BIOINFORMATICS

Vol. 21 Suppl. 1 2005, pages i302–i310
doi:10.1093/bioinformatics/bti1054

Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps

Elena Nabieva[1,2], Kam Jim[2], Amit Agarwal[1], Bernard Chazelle[1] and Mona Singh[1,2,]*

[1]Computer Science Department and [2]Lewis-Sigler Institute for Integrative Genomics, Princeton University, Princeton, NJ 08544, USA

Received on January 15, 2005; accepted on March 27, 2005

ABSTRACT

Motivation: Determining protein function is one of the most important problems in the post-genomic era. For the typical proteome, there are no functional annotations for one-third or more of its proteins. Recent high-throughput experiments have determined proteome-scale protein physical interaction maps for several organisms. These physical interactions are complemented by an abundance of data about other types of functional relationships between proteins, including genetic interactions, knowledge about co-expression and shared evolutionary history. Taken together, these pairwise linkages can be used to build whole-proteome protein interaction maps.

Results: We develop a network-flow based algorithm, FunctionalFlow, that exploits the underlying structure of protein interaction maps in order to predict protein function. In cross-validation testing on the yeast proteome, we show that FunctionalFlow has improved performance over previous methods in predicting the function of proteins with few (or no) annotated protein neighbors. By comparing several methods that use protein interaction maps to predict protein function, we demonstrate that FunctionalFlow performs well because it takes advantage of both network topology and some measure of locality. Finally, we show that performance can be improved substantially as we consider multiple data sources and use them to create weighted interaction networks.

Availability: http://compbio.cs.princeton.edu/function

Contact: msingh@princeton.edu

INTRODUCTION

A major challenge in the post-genomic era is to determine protein function at the proteomic scale. Even the best-studied model organisms contain a large number of proteins whose functions are currently unknown. For example, about one-third of the proteins in the baker's yeast *Saccharomyces cerevisiae* remain uncharacterized. Traditionally, computational methods to assign protein function have relied largely on sequence homology. The recent emergence of high-throughput experimental datasets have led to a number of alternative, non-homology based methods for functional annotation. These methods have generally exploited the concept of guilt by association, where proteins are functionally linked through either experimental or computational means.

Large-scale experiments have linked proteins that physically interact (Ito *et al*., 2001; Uetz *et al*., 2000; Gavin *et al*., 2002; Ho *et al*., 2002; Rain *et al*., 2001; Giot *et al*., 2003; Li *et al*., 2004), that are synthetic lethals (Tong *et al*., 2001, 2004) and that are coexpressed (Edgar *et al*., 2002) or coregulated (Lee *et al*., 2002; Harbison *et al*., 2004). In addition, computational techniques linking pairs of proteins include phylogenetic profiles (Gaasterland and Ragan, 1998; Pellegrini *et al*., 1999), gene clusters (Overbeek *et al*., 1999), conserved gene neighbors (Dandekar *et al*., 1998) and gene fusion analysis (Enright *et al*., 1999; Marcotte *et al*., 1999a). Perhaps not surprisingly, integrating the information from several sources provides the best method for linking proteins functionally (Marcotte *et al*., 1999b; von Mering *et al*., 2003a; Troyanskaya *et al*., 2003; Jansen *et al*., 2003; Lee *et al*., 2004).

Taken together, these functional linkages form large protein interaction networks, with nodes corresponding to proteins and edges between any two proteins that are functionally linked with each other (Fig. 1). It has been postulated that analysis of these protein interaction maps should provide hints to the higher-level organization of the cell, and help uncover protein functions and pathways (reviews, Alm and Arkin, 2003; Ideker, 2004). We focus here on the problem of predicting protein function by analyzing proteins as components within protein interaction networks.

Several groups have attempted to partition interaction networks into functional modules that correspond to sets of proteins that are part of the same cellular function or take part in the same protein complex. These functional modules, or clusters, are useful for annotating uncharacterized proteins, as the most common functional annotation within a cluster can be transferred to uncharacterized proteins. Proteins in experimentally and computationally determined interaction graphs have been grouped together based on shared interactions (Brun *et al*., 2003; Schlitt *et al*., 2003; Strong *et al*.,

*To whom correspondence should be addressed.

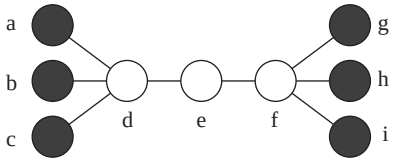

Fig. 1. A protein interaction graph. Nodes represent proteins and edges represent interactions between proteins. For example, protein d interacts with proteins a, b, c and e. Proteins a, b, c, g, h and i (shown in black) are known to take part in the same biological process, and proteins d, e and f are unannotated.

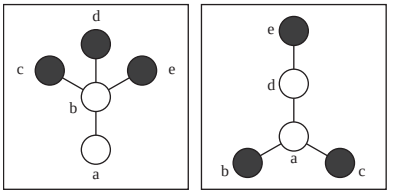

Fig. 2. Two protein interaction graphs that are treated identically by Neighborhood with radius 2 when annotating protein a. Dark colored nodes correspond to proteins that are known to take part in the same process.

2003; von Mering *et al*., 2003b; Lee *et al*., 2004), the similarity between shortest path vectors to all other proteins in the network (Rives and Galitski, 2003) and shared membership within highly connected components or cliques (Spirin and Mirny, 2003).

The research described here is more closely related to the recent attempts to classify proteins according to the functional annotations of their network neighbors; these methods do not explicitly cluster proteins. Schwikowski *et al*. (2000) use physical interaction data for baker's yeast, and predict the biological process for each protein by considering its neighboring interactions and taking the three most frequent annotations. Although such a simple majority vote approach, which we refer to as Majority, has clear predictive value, it takes only limited advantage of the underlying graph structure of the network. For example, in the interaction network given in Figure 1, Majority would assign functions to proteins d and f, but not to protein e, even though our intuition might indicate that protein e has the same function as proteins d and f; there are several examples in the yeast proteome similar to this one (Schwikowski *et al*., 2000). Naturally, one wishes to generalize this principle to consider functional linkages beyond the immediate neighbors in the interaction graph, both to provide a systematic framework for analyzing the entirety of physical interaction data for a given proteome and to make predictions for proteins with no annotated interaction partners.

