* (2005), pp. 302–310.

[14] Rain, J., Selig, L., and *et al.* The protein-protein interaction map of *Helicobacter pylori*. *Nature 409* (2001), 211–215.

[15] Rual, J.-F., Venkatesan, K., and *et al.* Towards a proteome-scale map of the human protein-protein interaction network. *Nature 437* (2005), 1173–1178.

[16] Salwinski, L., Miller, C. S., Smith, A. J., Pettit, F. K., Bowie, J. U., and Eisenberg, D. The database of interacting proteins: 2004 update. *Nucleic Acids Research 32* (2004), D449.

[17] Schwikowski, B., Uetz, P., and Fields, S. A network of protein-protein interactions in yeast. *Nature Biotechnology 18* (2000), 1257 – 1261.

[18] Srinivasan, B. S., Novak, A. F., Flannick, J. A., Batzoglou, S., and McAdams, H. H. Integrated protein interaction networks for 11 microbes. In *Proceedings of ACM RECOMB* (2006), pp. 1–14.

[19] Uetz, P., Giot, L., and *et al.* A comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae*. *Nature 403*, 6770 (2000), 623–627.

[20] Vazirani, V. V. *Approximation algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.

[21] Vazquez, A., Flammini, A., Maritan, A., and Vespignani, A. Global protein function prediction in protein-protein interaction networks. *Nature Biotechnology 21* (2003), 697.