OPEN ACCESS Freely available online

PLOS ONE

Going the Distance for Protein Function Prediction: A New Distance Metric for Protein Interaction Networks

Mengfei Cao[1], Hao Zhang[1], Jisoo Park[1], Noah M. Daniels[1], Mark E. Crovella[2], Lenore J. Cowen[1]*, Benjamin Hescott[1]*

1 Department of Computer Science, Tufts University, Medford, Massachusetts, United States of America, **2** Department of Computer Science, Boston University, Boston, Massachusetts, United States of America

Abstract

In protein-protein interaction (PPI) networks, functional similarity is often inferred based on the function of directly interacting proteins, or more generally, some notion of interaction network proximity among proteins in a local neighborhood. Prior methods typically measure proximity as the shortest-path distance in the network, but this has only a limited ability to capture fine-grained neighborhood distinctions, because most proteins are close to each other, and there are many ties in proximity. We introduce diffusion state distance (DSD), a new metric based on a graph diffusion property, designed to capture finer-grained distinctions in proximity for transfer of functional annotation in PPI networks. We present a tool that, when input a PPI network, will output the DSD distances between every pair of proteins. We show that replacing the shortest-path metric by DSD improves the performance of classical function prediction methods across the board.

Citation: Cao M, Zhang H, Park J, Daniels NM, Crovella ME, et al. (2013) Going the Distance for Protein Function Prediction: A New Distance Metric for Protein Interaction Networks. PLoS ONE 8(10): e76339. doi:10.1371/journal.pone.0076339

Editor: Stefano Boccaletti, Technical University of Madrid, Italy

Received May 17, 2013; **Accepted** August 23, 2013; **Published** October 23, 2013

Funding: MC, HZ, NMD and LJC were supported in part by National Institutes of Health (NIH) R01 grant GM080330. JP was supported in part by NIH grant R01 HD058880. This material is based upon work supported by the National Science Foundation under grant numbers CNS-0905565, CNS-1018266, CNS-1012910, and CNS-1117039, and supported by the Army Research Office under grant W911NF-11-1-0227 (to MEC). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Competing Interests: The authors have declared that no competing interests exist.

* E-mail: lenore.cowen@tufts.edu (LJC); benjamin.hescott@tufts.edu (BH)

Introduction

One of the best-studied classical problems on biological networks involves using proteins of known function, together with the structure of the network of known protein-protein interactions (PPI) to make predictions of functions of unlabeled protein nodes. This is an important problem because, even in the best-studied model organisms, such as *S. cerevisiae*, these networks contain many proteins whose function is still completely uncharacterized. There are many proposed methods for this problem, including versions of majority voting [1], neighborhood algorithms [2,3], clustering algorithms [4–6], algorithms based on maximum flow [7], or multi-way cut [8,9], and a number of others [10].

Modern approaches also seek to deal with data quality: generally, the known network is missing many valid interactions, and some interactions may be known with higher confidence than others [11,12]. Other modern approaches integrate PPI network data with high-throughput biological interaction data, such as sequence information, genetic interactions, structural information, and expression data [8,13–15]. However, nearly all methods that predict function using PPI network structure depend, entirely or at least partially, on the simple shortest-path distance metric applied to the PPI graph.

We start with the observation that there is an intrinsic drawback to relying on the ordinary shortest-path distance metric in PPI networks. PPI networks are known to be "small world" networks in the sense that they are small-diameter, and most nodes are close to all other nodes (though the exact details of the degree distribution and the resulting extent to which they are "scale free" is a subject of lively debate, see [13,16–19]). Thus any method that infers similarity based on proximity will find that a large fraction of the network is proximate to any typical node. In fact, this issue has already been termed the "ties in proximity" problem in the computational biology literature [4].

Furthermore, the fact that two nodes are adjacent (i.e., have shortest-path distance 1) in a PPI network can signify something very different than the adjacency of two other nodes. For example, as we discuss below, in PPI networks two nodes with many low-degree neighbors in common should be thought of as "more similar" than nodes with few low-degree neighbors in common; and such nodes should also be thought of as "more similar" than two nodes whose common neighbors have high degree. Thus, characterizing node pairs based only on a shortest-path notion of distance fails to capture important knowledge encoded in the structure of the network.

What is needed instead is a finer-grained distance metric, capable of making more subtle distinctions of similarity than ordinary shortest-path distance would in a small-world network. To that end we introduce a new metric called *Diffusion State Distance*, or DSD. We show that DSD is much more effective than shortest-path distance for the problem of transferring functional labels across nodes in PPI networks.

We demonstrate the utility of the DSD metric by modifying a number of the most popular classical algorithms for function prediction to replace the ordinary notion of distance with DSD. For the problem of predicting functional labels in the *S. cerevisiae* network, this change improves *all* the algorithms we consider. We

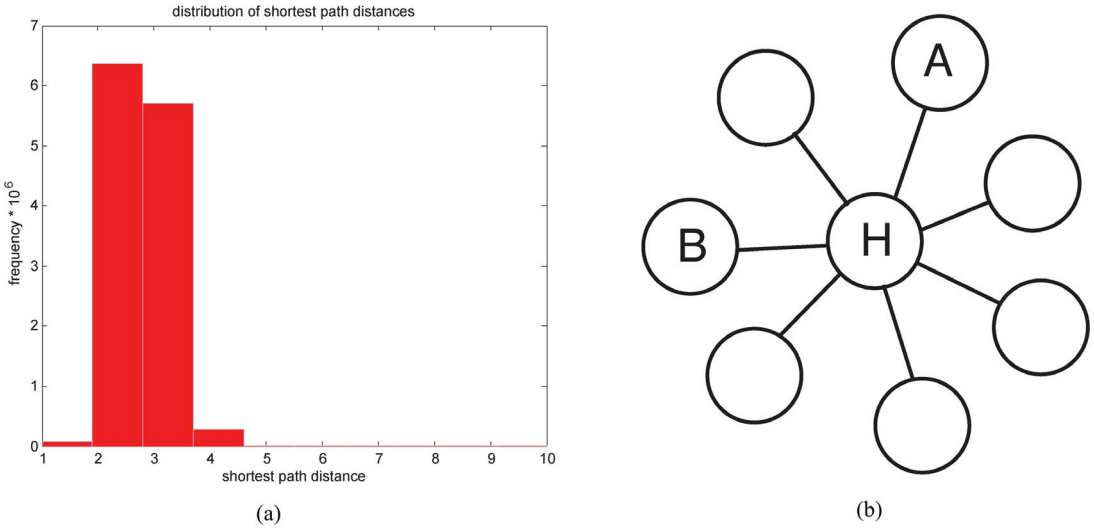

Figure 1. Structure of shortest paths in the yeast PPI network. (a) Distribution of shortest-path distances in the largest connected component of the yeast PPI network; (b) an example subgraph with a hub.
doi:10.1371/journal.pone.0076339.g001

then show that similar improvements hold for the more sparsely annotated *S. pombe* network, implying that our advances should generalize to other biological networks.

Motivation for DSD

To start, we consider all known physical interactions in the *S. cerevisiae* PPI network – specifically, version 3.2.102 of BioGRID [20] on verified ORFs for *S. cerevisiae*, which contains 128,643 annotated physical interactions. After removing redundant edges and selecting the largest connected component, the resulting PPI network has 4990 nodes (where each ORF corresponds to a node) and 74,310 edges (where each edge corresponds to an annotated physical interaction).

Figure 1(a) shows the histogram of shortest-path lengths from this network. The figure shows that almost all pairs of nodes (over 95%) are either 2 hops or 3 hops apart. Thus the "typical" distance between nodes under this metric is very small. Likewise, the overall network diameter is quite small as well.

The fact that most node pairs are quite close together means that the concept of "neighborhood" using this metric is not very useful. For example, if one looks at the 2-hop neighborhood of a typical node, it probably includes around half of all nodes in the graph. Hence, shortest-path distance cannot be expected to give a good sense of locality or modularity in this network, and we observe that if one seeks to use graph structure to infer similarity of function in a PPI network, using shortest-paths to measure distance has serious drawbacks.