Hishigaki *et al*. (2001) extend Majority by predicting a protein's function by looking at all proteins within a particular radius and finding over-represented functional annotations. However, this approach, which we refer to as Neighborhood, does not consider any aspect of network topology within the local neighborhood. For example, Figure 2 shows two interaction networks that are treated equivalently when considering a radius of 2 and annotating protein a; however, in the first case, there is a single link that connects protein a to the annotated proteins, and in the second case, there are several independent paths between a and the annotated proteins, and moreover, two of these proteins are directly adjacent to a.

Two recent papers (Vazquez *et al*., 2003; Karaoz *et al*., 2004) exploit the global topological structure of the interaction network by annotating proteins so as to minimize the number of times different annotations are associated with neighboring proteins. Karaoz *et al*. (2004) additionally consider the case where edges in physical interaction networks are weighted using gene-expression data. We refer to this overall approach as GenMultiCut, as it is a generalization of the well-studied multiway k-cut problem in computer science. While GenMultiCut takes into account more global properties of interaction maps, it does not reward local proximity in the graph. For example, if only two proteins have annotations in a particular network, all other proteins will be labeled by one of these annotations, regardless of the size of the network.

To overcome the weaknesses of previous methods, we introduce an algorithm, FunctionalFlow, for annotating protein function in interaction networks. FunctionalFlow uses the idea of network flow, which is dual to the notion of graph cut (Cormen *et al*., 1990). Each protein of known functional annotation is treated as a 'source' of 'functional flow' which is then propagated to unannotated nodes, using the edges in the interaction graph as a conduit. This propagation is governed by simple local rules. By considering a formulation based on flow, we can incorporate a distance effect. That is, the effect of each annotated protein on any other protein decreases with increasing distance between them. In addition, network connectivity is exploited, as each edge has a 'capacity' and multiple paths between two proteins result possibly in more flow between them. After simulating the spread of this functional flow for a fixed number of time steps (so that flow from a source is restricted to a local neighborhood around it) we obtain the 'functional score' for each protein. This score corresponds to the amount of flow for that function the protein has received over the course of simulation. In contrast to Majority, FunctionalFlow considers functional annotations from proteins that are not immediate neighbors, and thus can annotate proteins that have no neighbors with known annotations. In contrast to Neighborhood, FunctionalFlow considers the underlying topology of the graph, and the multiple edge-disjoint interaction paths between two proteins give additional evidence for common function. Finally, in contrast to GenMultiCut, FunctionalFlow takes into account the network locality.

The locality effect in FunctionalFlow is similar in some ways to the locally constrained diffusion kernel developed by Tsuda and Noble (2004). However, the flow in the

FunctionalFlow algorithm is limited by capacities on edges, and in the context of our method, this prevents proteins that have the same annotation but have largely overlapping paths to protein a from exerting too much influence on a. Moreover, Tsuda and Noble (2004) use the diffusion kernel with support vector machines, whereas FunctionalFlow is not a learning method and does not require any training data to be used.

We compare the performance of FunctionalFlow with Majority, Neighborhood and GenMultiCut. In the process, we reformulate the computational problem given by the objective function of Vazquez *et al*. (2003) and Karaoz *et al*. (2004) as an integer linear program (ILP), and as opposed to the previous two studies, we find optimal (not heuristic) solutions to the problem using ILP. Since we find optimal solutions, we directly test the utility of the GenMultiCut objective function. In addition, we show how to obtain multiple optimal solutions using ILP, and show that this is one way to incorporate the idea of distance implicitly within the GenMultiCut framework. In cross-validation testing on the yeast physical interaction network, we show that FunctionalFlow outperforms Neighborhood and GenMultiCut, and has better performance than Majority in predicting the function of proteins with few (or no) annotated protein neighbors. We estimate that in the yeast proteome, there are currently ∼1200 such unannotated proteins where FunctionalFlow would make improved predictions over Majority. This number is 2400 for fruit fly, and the fraction of such proteins should be much higher for less characterized proteomes. Finally, we propose a simple weighting scheme that captures the variation in reliability of the experimental data that form the basis of the interaction network, and show that this scheme results in improved performance for all methods.

Overall, we demonstrate that network analysis algorithms, such as FunctionalFlow, provide an effective new line of attack in determining protein function. Moreover, we show empirically that network analysis algorithms for function prediction obtain the best performance when incorporating overall network topology, network distance and edges weighted by a reliability parameter estimated from multiple data sources. The FunctionalFlow method we introduce incorporates these features and outperforms previously published methods. Although all of our cross-validation testing has been on baker's yeast, FunctionalFlow is likely to be especially useful in characterizing less-studied proteomes.

MATERIALS AND METHODS

Physical interaction network

We construct the protein–protein physical interaction network using the protein interaction dataset compiled by GRID (Breitkreutz *et al*., 2003). The resulting network is a simple undirected graph $G = (V, E)$, where there is a vertex or node $v \in V$ for each protein, and an edge between nodes u and v if the corresponding proteins are known to interact physically (as determined by one or more experiments). Initially, we consider a graph with unit-weighted edges, and then consider weighting the edges by our 'confidence' in the edge (see below). The weight of the edge between u and v is denoted by $w_{u,v}$. For all reported results, we consider only the proteins making up the largest connected component of the physical interaction map (4495 proteins and 12 531 physical interaction links).

Functional annotations

Several controlled vocabulary systems exist for describing biological function, including Munich Information Center for Protein Sequences (MIPS) (Mewes *et al*., 2002) and the Gene Ontology (GO) project (Ashburner *et al*., 2000). We use the MIPS functional hierarchy, and consider the 72 MIPS biological processes that comprise the second level of hierarchy. Of the 4495 proteins in the largest connected component of the yeast physical interaction map, 2946 have MIPS biological process annotations. We also experimented with GO annotations; the overall conclusions made in this paper are not affected.

Weighting functional linkages

It is well known that the reliability of different data sources vary, even if they are based on the same underlying technology (von Mering *et al*., 2002; Deng *et al*., 2003; Sprinzak *et al*., 2003). In the context of network-based algorithms, it is possible to weight edges so as to model the reliability of each interaction. For physical interactions, this reliability is in turn based on the experimental sources that contribute to our knowledge about the existence of the interaction. To determine these values, we separate all experimental sources of physical interaction data into several groups, placing each high-throughput dataset into a separate group (five groups corresponding to each of Ito *et al*., 2001, 2000; Fromont-Racine *et al*., 1997; Uetz *et al*., 2000; Gavin *et al*., 2002; Ho *et al*., 2002), and allocating one group for the family of all specific experiments. For each group of experiments, we compute what fraction of its interactions connect proteins with a known shared function. We assume that the reliabilities of different sources are independent, and thus conclude by estimating the reliability of an interaction to be the noisy-or of the unreliability of the underlying data sources. That is, if r_i is the reliability of experimental group i, we compute the reliability of the edge by $1 - \Pi_i(1 - r_i)$, where the product is taken over all experiments i where this interaction is found. This treats each r_i as a probability and assumes independence; this approach is very similar to the one taken by von Mering *et al*. (2003a).

We also consider augmenting the interaction network by considering genetic interactions from GRID (Breitkreutz *et al*., 2003). Almost all of these interactions are synthetic lethals, and the weighting scheme can be immediately extended to this network by treating the new types of interactions

as an additional experimental source. Thus, our weighting scheme gives us a way of integrating data of different types in addition to integrating different sources of data of one type.

Cross-validation testing and evaluation

We test the performance using n-fold cross-validation, i.e. the yeast proteome is divided into n groups, and each group, in turn, is separated from the original dataset and used for testing. The goal of each method is to predict the annotations of the proteins in the test set using the functional annotations of the remaining proteins. We performed experiments with 2-, 3-, 5- and 10-fold cross-validation. All our cross-validation testing gives qualitatively similar results. We report our findings using a 2-fold cross-validation, as baker's yeast is the most extensively studied organism, and 2-fold cross-validation better represents what one may expect to see in other organisms.

We evaluate the performance of the algorithms by considering, for each protein in the test set, whether the top scoring prediction above some threshold is a known functional annotation (true positive, TP) or not (false positive, FP). In the case of multiple predictions, the TP versus FP status is tricky. For example, we may choose to count a prediction for a protein as a TP if at least one of the predictions made for it is correct, and as a FP otherwise. However, a method that predicts every protein to participate in every function would only have TPs in this framework. Alternatively, we could count a protein as a TP if every prediction made for it is correct. This, however, would count as FPs those proteins that get many correct predictions and only one incorrect one. We settle for a compromise approach, in which we count a protein's prediction as a TP if more than half of the predictions made for it are correct and as a FP otherwise. All results will be reported using this interpretation of TP and FP, and we use a variant of receiver operating characteristic (ROC) curves, where we plot the number of TPs as a function of the number of FPs as we vary the scoring threshold.

ALGORITHMS

Majority

We consider all neighboring proteins and sum up the number of times each annotation occurs for each protein as described in Schwikowski *et al*. (2000). In the case of weighted interaction graphs, we simply extend the method by taking a weighted sum instead. For each protein, the score of a particular function is the corresponding sum.

Neighborhood

For each protein, we consider all other proteins within a radius r as described in Hishigaki *et al*. (2001), and then for each function, we use a χ^2-test to determine if it is over-represented. For each protein, the score of a particular function is given by the value of the χ^2-test. Neighborhoods of radius 1, 2 and 3 are considered. This method does not extend naturally to the case of weighted interaction graphs.

GenMultiCut

Two groups of researchers have suggested that functional annotations on interaction networks should be made in order to minimize the number of times different annotations are associated with neighboring proteins (Vazquez *et al*., 2003; Karaoz *et al*., 2004). Vazquez *et al*. (2003) use simulated annealing in an attempt to minimize this objective function and aggregate results from multiple runs, whereas Karaoz *et al*. (2004) use a deterministic approximation, and consider the case where edges are weighted using gene expression information. As mentioned earlier, the formulation in these two studies is similar to the minimum multiway k-cut problem. In multiway k-cut, the task is to partition a graph in such a way that each of k terminal nodes belongs to a different subset of the partition and so that the (weighted) number of edges that are 'cut' in the process is minimized. In the more general version of the multiway k-cut problem considered here, the goal is to assign a unique function to all the unannotated nodes so as to minimize the sum of the costs of the edges joining nodes with no function in common.

Our implementation of GenMultiCut Although minimum multiway k-cut is NP-hard (Dahlhaus *et al*., 1994), we have found that the particular instances of minimum multiway cut arising here can, in practice, be solved exactly when stated as an ILP. We introduce a node variable $x_{u,a}$ for each protein u and function a. This variable will be set to 1 if protein u is predicted to have function a. If a protein u has known functional annotations, variable $x_{u,a}$ is fixed as 1 for its known annotations a and as 0 for all other annotations. We also introduce an edge variable $x_{u,v,a}$ for each function a and each pair of adjacent proteins u and v. This variable is set to 1 if both proteins u and v are annotated with function a. Minimizing the weighted number of neighboring proteins with different annotations is the same as maximizing the number with the same annotation, and so we have the following ILP:

$$\text{maximize} \sum_{(u,v)\in E, a\in \text{FUNC}} x_{u,v,a} w_{u,v}$$

subject to

$$\begin{aligned}
&\sum_a x_{u,a} = 1 && \text{if annot}(u) = \emptyset \\
&x_{u,a} = 1 && \text{if } a \in \text{annot}(u) \\
&x_{u,a} = 0 && \text{if } a \notin \text{annot}(u), \text{annot}(u) \neq \emptyset \\
&x_{u,v,a} \leq x_{u,a} && \text{for } (u,v) \in E \text{ and } a \in \text{FUNC} \\
&x_{u,v,a} \leq x_{v,a} && \text{for } (u,v) \in E \text{ and } a \in \text{FUNC} \\
&x_{u,v,a}, x_{u,a} \in \{0,1\} && \text{for all } u, v \text{ and } a.
\end{aligned}$$