One of the reasons that paths are typically short in biological networks like the PPI network is due to the presence of *hubs* – very high degree nodes which represent proteins that have many physical interaction partners. In fact, in the case of PPI networks, hubs often represent proteins with *different* functional roles than their neighbors. For example, chaperone proteins (proteins that help other proteins fold correctly) are often hubs, and are not

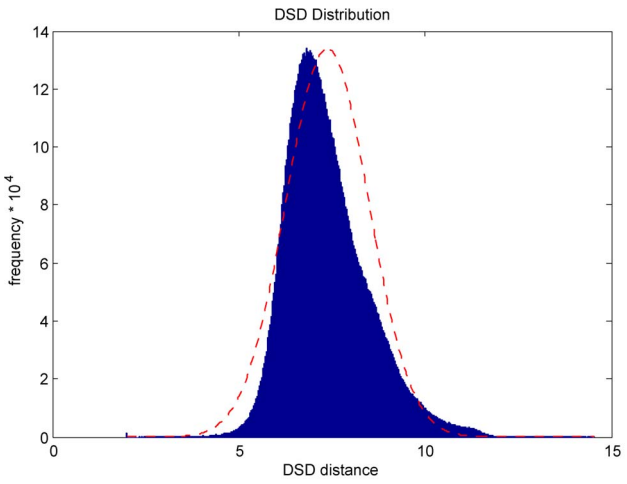

Figure 2. Distribution of DSD in the largest connected component of the yeast PPI network; the red curve represents a fitted normal distribution for comparison.
doi:10.1371/journal.pone.0076339.g002

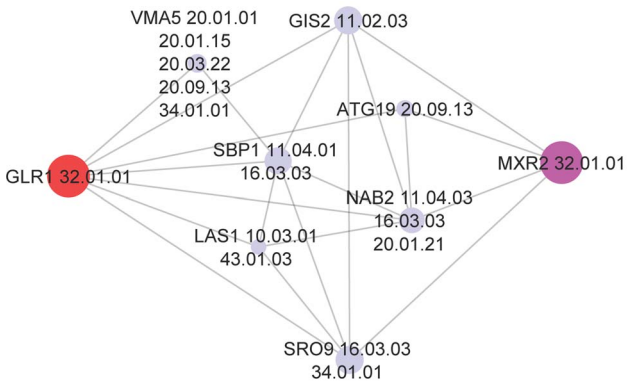

Figure 3. An example of functional annotation with DSD. The correct functional annotation for GLR1, on the third level of the MIPS hierarchy, 32.01.01 (oxidative stress response) is found among none of its direct neighbors, but with the node that is closest in DSD, MXR2. MXR2 is closest in DSD because it has the most similar neighborhood to GLR1.
doi:10.1371/journal.pone.0076339.g003

Table 1. Summary of DSD improvements of all four methods in 2-fold cross validation (mean and standard deviation in percentage) for the PPI network of *S. cerevisiae*.

	MIPS 1		MIPS 2		MIPS 3	
	Accuracy	**F1 score**	**Accuracy**	**F1 score**	**Accuracy**	**F1**
Majority Vote (MV)	50.0±0.5	41.6±0.2	40.7±0.5	30.7±0.4	38.4±0.4	29.5±0.4
MV with DSD	63.7±0.4	47.2±0.2	49.3±0.5	35.6±0.2	43.8±0.4	32.3±0.3
MV (weighted DSD)	63.2±0.5	48.1±0.3	50.6±0.4	36.6±0.2	45.3±0.3	33.6±0.2
Neighborhood (NH)	43.3±0.3	34.5±0.2	32.4±0.6	26.1±0.3	31.3±0.5	24.8±0.3
NH with DSD	51.5±0.4	40.6±0.3	34.8±0.5	27.7±0.2	32.6±0.6	25.1±0.3
Multi-cut	55.2±0.4	42.1±0.2	42.0±0.6	28.1±0.2	36.6±0.4	24.8±0.3
Multi-cut with DSD	58.3±0.3	42.2±0.2	44.6±0.4	29.6±0.1	38.2±0.3	25.3±0.2
Functional Flow (FF)	50.5±0.6	37.0±0.3	32.1±0.4	22.6±0.3	25.4±0.6	18.3±0.3
FF with DSD	54.0±0.4	40.8±0.2	38.3±0.3	27.1±0.3	31.5±0.3	22.8±0.2

doi:10.1371/journal.pone.0076339.t001

typically functionally related to their interaction partners. The same is true for proteins involved in translation. In our network, as an extreme example, the highest degree node is the protein NAB2, which has 2325 physical interactions in our PPI network. Its functional annotation terms: "3′-end processing", "RNA binding" and "RNA transport" [21] suggest that this protein is involved in translation machinery and thus will bind with a highly diverse set of proteins with unrelated function. Hubs are also more likely to be proteins with multiple, distinct functions [22].

Hence, not all short paths provide equally strong evidence of similar function in PPI networks. Consider the network in Figure 1(b). Although nodes B and H are only one hop apart, this does not suggest they are functionally related, since H is a hub. Likewise, B and A are not necessarily functionally related, since they are connected through a hub.

To capture the notion that A and B are not necessarily related, we note that a random walk starting at A is not likely to reach B quickly. If we restrict attention to random walks of, say, 3 hops, then often one will not reach B from A at all.

This motivates the first element of our new metric definition. Given some fixed $k>0$, we define $He^{\{k\}}(A,B)$ to be the expected number of times that a random walk starting at A and proceeding for k steps, will visit B. Note that $He^{\{k\}}(A,B)$ bears some resemblance to previous diffusion approaches suggested for PPI networks, see [23,24], though we will next be building metric structure on top of $He^{\{k\}}(A,B)$ in a completely novel way. For now, note that $He^{\{k\}}(A,B)$ captures our intuition regarding similarity, because node pairs connected by many short paths of low-degree nodes will tend to have high $He^{\{k\}}()$ values. If node pairs with a large $He^{\{k\}}()$ value are then somehow considered 'similar' in our metric, then clearly $He^{\{k\}}()$ does a better job of capturing 'similarity' than does shortest-path distance. This is because A and B are relatively far under this metric; the influence of the hub H decreases the likelihood of a random walk from A landing at B.

But a metric whose notion of similarity is based only on using $He^{\{k\}}()$ between the two given nodes directly does not solve all our problems. In particular, note that $He^{\{k\}}(A,H)$ will indicate strong similarity, even though (as we have argued) A is not likely to be strongly functionally related to H. Furthermore, $He^{\{k\}}()$ is still far from a metric; note it is not even symmetric. Also, $He^{\{k\}}()$ is only a pairwise, and not a global measure of node similarity. This observation leads us to use $He^{\{k\}}()$ in a more subtle way, and in a different manner than previous diffusion approaches, resulting in the definition of DSD, which we describe next.

Definition of the New Distance Metric

Consider the undirected graph $G(V,E)$ on the vertex set $V=\{v_1,v_2,v_3,...,v_n\}$ and $|V|=n$. Recall that $He^{\{k\}}(A,B)$ is defined as the expected number of times that a random walk starting at node A and proceeding for k steps, will visit node B. In what follows, assume k is fixed, and when there is no ambiguity, in the value of k, we will denote $He^{\{k\}}(A,B)$ by $He(A,B)$. We further define a n-dimensional vector $He(v_i),\forall v_i\in V$, where

$$He(v_i)=(He(v_i,v_1),He(v_i,v_2),...,He(v_i,v_n)).$$

Then, the Diffusion State Distance (DSD) between two vertices u and v, $\forall u,v\in V$ is defined as:

$$DSD(u,v)=||He(u)-He(v)||_1.$$

where $||He(u)-He(v)||_1$ denotes the L_1 norm of the He vectors of u and v.

We will show for any fixed k, that DSD is a metric, namely that it is symmetric, positive definite, and non-zero whenever $u\neq v$, and it obeys the triangle inequality. Thus, one can use DSD to reason about distances in a network in a sound manner. Further, we show that DSD converges as the k in $He^{\{k\}}(A,B)$ goes to infinity, allowing us to define DSD independent from the value k.

Characteristics of DSD

Figure 2 shows the distribution of DSD values in the PPI network of *S. cerevisiae* as downloaded in BioGRID [20]. The figure shows that DSD distances take on a smooth, continuous range of values. This is in sharp contrast to shortest-path distances, as shown in Figure 1(a), and shows that DSD is capable of making much finer-grained distinctions of similarity over pairs of nodes in the network.