Here, annot(u) is the set of known annotations for protein u, and FUNC $= \cup_u$annot(u) is the set of all functional annotations. The first constraint specifies that exactly one

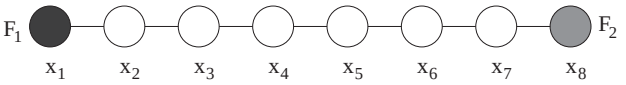

Fig. 3. Proteins x_1 and x_8 are annotated with functions F_1 and F_2, respectively. There are seven ways to annotate proteins so that there is only one edge that connects proteins with different annotations. However, proteins farther away from protein x_1 are less likely to have function F_1 than those closer to x_1. *GenMultiCut* does not take into account such distance effects.

functional annotation is made for any protein. The second and third constraints ensure that if protein u is annotated with function a, $x_{u,a}$ is set as a constant to 1, and if protein u is annotated but not with function a, $x_{u,a}$ is set as a constant to 0. The third and fourth constraints ensure that a particular function is picked for an edge only if it is also chosen for the corresponding proteins.

Considering multiple GenMultiCut optimal solutions An important consideration in this framework is the existence of multiple optimal solutions. For example, the network in Figure 3 has seven minimum cuts of value 1, and while the GenMultiCut criterion does not favor any one cut over the other, if we find all optimal cuts for this graph, we observe that x_2 is in fact annotated with F_1 more often than with F_2 in the assignments made by these cuts. Thus, a sense of distance to annotated nodes is in fact present in the set of all optimal solutions.

The simulated annealing method of Vazquez *et al*. (2003) implicitly utilizes this information about multiple solutions. Vazquez *et al*. (2003) ran simulated annealing 100 times, and predicted for each protein the function that is assigned to it most often. If each run does indeed converge to an optimal solution, considering multiple runs amounts to sampling from the space of optimal solutions.

We deliberately attempt to sample from the space of optimal solutions. We explore two approaches for ensuring that multiple solutions are obtained by the solver. In the solution-exclusion approach, we add constraints to the ILP which require that each consecutive solution is different from any previous solution in the value it assigns to at least 5% of the node variables $x_{u,a}$. For the weighted yeast physical interaction graph, the first 50 solutions obtained with this restriction are all optimal. Note that in this approach, each successive solution takes longer to find. In the random weight perturbation approach, we introduce uniform self-weights $w_{u,a}$ for each protein u and function a. These self-weights are then perturbed by adding a very small offset to each, drawn at random from the uniform distribution on $(-0.00001, 0.00001)$. We now modify the objective function in the ILP given above to maximize

$$\sum_{(u,v)\in E, a\in \mathrm{FUNC}} x_{u,v,a} w_{u,v} + \sum_{u\in V, a\in \mathrm{FUNC}} x_{u,a} w_{u,a}.$$

The perturbation in weights is too small to change the solution to the underlying problem, but it does cause the solver to choose a different optimal solution each time. Both methods perform very similarly in the accuracy of predictions made. For the reported results, we use the latter method for obtaining multiple solutions.

We let the score for assigning a function to a protein be the number of times this function is assigned to the protein among the obtained solutions. We ran the ILP 50 times, and thus, there are 51 possible scores (0–50) for any function for any protein. One solution to the ILP problem on the yeast interaction network with annotations for 50% of the proteins cleared can be obtained by AMPL (Fourer *et al*., 2002) and CPLEX (http://www.ilog.com/products/cplex/; ILOG CPLEX, 2000) in ∼5 min when running on a public UNIX machine.

FunctionalFlow

The functional flow algorithm generalizes the principle of 'guilt by association' to groups of proteins that may or may not interact with each other physically. We achieve this by treating each protein of known functional annotation as a 'source' of 'functional flow' for that function. After simulating the spread over time of this functional flow through the neighborhoods surrounding the sources, we obtain the 'functional score' for each protein in the neighborhood; this score corresponds to the amount of 'flow' that the protein has received for that function, over the course of the simulation. The functional flow-based model allows us to incorporate a distance effect, i.e. the effect of each annotated protein on any other protein depends on the distance separating these two proteins. Running this process for each biological function in turn, we obtain, for each protein, the score for each function (the score may be 0 if the flow for a function did not reach that protein during the simulation). Thereupon, for any protein, we take the functions for which the highest score was obtained as its predicted functions.

More specifically, for each function in turn, we simulate the spread of functional flow by an iterative algorithm using discrete time steps. We associate with each node (protein) a 'reservoir' which represents the amount of flow that the node can pass on to its neighbors at the next iteration, and with each edge, a capacity constraint that dictates the amount of flow that can pass through the edge during one iteration. The capacity of an edge is taken to be its weight. Each iteration of the algorithm updates the reservoirs using simple local rules: a node pushes the flow residing in its reservoir to its neighbors proportionally to the capacities of the respective edges and subject to further constraints that the amount of flow pushed through an edge during an iteration does not exceed the capacity of the edge, and that flow only spreads 'downhill' (i.e. from proteins with more filled reservoirs to nodes with less filled reservoirs). Finally, at each iteration, an 'infinite' amount of flow is pumped into the source protein nodes; thus,

the sources always have enough flow in their reservoir to fill the capacity of their outgoing edges.

The functional score is the amount of flow that has entered a protein's reservoir in the course of all iterations. Since the flow is pumped into the sources at each step, the amount of flow a node receives from each source is greater for nodes that are closer to that source than for nodes that are farther away from it. Thus, a source's immediate neighbor in the graph receives d iterations worth of flow from the source, whereas a node that is two links away from the source receives $d-1$ iterations worth of flow. Similarly, the number of iterations for which the algorithm is run determines the maximum shortest-path distance that can separate a recipient node from a source in order for the flow to propagate from the source to the recipient. In the context of protein interaction, a relatively small number of iterations is sufficient. We choose $d=6$, which is half the diameter of the yeast physical interaction network.

More formally, for each protein u in the interaction network, we define a variable $R_t^a(u)$ that corresponds to the amount in the reservoir for function a that node u has at time t. For each edge (u,v) in the interaction network, we define variables $g_t^a(u,v)$ and $g_t^a(v,u)$ that represent the flow of function a at time t from protein u to protein v, and from protein v to protein u. We will run the algorithm for d time steps or iterations. At time 0, we only have reservoirs of function a at annotated nodes:

$$R_0^a(u) = \begin{cases} \infty, & \text{if } u \text{ is annotated with } a, \\ 0, & \text{otherwise.} \end{cases}$$

At each subsequent time step, we recompute the reservoir of each protein by considering the amount of flow that has entered the node and the amount that has left:

$$R_t^a(u) = R_{t-1}^a(u) + \sum_{v:(u,v)\in E} \left(g_t^a(v,u) - g_t^a(u,v)\right).$$

Initially, at time 0, there is no flow on the edges, and $g_0^a(u,v)=0$. At each subsequent time step, we have the flow proceeding downhill and satisfying the capacity constraints:

$$g_t^a(u,v) = \begin{cases} 0, & \text{if } R_{t-1}^a(u) < R_{t-1}^a(v) \\ \min\left(w_{u,v}, \frac{w_{u,v}}{\sum_{(u,y)\in E} w_{u,y}}\right), & \text{otherwise.} \end{cases}$$

Finally, the functional score for node u and function a over d iterations is calculated as the total amount of flow that has entered the node:

$$f_a(u) = \sum_{t=1}^{d} \sum_{v:(u,v)\in E} g_t^a(v,u).$$

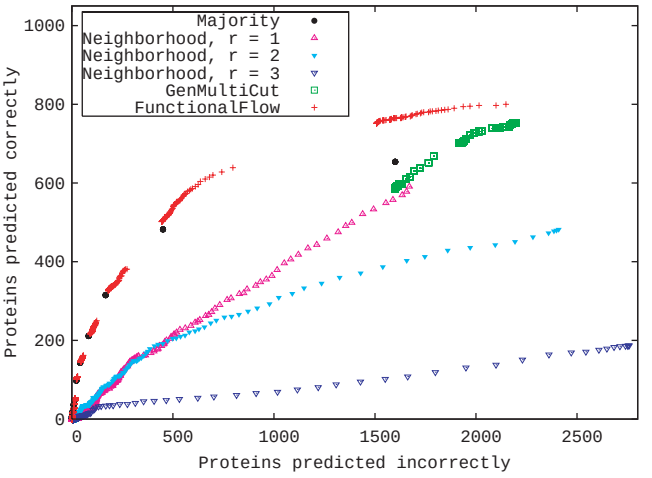

Fig. 4. ROC analysis of Majority, Neighborhood, GenMultiCut and FunctionalFlow on the yeast unweighted physical interaction map.

RESULTS AND DISCUSSION

Comparison of four basic methods on the unweighted physical interaction map

We compare the performance of Majority, Neighborhood, GenMultiCut and FunctionalFlow on the unweighted yeast physical interaction map, using a 2-fold cross-validation. Figure 4 plots as a function of FP the number of TPs each method predicts (i.e. these graphs are obtained by varying the scoring threshold for each of the methods). The FunctionalFlow algorithm identifies more TPs over the entire range of FPs than either GenMultiCut or Neighborhood using radius 1, 2 or 3. FunctionalFlow performs better than Majority when proteins are not directly interacting with at least three proteins of the same function; this is evident from Figure 4 since the score for Majority counts up the most frequent neighboring annotation (e.g. the rightmost point for Majority corresponds to proteins whose highest functional scores are one). Thus, FunctionalFlow is the method of choice when considering proteins that do not interact with many annotated proteins. Even in well-characterized proteomes, such as baker's yeast, there are ~1200 proteins that have fewer than three annotated neighbors.

The Neighborhood algorithm performs similarly with either radius 1 or 2 in the high-confidence region (i.e. corresponding to a low FP rate, given in left-most portion of the ROC curve). However, radius 1 (i.e. considering just direct interactions) has better overall performance than radius 2 or 3, demonstrating that Neighborhood's strategy of ignoring topology is not optimal. Moreover, comparing Majority with Neighborhood using radius 1 demonstrates that the χ^2-test is not as effective in scoring as just summing up the number of times a particular annotation occurs in the neighboring proteins.

Since the score for GenMultiCut comes from multiple solutions to the underlying optimization problem, each point in

Figure 4 for GenMultiCut corresponds to the proteins that are annotated with a particular function the same number of times. For example, the leftmost point for GenMultiCut corresponds to proteins where the top scoring functional prediction is found in each of the 50 solutions found. If we were to find just one optimal GenMultiCut solution, its performance in terms of TPs and FPs is comparable to the rightmost point for GenMultiCut (data not shown).[1] Thus, multiple solutions for GenMultiCut are necessary to identify its most confident predictions, and as pointed out earlier, these multiple solutions capture some notion of locality in the graph.