Figure 3 shows a typical example illustrating the nature of DSD. The Figure shows the gene GLR1 (in red at left) along with a subset of its local neighborhood. Node sizes have been made inversely proportional to their DSD distance from GLR1. None of

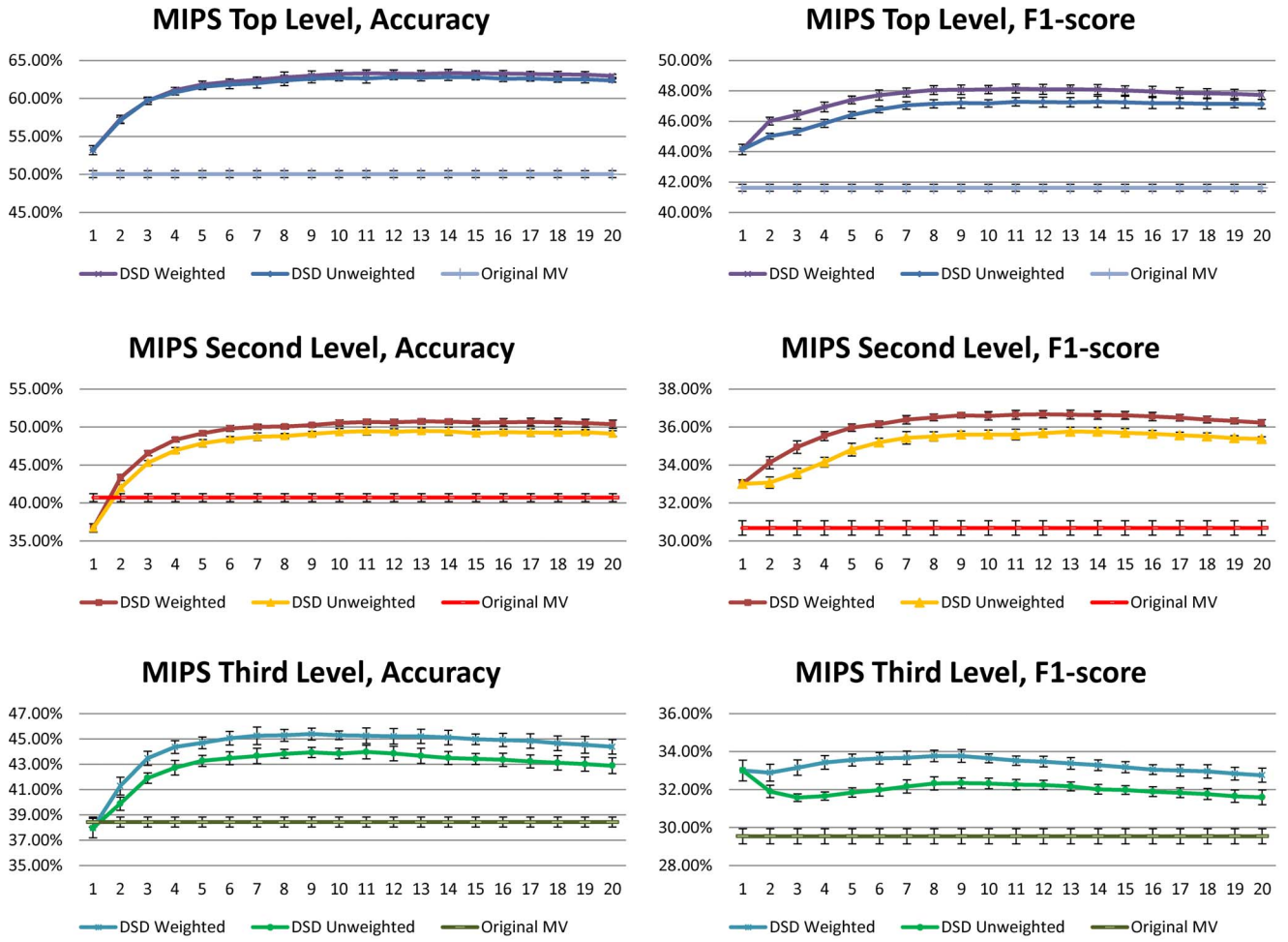

Figure 4. Improvement on Accuracy and F1 Score for at different neighborhood thresholds, for the majority voting algorithm in 10 runs of 2-fold cross validation for *S. cerevisiae*, with standard deviations.
doi:10.1371/journal.pone.0076339.g004

GLR1's immediate neighbors contain its correct functional label at the third level of the MIPS hierarchy (32.01.01: oxidative stress response). However, the node closest in DSD is MXR2, which has exactly the same functional label. DSD recognizes that nodes having large common low-degree neighborhoods are highly similar and correctly identifies functionally similar node pairs, and does so in situations where shortest-path distance fails.

Methods

Functional Categories

We continue to work with the dataset described above, the largest connected component from an experimentally derived physical interaction PPI network from *S. cerevisiae* with 4990 nodes, and 74,310 edges. As in the papers introducing the classical function prediction methods we consider, we primarily use the MIPS (Munich Information Center For Protein Sequences) functional catalogue (FunCat) for our functional category labels [21]. We use the latest version of FunCat (version 2.1) and the first, second and third level functional categories, retaining those for which at least three proteins in our dataset are annotated with those labels. We present results for the first level (17 functional categories) second level (74 of the 80 functional categories that annotate at least one protein, annotate at least 3 proteins) and third level (154 of 181 functional categories that annotate at least one protein, annotate at least 3 proteins) MIPS annotations. MIPS is a shallow, leveled, hierarchical classification scheme for protein function, but we also present results for the popular Gene Ontology (GO) [25], where the variable depth hierarchy of the annotation labels makes the evaluation of labeling methods more complicated.

We assumed all published labels were correct and did not attempt for this study to weigh them by a level of confidence in each type of biological experiment. The following classical methods were tested in their original form (using shortest-path distance), against a DSD variant in cross validation.

Cross Validation Task

We considered 2-fold cross validation tasks. In each of the 2-fold cross validation tasks, we first randomly split the annotated proteins into 2 sets, and consider only the annotations on one partition in the training set, when trying to predict the annotations on proteins in the test set, and average the performance over the 2 folds of the cross validation. We conduct 10 runs of 2-fold cross-validation and report the mean and standard deviation of the following performance measures over these 10 runs.

Accuracy. This is the performance measurement suggested in [1]. Each protein is assigned its highest-scoring functional label.

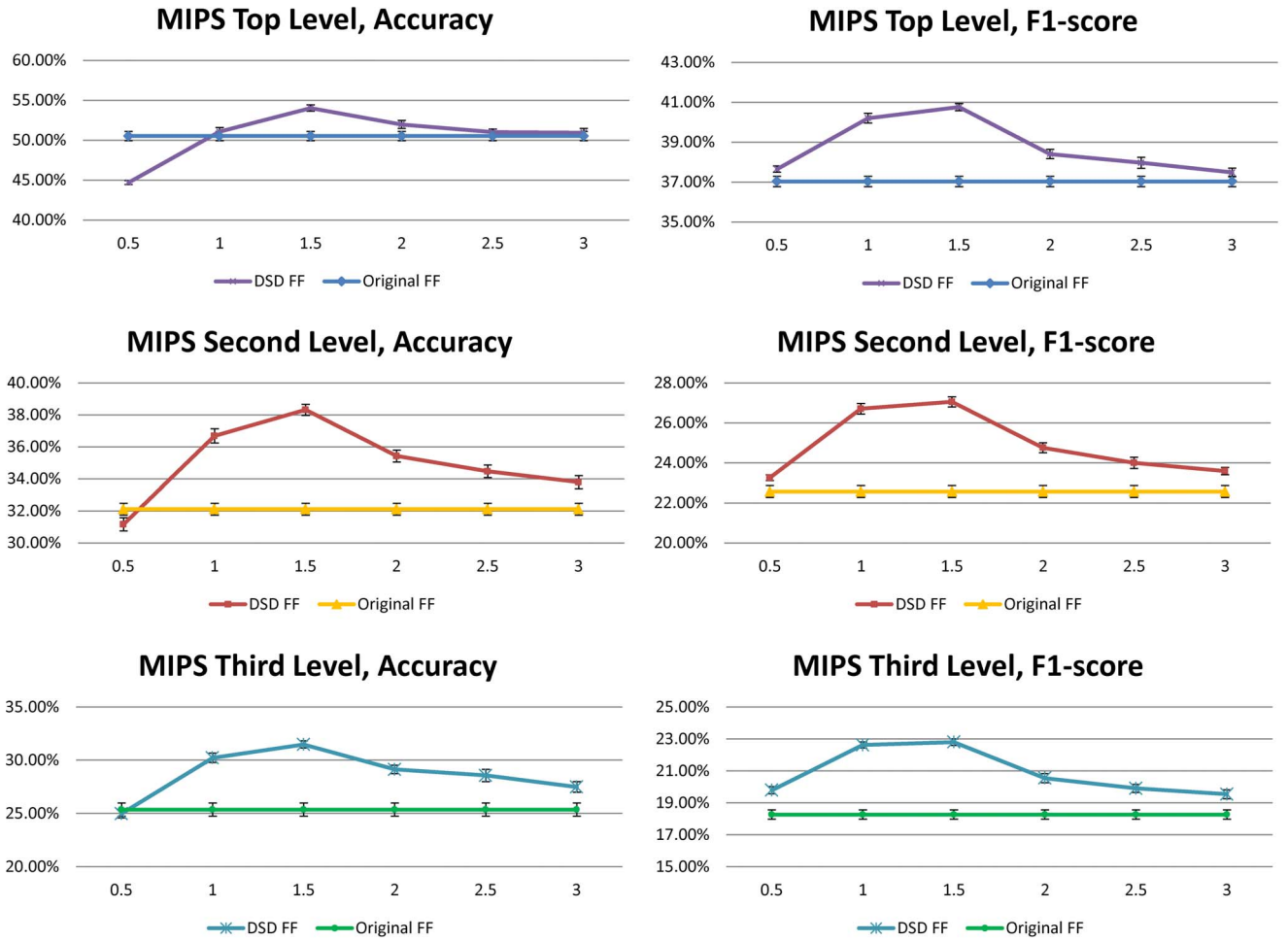

Figure 5. Improvement on Accuracy and F1 Score for DSD at different neighborhood thresholds, for the functional flow (FF) algorithm in 10 runs of 2-fold cross validation for *S. cerevisiae*, with standard deviations.
doi:10.1371/journal.pone.0076339.g005

The label is considered correct if it appears among the labels that were assigned to the protein. We calculate the percentage of proteins that are assigned a correct label.