Vazquez *et al*. (2003) report in their paper improved performance for GenMultiCut over Majority for proteins with degree >1. Their measure of success is the fraction of times the top prediction for each protein is correct. Although they do not specify how they deal with multiple top predictions, we note that this measure corresponds to computing TPs and FPs for the rightmost points in Figure 4 for each of the methods. Assuming that the top predictions for each protein are treated separately, and that failure to make a prediction for a protein corresponds to an incorrect prediction, the top predictions for proteins with degree >1 are correct 0.267 of the time for Majority. These values are 0.242 for Neighborhood with radius 1, 0.188 for Neighborhood with radius 2, 0.297 for GenMultiCut and 0.311 for FunctionalFlow. Although we believe ROC curve analysis gives a more complete picture of performance, FunctionalFlow performs better than the other methods using this measure. Moreover, we tested the performance of all methods clearing a smaller fraction of the annotated proteins. In a 10-fold cross-validation (i.e. where only 10% of the yeast annotations are cleared), GenMultiCut has a slight advantage (25 proteins out of ∼2500) over FunctionalFlow in the very low-confidence region; all other observations are qualitatively the same as for 2-fold cross-validation.

Reliability and data integration

To evaluate our approach for modeling physical interaction reliability as edge weights, we test the performance of FunctionalFlow using three ways of assigning physical interaction weights. First, we assign each edge a unit weight; this corresponds to the unweighted physical interaction map used above. Second, we assign each experimental source a reliability score of 0.5; this rewards interactions that are found by more than one experiment. Finally, we assign each experimental source the predictive value (estimated in cross-validation) as described in the Materials and Methods section; here, edges obtained from multiple, more reliable experiments are given higher weights. Figure 5 shows that rewarding multiple experimental evidence is beneficial, but that the main advantage comes from taking into account the actual reliability values for the different experiments.

[1]It is not precisely the rightmost point in Figure 4 since this point aggregates solutions from multiple runs.

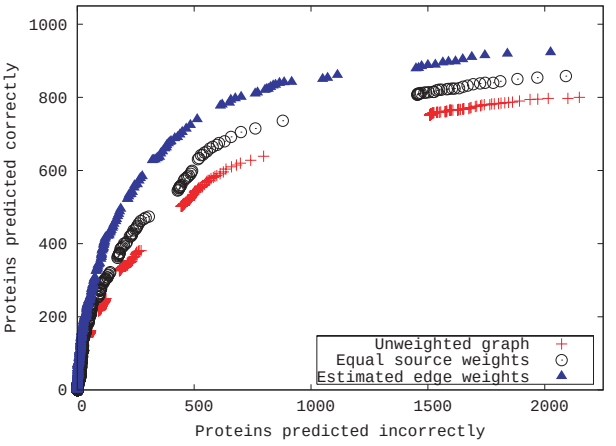

Fig. 5. The FunctionalFlow algorithm on (1) the unweighted physical interaction map, (2) the physical interaction map with edges weighted using equal reliabilities for each experiment and (3) the physical interaction map with edges weighted by reliabilities estimated individually for each experiment.

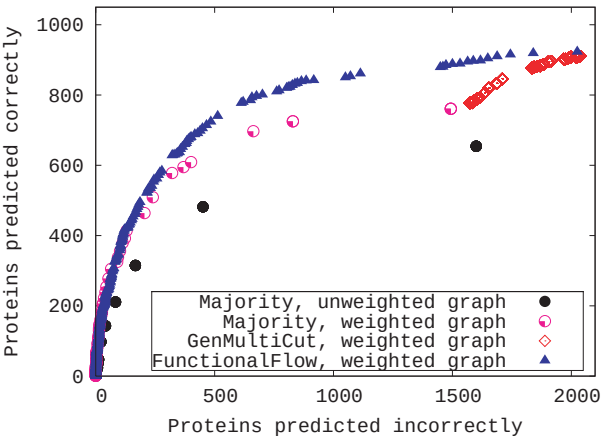

Fig. 6. Performance of Majority, GenMultiCut and FunctionalFlow on the physical interaction map where experimental reliabilities are incorporated. The performance of Majority on the unweighted graph is also given as a reference.

Figure 6 shows how Majority, GenMultiCut and FunctionalFlow perform on the yeast physical interaction map, where edges are weighted by individual experimental reliability. The baseline performance of Majority on the unweighted physical interaction graph is also shown. There is substantial improvement in predictions using all three methods when incorporating edges weighted by reliability.

We further explored whether the network analysis algorithms would perform well when other types of experimental information are added. As a proof of principle, we explore the effect of adding genetic linkages to the graph. Reliabilities for genetic interactions are estimated as described earlier, and incorporated into the edge weights. Figure 7 shows the performance of FunctionalFlow on the weighted physical

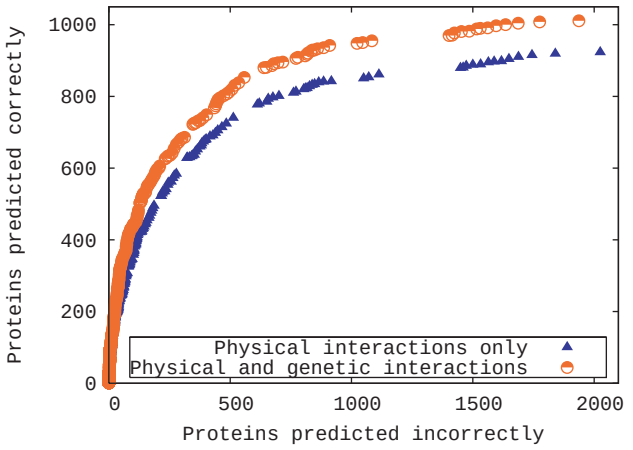

Fig. 7. Comparison of functional predictions of FunctionalFlow when considering (1) the physical interaction map weighted by experimental source reliability and (2) the integrated physical and genetic interaction map.

interaction network and the weighted physical and genetic interaction network. As is evident, adding genetic interaction data significantly improves prediction quality. Majority and GenMultiCut show similar improvements (data not shown).

CONCLUSIONS

We have shown that our network analysis algorithm FunctionalFlow provides an effective means for predicting protein function from protein interaction maps. Our algorithm utilizes indirect network interactions, network topology, network distances and edges weighted by reliability estimated from multiple data sources. However, we have also shown that the simplest methods, such as Majority, perform well if there are enough direct neighbors with known function. In the present work, simple independence assumptions are made for estimating the reliability of interactions. Although these work reasonably well, it may be even more beneficial to use a more sophisticated approach for weight assignment and perform more complete data integration. Finally, although we have applied our method to baker's yeast, FunctionalFlow is likely to be especially useful when analyzing less characterized proteomes.