F1 score. This is the performance measurement suggested in [26]. For each protein, the possible functional labels the algorithm could assign are stored in a ranked list according to score. Each label is considered correct if it appears among the labels that were assigned to the protein, and incorrect otherwise. We calculate precision and recall by looking at the top α (in our case, we present results for $\alpha = 3$) predicted annotations. Then the F1 score for each function can be calculated as:

$$F1 = \frac{2 * precision * recall}{precision + recall}$$

We average F1 scores over the individual functions and obtain the overall F1 score for each algorithm.

Protein Function Prediction Methods

Neighborhood Majority Voting Algorithm. This is the simplest of all function prediction methods. Directly applying the concept of 'guilt by association', Schwikowski et al. [1] considered for each protein $u \in V$ its neighboring proteins. Each neighbor

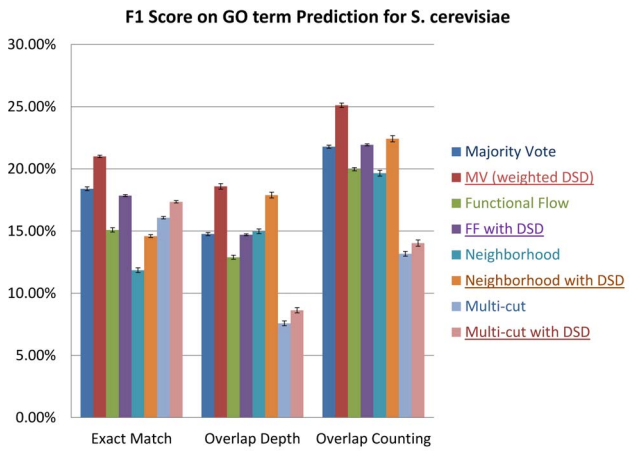

Figure 6. Improvement on F1 Score for DSD using three evaluation methods: exact match, overlap depth and overlap counting, on informative GO terms for the four algorithms for *S. cerevisiae* in 10 runs of 2-fold cross validation.
doi:10.1371/journal.pone.0076339.g006

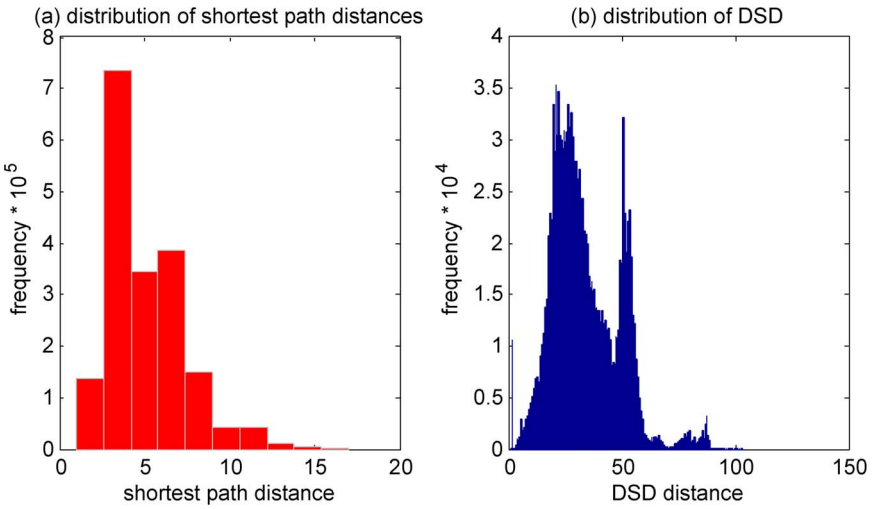

Figure 7. Shortest path distance and DSD distribution for *S. pombe*.
doi:10.1371/journal.pone.0076339.g007

votes for their own annotations, and the majority is used as the predicted functional label. To incorporate DSD, the neighborhood of u is defined simply as the t nearest neighbors of u under the DSD metric. Furthermore, two schemes are considered: an unweighted scheme where all new neighbors vote equally, and a DSD weighted scheme where all new neighbors get a vote proportional to the reciprocal of their DSD distance.

χ^2 Neighborhood Algorithm. In the original χ^2 neighborhood algorithm [3], each annotation a present in protein u's neighborhood will be assigned a χ^2 value based on the following formula:

$$\chi_a^2 = \frac{(n_a - e_a)^2}{e_a}$$

where n_a is the observed number of annotations a in the neighborhood, and e_a is the expected number of annotations a based on the whole protein-protein interaction network. Then protein u is assigned the functional label a with the maximum χ^2 value. Again, it is straightforward to adapt this to use DSD: the neighborhood of u is simply defined as the t closest nodes to u under the DSD metric.

Multi-way cut Algorithm. We consider the minimal multi-way k-cut algorithm of Vazquez et al. [9] as implemented by [7]. The motivation is to minimize the number of times that annotations associated with neighboring proteins differ. In particular, the dual ILP (integer linear programming) is formulated, so we instead seek to maximize

$$\sum_{(u,v)\in E, a\in FUNC} X_{u,v,a}$$

subject to the constraints $\sum_{a\in FUNC} X_{u,a} = 1$, $X_{u,v,a} \le X_{u,a}$, $X_{u,v,a}\in\{0,1\}, X_{v,a}\in\{0,1\}$ where the edge variables $X_{u,v,a}$ are defined for each function a whenever there exists an edge between proteins u and v. It is set to 1, if protein u and v both are assigned function a, and 0 otherwise. The node variables $X_{u,a}$ are set to 1 when u is labeled with function a and 0 otherwise. The first constraint insures that each protein is only given one annotation. The second constraint makes sure only annotations that appear among the vertices can be assigned to the edges. While this problem is NP-hard, the ILP is tractable in practice; in our case we use the IBM CPLEX solver (version 12.4, dated 12/2011, http://www.ilog.com/products/cplex/). For the DSD version, we simply add additional edges between vertices whose DSD is below a threshold. We set a global threshold D based on the average DSD of all pairs, specifically we set $D = \mu - c * \sigma$, where μ is the average, and σ is the standard deviation of the global set of DSD values among all pairs of nodes in the graph. We experiment with c in the range $\{1.5,2.0,2.5,3\}$.

Functional Flow Algorithm. Nabieva at al. [7] use a network flow algorithm on the graph of protein interactions to label proteins. The idea is to consider each protein having a known function annotation as a 'reservoir' of that function, and to simulate flow of functional association through the network to make predictions. We adapt the approach to use DSD by creating an edge between each node pair, with a weight inversely proportional to DSD. For computational efficiency we do not create edges when the reciprocal of DSD is below a small value. As in the original functional flow, we calculate flow through this new network at each time step. We denote the size of the reservoir of function a at node u and time step i, to be $R_i^a(u)$. For a given function (annotation) a, we initialize the reservoir size at node u to be infinite if protein u has been annotated with function a; otherwise we set it to be 0. More formally:

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

We then update the reservoir over a sequence of timesteps (we use 6 timesteps, as in the original version):

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

where $g_t^a(v,u)$ is the amount of flow a that moves from u to v at time t. We incorporate DSD into the edge weight as follows:

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

The final functional score for node u and function a over 6 timesteps is computed as the total amount of incoming flow.

Formal Properties of DSD

We now present the formal proofs that DSD is a metric for any fixed k, that it converges as k goes to infinity, and obtain the explicit form that allows the calculation of DSD values in the limit.

Lemma 1. *DSD* is a metric on V, where V is the vertex set of a simple connected graph $G(V,E)$.