ACKNOWLEDGMENTS

M.S. thanks the NSF for PECASE grant MCB-0093399, DARPA for grant MDA972-00-1-0031 and NIH for grant PO1-CA-041086. B.C. thanks the NSF for grant CCR-0306283, DARPA for ARO grant DAAH04-96-1-0181 and the NEC Research Institute. The authors thank the members of the Singh group, especially Carl Kingsford, for many helpful discussions.

REFERENCES

Alm,E. and Arkin,A.P. (2003) Biological networks. *Curr. Opin. Struct. Biol.*, **13**, 193–202.

Ashburner,M., Ball,C.A., Blake,J.A., Botstein,D., Butler,H., Cherry,J.M., Davis,A.P., Dolinski,K., Dwight,S.S., Eppig,J.T. *et al.* (2000) Gene Ontology: tool for the unification of biology. The Gene Ontology Consortium. *Nat. Genet.*, **25**, 25–29.

Breitkreutz,B.J., Stark,C. and Tyers,M. (2003) The GRID: the general repository for interaction datasets. *Genome Biol.*, **4**, R23.

Brun,C., Chevenet,F., Martin,D., Wojcik,J., Guénoche,A. and Jacq,B. (2003) Functional classification of proteins for the prediction of cellular function from a protein–protein interaction network. *Genome Biol.*, **5**, R6.

Cormen,T.H., Leiserson,C.E. and Rivest,R.L. (1990) *Introduction to Algorithms*. MIT Press, Cambridge, MA.

Dahlhaus,E., Johnson,D.S., Papadimitriou,C.H., Seymour,P.D. and Yannakakis,M. (1994) The complexity of multiway cuts. *SIAM J. Comput.*, **23**, 864–894.

Dandekar,T., Snel,B., Huynen,M. and Bork,P. (1998) Conservation of gene order: a fingerprint of proteins that physically interact. *Trends Biochem. Sci.*, **23**, 324–328.

Deng,M., Sun,F. and Chen,T. (2003) Assessment of the reliability of protein–protein interactions and protein function prediction. In *Proceeding of the Pacific Symposium on Biocomputing*, pp. 140–151.

Edgar,R., Domrachev,M. and Lash,A.E. (2002) Gene Expression Omnibus: NCBI gene expression and hybridization array data repository. *Nucleic Acids Res.*, **30**, 207–210.

Enright,A.J., Iliopoulos,I., Kyrpides,N.C. and Ouzounis,C.A. (1999) Protein interaction maps for complete genomes based on gene fusion events. *Nature*, **402**, 86–90.

Fourer,R., Gay,D.M. and Kernighan,B.W. (2002) *AMPL: A Modeling Language for Mathematical Programming*. Brooks/Cole Publishing Company, Pacific Grove, CA.

Fromont-Racine,M., Rain,J.C. and Legrain,P. (1997) Toward a functional analysis of the yeast genome through exhaustive two-hybrid screens. *Nat. Genet.*, **16**, 277–282.

Gaasterland,T. and Ragan,M.A. (1998) Constructing multigenome views of whole microbial genomes. *Microb. Comp. Genom.*, **3**, 177–192.

Gavin,A.C., Bosche,M., Krause,R., Grandi,P., Marzioch,M., Bauer,A., Schultz,J., Rick,J.M., Michon,A.M., Cruciat,C.M. *et al.* (2002) Functional organization of the yeast proteome by systematic analysis of protein complexes. *Nature*, **415**, 141–147.

Giot,L., Bader,J.S., Brouwer,C., Chaudhuri,A., Kuang,B., Li,Y., Hao,Y.L., Ooi,C.E., Godwin,B., Vitols,E. *et al.* (2003) A protein interaction map of *Drosophila melanogaster*. *Science*, **302**, 1727–1736.

Harbison,C.T., Gordon,D.B., Lee,T.I., Rinaldi,N.J., Macisaac,K.D., Danford,T.W., Hannett,N.M., Tagne,J.-B., Reynolds,D.B., Yoo,J. *et al.* (2004) Transcriptional regulatory code of a eukaryotic genome. *Nature*, **431**, 99–104.

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

Ho,Y., Gruhler,A., Heilbut,A., Bader,G.D., Moore,L., Adams,S.L., Millar,A., Taylor,B., Bonnett,K., Boutilier,K. *et al.* (2002)

Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry. *Nature*, **415**, 180–183.

Ideker,T. (2004) A systems approach to discovering signaling and regulatory pathways—or, how to digest large interaction networks into relevant pieces. *Adv. Exp. Med. Biol.*, **547**, 21–30.

ILOG CPLEX (2000) ILOG CPLEX 7.1.

Ito,T., Chiba,T., Ozawa,R., Yoshida,M., Hattori,M. and Sakaki,Y. (2001) A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proc. Natl Acad. Sci. USA*, **98**, 4569–4574.

Ito,T., Tashiro,K., Muta,S., Ozawa,R., Chiba,T., Nishizawa,M., Yamamoto,K., Kuhara,S. and Sakaki,Y. (2000) Toward a protein–protein interaction map of the budding yeast: a comprehensive system to examine two-hybrid interactions in all possible combinations between the yeast proteins. *Proc. Natl Acad. Sci. USA*, **97**, 1143–1147.

Jansen,R., Yu,H., Greenbaum,D., Kluger,Y., Krogan,N.J., Chung,S., Emili,A., Snyder,M., Greenblatt,J.F. and Gerstein,M. (2003) A Bayesian networks approach for predicting protein–protein interactions from genomic data. *Science*, **302**, 449–453.