Proof. Clearly the DSD of a node to itself is 0, and DSD is non-negative and symmetric. It remains only to show that DSD satisfies the triangle inequality, and that $DSD(u,v)$ is strictly positive whenever $u \neq v$.

For all $u,v,w \in V$, we have by the L_1 norm property:

$$||He(u)-He(v)||_1+||He(v)-He(w)||_1 \geq ||He(u)-He(v)+He(v)-He(w)||_1$$

and therefore:

$$||He(u)-He(v)||_1+||He(v)-He(w)||_1 \geq ||He(u)-He(w)||_1.$$

Thus, the triangle inequality follows easily:

$$DSD(u,v)+DSD(v,w) \geq DSD(u,w).$$

Next we prove the *identity of indiscernibles*, namely, $DSD(u,v)$ is non-zero for all $u \neq v$. We first need some notation. We define the one step transition probability matrix P as the n-dimensional square matrix, whose (i,j)th entry is given by:

$$p_{ij}=\begin{cases}\frac{1}{d_i} & \text{if } (v_i,v_j)\in E\\ 0 & \text{otherwise}\end{cases}$$

where d_i is the degree of node v_i $\forall i=1,2...,n$. Note that P represents the probability to reach each neighbor in the random walk where all neighbors are reached with equal probability.

Then the definition of k step transition probability matrix $P^{\{k\}}=P^k$ follows for all positive k. We also define the 0 step transition matrix $P^{\{0\}}$ to be the identity matrix I, because every random walk starts from the source vertex, and thus a zero-step random walk reaches its source vertex with probability 1 and all other vertices with probability 0. We denote by $p_{ij}^{\{l\}}$ or $p_{v_i,v_j}^{\{l\}}$ the (i,j)th entry in the lth transition probability matrix, where $l \geq 0$ and $i,j \in \{1,2,...,n\}$.

For each ordered pair of vertices (v_i,v_j), recall that $He(v_i,v_j)$ is defined as the expected number of times that a random walk with length k starting from v_i will visit v_j. In order to calculate $He(v_i,v_j)$, we define an indicator variable $I_{ij}^{\{l\}}, \forall i,j,l$, where:

$$I_{ij}^{\{l\}}=\begin{cases}1 & \text{if the random walk starting from } u_i \text{ visits } u_j \text{ at the } l\text{th step}\\ 0 & \text{otherwise}\end{cases}$$

Therefore we have $He(v_i,v_j)=E(\sum_{l=0}^{k} I_{ij}^{\{l\}})=\sum_{l=0}^{k} E(I_{ij}^{\{l\}})$ by linearity of expectation. Clearly, $E(I_{ij}^{\{l\}})=p_{ij}^{\{l\}}$, and thus we have $He(v_i,v_j)=\sum_{l=0}^{k} p_{ij}^{\{l\}}$. Note that because we are adding the zero-step transition matrix, we will have that $He^{\{l\}}(v_i,v_i) \geq 1$, $\forall u_i,u_j \in V$.

Now we are ready to show the *identity of indiscernibles*. Recall we are assuming the graph is connected. It is trivial when $n=|V|=2$, so consider the case where $n \geq 3$. We prove this next by contradition.

Assume there exists a pair of distinct vertices a and $b \in V$ in the simple connected graph G, where $DSD(a,b)=0$. Thus $||He^{\{k\}}(a)-He^{\{k\}}(b)||_1=0$, and $He(a)=He(b)$, which indicates that $He(a,v_i)=He(b,v_i)$, $\forall v_i \in V$. We have n equations now: $\sum_{l=0}^{k} p_{a,v_i}^{\{l\}}=\sum_{l=0}^{k} p_{b,v_i}^{\{l\}}$. Since the zero-step transition matrix $P^{\{0\}}$ is an identity matrix, we know that $p_{v_i,v_j}^{\{0\}}=1$ for $i=j$ and 0, otherwise.

Thus we have

$$\begin{cases}1+\sum_{l=1}^{k} p_{a,a}^{\{l\}}=\sum_{l=1}^{k} p_{b,a}^{\{l\}}\\ \sum_{l=1}^{k} p_{a,b}^{\{l\}}=1+\sum_{l=1}^{k} p_{b,b}^{\{l\}}\\ \sum_{l=1}^{k} p_{a,v_i}^{\{l\}}=\sum_{l=1}^{k} p_{b,v_i}^{\{l\}}\end{cases} \quad (1)$$

where the third line represents a set of $n-2$ equations: one for each v_i, distinct from a and b.

By multiplying a factor $p_{v_i,a}=p_{v_i,a}^{\{1\}}$ to both sides of all equations in the third line and summing over i, we have:

$$\sum_{\substack{i=1,\\ v_i\neq a,\\ v_i\neq b}}^{n}\sum_{l=1}^{k} p_{a,v_i}^{\{l\}}\cdot p_{v_i,a}=\sum_{\substack{i=1,\\ v_i\neq a,\\ v_i\neq b}}^{n}\sum_{l=1}^{k} p_{b,v_i}^{\{l\}}\cdot p_{v_i,a}$$

By completing the sum over i, we have:

$$\sum_{i=1}^{n}\sum_{l=1}^{k} p_{a,v_i}^{\{l\}}\cdot p_{v_i,a}-\sum_{l=1}^{k} p_{a,a}^{\{l\}}\cdot p_{a,a}-\sum_{l=1}^{k} p_{a,b}^{\{l\}}\cdot p_{b,a}=$$
$$\sum_{i=1}^{n}\sum_{l=1}^{k} p_{b,v_i}^{\{l\}}\cdot p_{v_i,a}-\sum_{l=1}^{k} p_{b,a}^{\{l\}}\cdot p_{a,a}-\sum_{l=1}^{k} p_{b,b}^{\{l\}}\cdot p_{b,a}$$

G doesn't contain self-loops; thus $(u,u) \notin E$ and $p_{u,u}=0$, $\forall u \in V$. For each $l \geq 1$, we have

$$\sum_{i=1}^{n} p_{a,v_i}^{\{l\}}\cdot p_{v_i,a}=p_{a,a}^{\{l+1\}}$$

and

$$\sum_{i=1}^{n} p_{b,v_i}^{\{l\}}\cdot p_{v_i,a}=p_{b,a}^{\{l+1\}}$$

Therefore, we have

$$\sum_{l=1}^{k} p_{a,a}^{\{l+1\}}-\sum_{l=1}^{k} p_{a,b}^{\{l\}}\cdot p_{b,a}=\sum_{l=1}^{k} p_{b,a}^{\{l+1\}}-\sum_{l=1}^{k} p_{b,b}^{\{l\}}\cdot p_{b,a}$$

namely,

$$(\sum_{l=2}^{k} p_{a,a}^{\{l\}} - \sum_{l=2}^{k} p_{b,a}^{\{l\}}) + p_{a,a}^{\{k+1\}} - p_{b,a}^{\{k+1\}} = p_{b,a} \cdot (\sum_{l=1}^{k} p_{a,b}^{\{l\}} - \sum_{l=1}^{k} p_{b,b}^{\{l\}})$$

By applying to the equation above the first and the second equation in (1), where:

$$\begin{cases} \sum_{l=2}^{k} p_{a,a}^{\{l\}} - \sum_{l=2}^{k} p_{b,a}^{\{l\}} = p_{b,a} - 1 \\ \sum_{l=1}^{k} p_{a,b}^{\{l\}} - \sum_{l=1}^{k} p_{b,b}^{\{l\}} = 1 \end{cases} \quad (2)$$

we have:

$$p_{b,a} - 1 + p_{a,a}^{\{k+1\}} - p_{b,a}^{\{k+1\}} = p_{b,a} \cdot 1$$

namely,

$$p_{a,a}^{\{k+1\}} - p_{b,a}^{\{k+1\}} = 1.$$

Since $p_{a,a}^{\{k+1\}}, p_{b,a}^{\{k+1\}} \in [0,1]$, we have $p_{a,a}^{\{k+1\}} = 1$.

We next argue that it must be the case that $d(v_i) = 1$, for all v_i with $(v_i,a) \in E$, namely, all of a's neighbors must have degree 1.