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

Lee,I., Date,S.V., Adai,A.T. and Marcotte,E.M. (2004) A probabilistic functional network of yeast genes. *Science*, **306**, 1555–1558.

Lee,T.I., Rinaldi,N.J., Robert,F., Odom,D.T., Bar-Joseph,Z., Gerber,G.K., Hannett,N.M., Harbison,C.T., Thompson,C.M., Simon,I. *et al*. (2002) Transcriptional regulatory networks in *Saccharomyces cerevisiae*. *Science*, **298**, 799–804.

Li,S., Armstrong,C.M., Bertin,N., Ge,H., Milstein,S., Boxem,M., Vidalain,P.-O., Han,J.-D.J., Chesnean,A., Hao,T. *et al*. (2004) A map of the interactome network of the metazoan *C. elegans*. *Science*, **303**, 540–543.

Marcotte,E.M., Pellegrini,M., Ng,H.L., Rice,D.W., Yeates,T.O. and Eisenberg,D. (1999a) Detecting protein function and protein–protein interactions from genome sequences. *Science*, **285**, 751–753.

Marcotte,E.M., Pellegrini,M., Thompson,M.J., Yeates,T.O. and Eisenberg,D. (1999b) A combined algorithm for genome-wide prediction of protein function. *Nature*, **402**, 83–86.

Mewes,H.W., Frishman,D., Guldener,U., Mannhaupt,G., Mayer,K., Mokrejs,M., Morgenstern,B., Münsterkötter,M., Rudd,S. and Weil,B. (2002) MIPS: a database for genomes and protein sequences. *Nucleic Acids Res.*, **30**, 31–34.

Overbeek,R., Fonstein,M., D'Souza,M., Pusch,G.D. and Maltsev,N. (1999) The use of gene clusters to infer functional coupling. *Proc. Natl Acad. Sci. USA*, **96**, 2896–2901.

Pellegrini,M., Marcotte,E.M., Thompson,M.J., Eisenberg,D. and Yeates,T.O. (1999) Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. *Proc. Natl Acad. Sci. USA*, **96**, 4285–4288.

Rain,J.C., Selig,L., De Reuse,H., Battaglia,V., Reverdy,C., Simon,S., Lenzen,G., Petel,F., Wojcik,J., Schächter,V. *et al*. (2001) The protein–protein interaction map of *Helicobacter pylori*. *Nature*, **409**, 211–215.

Rives,A.W. and Galitski,T. (2003) Modular organization of cellular networks. *Proc. Natl Acad. Sci. USA*, **100**, 1128–1133.

Schlitt,T., Palin,K., Rung,J., Dietmann,S., Lappe,M., Ukkonen,E. and Brazma,A. (2003) From gene networks to gene function. *Genome Res.*, **13**, 2568–2576.

Schwikowski,B., Uetz,P. and Fields,S. (2000) A network of protein–protein interactions in yeast. *Nat. Biotechnol.*, **18**, 1257–1261.

Spirin,V. and Mirny,L.A. (2003) Protein complexes and functional modules in molecular networks. *Proc. Natl Acad. Sci. USA.*, **100**, 12123–12128.

Sprinzak,E., Sattath,S. and Margalit,H. (2003) How reliable are experimental protein–protein interaction data? *J. Mol. Biol.*, **327**, 919–923.

Strong,M., Graeber,T.G., Beeby,M., Pellegrini,M., Thompson,M.J., Yeates,T.O. and Eisenberg,D. (2003) Visualization and interpretation of protein networks in *Mycobacterium tuberculosis* based on hierarchical clustering of genome-wide functional linkage maps. *Nucleic Acids Res*, **31**, 7099–7109.

Tong,A.H., Evangelista,M., Parsons,A.B., Xu,H., Bader,G.D., Page,N., Robinson,M., Raghibizadeh,S., Hogue,C.W., Bussey,H. *et al*. (2001) Systematic genetic analysis with ordered arrays of yeast deletion mutants. *Science*, **294**, 2364–2368.

Tong,A.H., Lesage,G., Bader,G.D., Ding,H., Xu,H., Xin,X., Young,J., Berriz,G.F., Brost,R.L., Chang,M. *et al*. (2004) Global mapping of the yeast genetic interaction network. *Science*, **303**, 808–813.

Troyanskaya,O.G., Dolinski,K., Owen,A.B., Altman,R.B. and Botstein,D. (2003) A Bayesian framework for combining heterogeneous data sources for gene function prediction (in *Saccharomyces cerevisiae*). *Proc. Natl Acad. Sci. USA*, **100**, 8348–8353.

Tsuda,K. and Noble,W.S. (2004) Learning kernels from biological networks by maximizing entropy. *Bioinformatics*, **20** (Suppl 1), I326–I333.

Uetz,P., Giot,L., Cagney,G., Mansfield,T.A., Judson,R.S., Knight,J.R., Lockshon,D., Narayan,V., Srinivasan,M., Pochart,P. *et al*. (2000) A comprehensive analysis of protein–protein interactions in *Saccharomyces cerevisiae*. *Nature*, **403**, 623–627.

Vazquez,A., Flammini,A., Maritan,A. and Vespignani,A. (2003) Global protein function prediction from protein–protein interaction networks. *Nat. Biotechnol.*, **21**, 697–700.

von Mering,C., Huynen,M., Jaeggi,D., Schmidt,S., Bork,P. and Snel,B. (2003a) STRING: a database of predicted functional associations between proteins. *Nucleic Acids Res.*, **31**, 258–261.

von Mering,C., Krause,R., Snel,B., Cornell,M., Oliver,S.G., Fields,S. and Bork,P. (2002) Comparative assessment of large-scale data sets of protein–protein interactions. *Nature*, **417**, 399–403.

von Mering,C., Zdobnov,E.M., Tsoka,S., Ciccarelli,F.D., Pereira-Leal,J.B., Ouzounis,C.A. and Bork,P. (2003b) Genome evolution reveals biochemical networks and functional modules. *Proc. Natl Acad. Sci. USA*, **100**, 15428–15433.