Starting from

$$1 = p_{a,a}^{\{k+1\}} = \sum_{(v_i,a)\in E} p_{a,v_i}^{\{k\}} \cdot p_{v_i,a} + \sum_{(v_i,a)\notin E} p_{a,v_i}^{\{k\}} \cdot p_{v_i,a} =$$

$$\sum_{(v_i,a)\in E} p_{a,v_i}^{\{k\}} \cdot p_{v_i,a} \leq \sum_{(v_i,a)\in E} p_{a,v_i}^{\{k\}} \leq 1,$$

we must therefore have $\sum_{(v_i,a)\in E} p_{a,v_i}^{\{k\}} \cdot p_{v_i,a} = \sum_{(v_i,a)\in E} p_{a,v_i}^{\{k\}} = 1$, and $\forall v_i : (v_i,a) \in E$ and $p_{a,v_i}^{\{k\}} > 0$, $p_{v_i,a} = 1$. Since we have $\sum_{(v_i,a)\in E} p_{a,v_i}^{\{k\}} = 1$, there must exist at least one neighbor x, s.t. $p_{a,x}^{\{k\}} > 0$, and $p_{x,a} = 1$ as well. Because $p_{x,a} = 1$, we know that a is the only neighbor to x and $d(x) = 1$. Also due to the fact that $p_{a,x}^{\{k\}} > 0$, there exists a path of length k from a to x, which should consists of two parts, one path (denoted as $path^*(a \to a)$) of length $k-1$ from a to a and an edge from a to x as the last step in the path.

Consider any neighbor $y : (y,a) \in E$. By using the $path^*(a \to a)$ and the edge $(a,y) \in E$, we can construct a path of length k from a to y and therefore $p_{a,y}^{\{k\}} > 0$. Thus, $p_{y,a} = 1$, and therefore $d(y) = 1$. Thus we have shown that it must be the case that $d(v_i) = 1$, for all v_i, with $(v_i,a) \in E$.

As a result, all of a's neighbors are only connected to a.

If $(a,b) \in E$, then a and b can't have any more neighbors because they must be the only neighbor of each other; thus for all $c \in V - \{a,b\}$, c is not connected to a or b, which contradicts the fact that the graph G is connected. If $(a,b) \notin E$, a and b are not connected because all of their neighbors are only adjacent to themselves respectively.

Therefore, the existence of such pair (a,b) where $a \neq b$ and $DSD^{\{k\}}(a,b) = 0$ contradicts the fact that the graph is connected, and we can conclude that $DSD(a,b) = 0$ if and only if $a = b$.

In the above we were reasoning about DSD for a fixed value of k. Denote by $DSD^{\{k\}}$ the value of DSD for a particular fixed k. Next we discuss how DSD values depend on k. We show that when the one-step transition matrix P is diagonalizable then $DSD^{\{k\}}(u,v)$ converges to a stationary value $DSD(u,v)$.

Lemma 2. Let G be a connected graph whose random walk one-step transition probability matrix P is diagonalizable and ergodic as a Markov chain, then for any $u,v \in V$, $DSD^{\{k\}}(u,v)$ converges as k, the length of the random walk, approaches infinity.

Proof. Since the one step transition probability matrix P is diagonalizable, we have by diagonalization:

$$P = V \Lambda U,$$

where the orthogonal matrix $V = U^{-1}$; each column of V is the normalized right eigenvector of P, each row of U is the normalized left eigenvector of P, and $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_n)$ is the diagonal matrix of all eigenvalues where $1 = \lambda_1 > |\lambda_2| \geq ... \geq |\lambda_n|$ by ergodicity. Let $\boldsymbol{v_1}, \boldsymbol{v_2}, ..., \boldsymbol{v_n}$ denote the normalized orthogonal column vectors in V and $\boldsymbol{u_1^T}, \boldsymbol{u_2^T}, ..., \boldsymbol{u_n^T}$ the normalized orthogonal row vectors in U. We denote v_{ij} as the jth entry in vector $\boldsymbol{v_i}$ and u_{ij} as the jth entry in vector $\boldsymbol{u_i}$.

Thus it follows that $P^{\{t\}} = V \Lambda^t U$, namely $\forall i,j \in V, t \geq 1$:

$$p_{i,j}^{\{t\}} = \sum_{c=1}^{n} \lambda_c^t v_{ci} u_{cj} = v_{1i} u_{1j} + \lambda_2^t \sum_{c=2}^{n} (\frac{\lambda_c}{\lambda_2})^t v_{ci} u_{cj}$$

Next, for all $i,j,\beta \in V$, we define an infinite sequence $A^t(i,j,\beta)$ *for* $t = 1,2,...,\infty$:

$$A^t(i,j,\beta) = p_{i,\beta}^{\{t\}} - p_{j,\beta}^{\{t\}}$$

$$= \sum_{c=2}^{n} \lambda_c^t v_{ci} u_{c\beta} - \sum_{c=2}^{n} \lambda_c^t v_{cj} u_{c\beta}$$

$$= \sum_{c=2}^{n} \lambda_c^t (v_{ci} - v_{cj}) u_{c\beta}$$

and $A^0(i,j,\beta) = I(i,\beta) - I(j,\beta)$ where I is the identity matrix.

Therefore, by definition, we can rewrite $DSD^{\{k\}}$ as the partial sum:

$$DSD^{\{k\}}(u,v) = \sum_{\beta\in V} |\sum_{t=0}^{k} A^k(u,v,\beta)|.$$

Claim 1. For all $u,v,\beta \in V$, $\sum_{t=0}^{k} A^t(u,v,\beta)$ converges absolutely.

Before we prove the claim, we show that it produces what we need. If the claim is true, then the limit $|\lim_{k\to+\infty} \sum_{t=0}^{\{k\}} A^t(u,v,\beta)|$ exists, and we can denote it as $A^*(u,v,\beta)$. Then it follows that:

$$\lim_{k\to+\infty} DSD^{\{k\}}(u,v) = \sum_{\beta\in V} |\lim_{k\to+\infty} \sum_{t=0}^{k} A^t(u,v,\beta)|$$

$$= \sum_{\beta\in V} A^*(u,v,\beta).$$

and we have proved convergence.

But Claim 1 is true from the Cauchy root test, namely,

$$\sqrt[t]{|A^t(i,j,\beta)|} = \sqrt[t]{|\sum_{c=2}^{n} \lambda_c^t (v_{ci} - v_{cj}) u_{c\beta}|} \quad (3)$$

$$\leq \sqrt[t]{\sum_{c=2}^{n} |\lambda_2|^t \sqrt{2} \cdot 1} \quad (4)$$

$$\leq \sqrt[t]{n\sqrt{2}\cdot\lambda_2^t} \to |\lambda_2| < 1, \textit{ when } t\to\infty \quad (5)$$

where (3) & (5) hold by definition and (4) holds because $1 = \lambda_1 > |\lambda_2| \geq \ldots \geq |\lambda_n|$ and $||\boldsymbol{v_c}||_2 = ||\boldsymbol{u_c}||_2 = 1$.

We now use this lemma to produce an explicit way to calculate in the limit. This frees DSD entirely from dependence on the parameter k, and this is the version of DSD we use in our experiments. (We remark, that for our yeast PPI network, we found empirically that $DSD^{\{k\}}$ values were very close to the limit when $k \geq 5$.)

Lemma 3. Let G be a connected graph whose random walk one-step transition probability matrix P is diagonalizable and ergodic as a Markov chain, then for any $u,v \in V$, we have $\lim_{k\to\infty} DSD^{\{k\}}(u,v) = (\boldsymbol{b_u^T} - \boldsymbol{b_v^T})(I - P + W)^{-1}$, where I is the identity matrix, W is the constant matrix in which each row is a copy of $\boldsymbol{\pi}^T$, $\boldsymbol{\pi}^T$ is the unique steady state distribution, and for any $i \in V$, $\boldsymbol{b_i^T}$ is the i^{th} basis vector, i.e., the row vector of all zeros except for a 1 in the i^{th} position.

Proof. To start, we denote the matrix $P - W$ by D. By Lemma 2 where we have shown that $\lim_{k\to\infty} DSD^{\{k\}}(u,v)$ exists and is finite, we can denote the limit as simply $DSD(u,v)$ in the following context. It follows that for any $u,v \in V$:

$$DSD(u,v) = \lim_{k\to\infty} DSD^{\{k\}}(u,v)$$

$$= \| \lim_{k\to\infty} He^{\{k\}}(u) - He^{\{k\}}(v) \|_1$$

$$= \| \lim_{k\to\infty} (\boldsymbol{b_u^T} - \boldsymbol{b_v^T})(I + P + P^2 + \ldots + P^k) \|_1 \quad (6)$$

$$= \| \lim_{k\to\infty} (\boldsymbol{b_u^T} - \boldsymbol{b_v^T})(I + (P - W) + (P^2 - W) + \ldots + (P^k - W)) \|_1 \quad (7)$$

$$= \| \lim_{k\to\infty} (\boldsymbol{b_u^T} - \boldsymbol{b_v^T})(I + D + D^2 + \ldots + D^k) \|_1 \quad (8)$$

$$= \| (\boldsymbol{b_u^T} - \boldsymbol{b_v^T})(I - D)^{-1} \|_1 \quad (9)$$

where (6) holds by definition, (7) holds because $\boldsymbol{b_u^T} W = \boldsymbol{b_v^T} W = \boldsymbol{\pi^T}$, (8) holds because $(P^n - W) = (P - W)^n$ for all $n \geq 1$ by Claim 2, and (9) holds by Claim 3, where we finish the proof of Lemma 3 by proving Claims 2 and 3 next.

Claim 2. $(P^n - W) = (P - W)^n$, for any non-zero positive integer n.

Proof. The proof is based on the following observations: $WP = W$ and $PW = W$, so $WP^k = W$ and $P^k W = W$. Further, $W^m = W$, so $W^k P^m = W$ and $P^m W^k = W$ for any integers $k > 0$ and $m > 0$. Now, if you construct the binomial expansion of $(P - W)^n$, each cross term of the type $W^k P^m$ or $P^m W^k$ reduces to W, and all of the Ws cancel out except for one, leaving $(P^n - W)$. Thus, $(P^n - W) = (P - W)^n$.

Claim 3. $\lim_{t\to\infty} I + (P - W) + (P^2 - W) + \ldots + (P^t - W) = (I - P + W)^{-1}$.

Proof. By Claim 2, $I + (P - W) + (P^2 - W) + \ldots + (P^k - W) = 1 + D + D^2 + \ldots + D^k$

1. First we show that $(I - P + W)^{-1} = (I - D)^{-1}$ exists. We show that $(I - D)$ has full rank by showing that 0 is the only vector in the left nullspace of $(I - D)$. That is, if $\boldsymbol{x^T}(I - D) = 0$, then $\mathbf{x} = \mathbf{0}$.

$$\boldsymbol{x}^T(I - D) = 0$$

$$\boldsymbol{x}^T - x^T D = 0$$

$$\boldsymbol{x}^T D = \mathrm{x}^T$$

$$\boldsymbol{x}^T D^k = \boldsymbol{x}^T \quad \textit{for any } t > 0$$

Now, we already know that $\lim_{k\to\infty} D^k = \lim_{k\to\infty} P^k - W$ exists and is 0 by ergodicity. So we have

$$\lim_{t\to\infty} \boldsymbol{x}^T D^k = \boldsymbol{x}^T$$

$$\boldsymbol{x}^T 0 = \boldsymbol{x}^T$$

which is only satisfied by by $\boldsymbol{x} = \mathbf{0}$.

1. Since $(I - D)^{-1}$ exists and $\lim_{k\to\infty} D^k = 0$, we can calculate as follows:

$$C = I + D + D^2 + \ldots$$

$$DC = D + D^2 + \ldots$$

$$C - DC = I$$

$$C = (I - D)^{-1}$$

where C is defined as $I + \lim_{k\to\infty} \sum_{t=1}^{k} D^t$, which exists because $(I-D)^{-1}$ exists and $\lim_{k\to\infty} D^k = 0$.

That finishes the proof for Lemma 3.

Results

MIPS Results

Both the DSD majority voting method and the χ^2 neighborhood method sort vertices in order of their smallest DSD to a given vertex, and set a threshold t, where the first t vertices participate in the functional vote. Table 1 gives the results in 2-fold cross validation for both these methods when we set $t = 10$, whereas Figure 4 gives more details about the dependence on t; basically once enough neighbors were included (i.e. $t \geq 8$) results of the DSD version of majority voting converged. Similar details about how the χ^2 neighborhood method depends on t appears in (Figure S1). For the multi-way cut and functional flow results, we considered vertices to be in the DSD neighborhood of a node v if their DSD from v was less than c standard deviations below the mean DSD value across the entire dataset. Table 1 presents the 2-fold cross validation DSD multi-way cut results and DSD functional flow results with $c = 1.5$. In addition, Figure 5 gives more detail about the dependence of DSD functional flow on the parameter c, where we tested $c \in \{0.5, 1.0, 1.5, 2.0, 2.5, 3\}$. Similar details about how the multiway cut method depends on c appear in (Figure S2).

We notice from Table 1 that in every case, the DSD versions of all four methods perform better than the versions based on ordinary distance. It is particularly interesting that using DSD improves the functional flow algorithm, since functional flow already seeks to capture a notion of diffusion in the graph. In fact, the DSD versions of Majority Voting perform better than all four of the methods based on ordinary distance. The best performing algorithms among all the ordinary distance algorithms on the yeast MIPS annotations at all three levels of the MIPS hierarchy were the majority voting algorithms. The best performing algorithms overall are the DSD version of the majority voting algorithms, and they clearly achieve the best performance compared to all four ordinary distance and all three other DSD-based methods. In particular, they achieve 63.7% mean accuracy and 47.2% mean F1 score (unweighted) and 63.2% mean accuracy and 48.1% mean F1 score (weighted) on the first level of the MIPS hierarchy. This is compared to the original majority voting algorithm, which only obtains 50.0% mean accuracy and 41.6% mean F1 score. Hence DSD provides more than 13% improvement in accuracy and more than 5% improvement in F1 score for MIPS top level functional categories, and improves on ordinary DSD on the second and third levels of the MIPS hierarchy across the board.

GO Results

GO (Gene Ontology) [25] is a deeply hierarchical classification scheme, which makes defining and evaluating function prediction methods much more complicated. For this reason, most function prediction methods for yeast PPI networks use the "flatter" set of MIPS categories, but GO is the more widely used ontology. Thus we wished to measure performance for GO functional labels, despite the difficulties. For example, in GO, sometimes, nodes are labeled with child functions but not labeled with their parent functions, even though it is assumed that child functions are more specific and inherit the functions of their parent nodes. How much credit should be given to a node when we do not label it correctly at its most specific level, but succeed in labeling it with a less specific ancestor term? The exact match and functional path methods of Deng et al [27,28] are designed to directly answer this question and evaluate methods that perform GO annotation. We tested the performance of ordinary distance versus DSD versions all four protein function prediction methods considered above also on the GO, and evaluated the results using the exact match method and functional path method of Deng et al. [27]. We find, again, that using the DSD-based algorithm improves performance.

We consider labels in the biological process category of the GO hierarchy (we used OBO v1.2 [29], data version 2013-07-18) along with annotations downloaded from the SGD database (data version 2013-07-13). We exclude GO terms that are annotated with evidence codes "IEA" "RCA" or "IPI". For each protein that was labeled with a term in the GO hierarchy, we automatically also label it with all more general parent terms in the GO hierarchy as well. We then, define the *informative* nodes in the GO hierarchy to be functional annotation terms that 1) are at least three levels below the root and 2) are terms that annotate more than 50 proteins in our dataset, where the second condition on informative nodes, and the number 50 is suggested by the method of Deng et al. [27]. It is these GO functional terms, somehow capturing the middle levels in functional specificity, that we use instead of a level of the MIPS hierarchy for the functional labels. The result is 136 informative biological process GO terms among 4322 out of 4990 ORFs that will comprise our labeling set (the total number of annotations is 58,519); thus we have a label set that is close in size to the one from the third level of the MIPS hierarchy. The "exact match" method then simply counts the number of correct labels, just as we did for a fixed level of the MIPS hierarchy. We see similar improvements for GO annotation as we did for MIPS annotation; detailed results for accuracy and F1 score for all four methods appear in (Figure S3).

However, note that this exact match evaluation gives no credit when we label a protein with an incorrect child that still has many ancestor terms in common with the correct label. Thus, for a more fair evaluation of the predictions that takes into account hierarchical relations among the GO labels, we also use the functional path method of Deng et al. [27]. This presents a plausible way to give partial credit when a node is labeled partially correctly, in the sense that the lowest depth child label assigned to the node and the correct label of the node are different, but if we consider the path in the ontology from the root to these two labels, there are ancestor labels these two functional paths have in common. We use exactly the functional path method of [27] to calculate precision and recall values for each protein, and then overall precision and recall values are averaged over all the proteins. We calculated both alternative ways to count functional paths overlap presented in Deng et al. [27], one that simply counts the number of nodes that appear jointly in sets of known and predicted functional annotations (which we will call the overlap counting method) and the other which takes into account at what depth the functional paths from both sets diverge (which we will call the overlap depth method).

Setting t, the neighborhood threshold for DSD majority voting anywhere in the range starting from 2, the DSD version of majority voting improves precision and recall simultaneously, as

compared to original majority voting, regardless of the method used for counting overlaps. For example at $t=8$, we get precision = 67.3% ±0.3% and recall = 40.3% ±0.2% compared to precision = 59.6% ±0.4% and recall = 34.3% ±0.4% for the overlap counting method, and we get precision = 61.1% ±0.3% and recall = 27.3% ±0.1% compared to precision = 52.9% ±0.5% and recall = 20.5% ±0.4% for the overlap depth method, for 2-fold cross validation on the yeast PPI network. In fact, we find that all DSD versions of the four algorithms perform better than their ordinary distance versions, regardless of the methods used for keeping track of overlaps. Figure 6 shows the improvements in F1 scores (setting $t=10$ and $c=1.5$ as before) over all three methods of counting the overlaps. Thus DSD is improving functional annotation also for GO annotation.

Discussion

Our function prediction results demonstrate the utility of defining and using a fine-grained distance measure that is specifically tailored to the subject application domain. For the PPI network, the observation that shortest-paths that go through hubs are less informative led to the particular design of the He () measure incorporated into the DSD definition. The He () measure has connections to other diffusion-based measures previously proposed [23]; however, using He () globally across all nodes, and comparing such vectors via L1 norm is entirely new and enables DSD to capture truly global properties of network topologies.

We have shown that substituting DSD for ordinary shortest path distance results in dramatic improvements when using network information to predict protein function for the *S. cerevisiae* PPI network. We expected that similar improvements would hold for the PPI-networks of other organisms as well, though the sparseness of current experimental known functional annotation and the number of PPI interactions currently known for other organisms means that we would not yet expect absolute levels of accuracy that are comparable to those achieved for *S. cervesiae*. We tested this intuition by also considering what is known of the PPI network of a different, less-well annotated yeast species, *S. pombe*. Based on the interactions in BIOGRID [20] version 3.2.102, the largest connected component has 1925 nodes and 4874 edges. Thus compared to the *S. cerevisiae* network, a much smaller and sparser subset of the PPI network is known. Figure 7 shows the shortest path and DSD distance distribution of this network. As expected, it is low-diameter but not yet as low-diameter as the *S. cerevisiae* network, and the DSD distribution is more spread out, but not yet as smooth as for the *S. cerevisiae* network. We would predict that the distribution would start looking more like *S. cerevisiae* as more of the network becomes known. We found no reliable MIPS annotation for *S. pombe*, so we instead used GO annotation, which we downloaded from the Pombase database, data version 2013-07-01 (http://www.pombase.org). We extract "biological process" GO terms only and remove GO terms annotated with evidence codes "IEA", "IPI" and "RCA", leaving 85 *informative* (see GO results section above, for definition) GO terms annotating 1722 out of 1925 proteins, with a total number of 5818 annotations. We did the "exact match" version of GO evaluation for 10 runs of cross-validation, and looked at the difference in performance for majority vote and functional flow methods using ordinary shortest path distance and DSD. We see similar improvements using DSD as with *S. cervisiae*; results appear in (Figure S4).

We present a simple to use, freely-available tool that given any PPI network will produce the DSD values between each pair of proteins, either as a webserver or for download from http://dsd.cs.tufts.edu/. This tool both allows the calculation of $DSD^{\{k\}}$ for some chosen k, and, as in the results presented in this paper, for DSD in the limit. In fact, we also tried $DSD^{\{5\}}$ in place of DSD for all the methods in this paper and results were already quite similar to DSD in the limit. We suggest based on the dramatic improvements for the *S. cerevisiae* PPI network that DSD values be used in place of ordinary distance when using network information to predict protein function, either using the PPI network alone, or as part of a modern integrative function prediction method that includes data from a variety of sources beyond the PPI network, such as sequence information, genetic interaction, structural information or expression data [8,13–15].

Acknowledgments

Crovella and Cowen thank the Institute for Mathematics and its Applications for inviting them to their "Network Links: Connecting Social, Communication, and Biological Network Analysis" workshop in March 2012, where we learned about each others' recent work and began the collaboration that resulted in this paper. We thank Andrew Gallant for helping review the DSD code.

Supporting Information

Figure S1 Improvement of mean Accuracy and F1 Score for DSD at different neighborhood thresholds for the χ^2 neighborhood algorithm in 10-runs of 2-fold cross validation (with standard deviation error bars).
(TIF)

Figure S2 Improvement of mean Accuracy and F1 Score for DSD at different neighborhood thresholds for the multi-way cut algorithm in 10-runs of 2-fold cross validation (with standard deviation error bars).
(TIF)

Figure S3 Improvement of mean Accuracy and F1 Score for DSD at different neighborhood thresholds for all four methods in 10-runs of 2-fold cross validation (with standard deviation error bars) using the GO catagories (using the method that gives wcredit only for exact matches to each GO term).
(TIF)

Figure S4 Performance evalutaion for GO term prediction on the *S. pombe* network. (a) Comparison of mean F1 score of DSD and non-DSD majority vote (setting t = 10) and functional flow (setting c = 1.5) algorithms using all three methods of counting GO term matches; (b1,b2) Comparison of mean accuracy and F1 Score under the exact match method for different settings of the parameters t and c, over 10 runs of 2-fold cross validation (with standard deviation error bars).
(TIF)

Author Contributions

Conceived and designed the experiments: MC HZ MEC LJC BH. Performed the experiments: MC HZ JP NMD. Analyzed the data: MC HZ LJC BH. Contributed reagents/materials/analysis tools: MC HZ NMD JP BH. Wrote the paper: MC MEC LJC BH.

References

1. Schwikowski B, Uetz P, Fields S (2000) A network of protein-protein interactions in yeast. Nature Biotechnology 18: 1257–1261.
2. Chua HN, Sung WK, Wong L (2006) Exploiting indirect neighbours and topological weight to predict protein function from protein–protein interactions. Bioinformatics 22: 1623–1630.
3. Hishigaki H, Nakai K, Ono T, Tanigami A, Takagi T (2001) Assessment of prediction accuracy of protein function from protein-protein interaction data. Yeast 18: 523–531.
4. Arnau V, Mars S, Marín I (2005) Iterative cluster analysis of protein interaction data. Bioinformatics 21: 364–378.
5. Bader G, Hogue C (2003) An automated method for finding molecular complexes in large protein interaction networks. BMC Bioinformatics 4.
6. Song J, Singh M (2009) How and when should interactome-derived clusters be used to predict functional modules and protein function? Bioinformatics 25: 3143–3150.
7. Nabieva E, Jim K, Agarwal A, Chazelle B, Singh M (2005) Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. Bioinformatics 21: 302–310.
8. Karaoz U, Murali T, Letovsky S, Zheng Y, Ding C, et al. (2003) Whole genome annotation by using evidence in functional-linkage networks. Proc Natl Acad Sci USA 101: 2888–2893.
9. Vazquez A, Flammini A, Maritan A, Vespignani A (2003) Global protein function prediction from protein-protein interaction networks. Nature Biotechnology 21: 696–700.
10. Sharan R, Ulitsky I, Shamir R (2007) Network-based prediction of protein function. Mol Syst Biol 3: 1–13.
11. Chiang T, Scholtens D, Sarkar D, Gentleman R, Huber W (2007) Coverage and error models of protein-protein interaction data by directed graph analysis. Genome Biology 8: R186.
12. Huang H, Bader JS (2009) Precision and recall estimates for two-hybrid screens. Bioinformatics 25: 372–378.
13. Lanckriet G, Deng M, Cristianini N, Jordan MI, Noble W (2004) Kernel-based data fusion and its application to protein function prediction in yeast. Pac Symp Biocomput: 300–311.
14. Tanay A, Sharan R, Kupiec M, Shamir R (2004) Revealing modularity and organization in the yeast molecular network by integrated analysis of highly heterogeneous genomewide data. Proc Natl Acad Sci USA 101: 2981–2986.
15. Wass MN, Barton G, Sternberg MJE (2012) Combfunc: predicting protein function using heterogeneous data sources. Nucleic Acids Res 40: W466–70.
16. Khanin R, Wit E (2006) How scale-free are biological networks? Journal of Computational Biology 13: 810–818.
17. Friedel CC, Zimmer R (2006) Toward the complete interactome. Nature Biotechnology 24: 614–615.
18. Han JDJ, Bertin N, Hao T, Goldberg DS, Berriz GF, et al. (2004) Evidence for dynamically organized modularity in the yeast protein-protein interaction network. Nature 430: 88–93.
19. Przulj N, Corneil D, Jurisica I (2004) Modeling interactome: scale-free or geometric? Bioinformatics 20: 3508–3515.
20. Stark C, Breitkreutz B, Reguly T, Boucher L, Breitkreutz A, et al. (2006) Biogrid: a general repository for interaction datasets. Nucleic Acids Res 34: D535–D539.
21. Ruepp A, Zollner A, Maier D, Albermann K, Hani J, et al. (2004) The funcat, a functional annotation scheme for systematic classification of proteins from whole genomes. Nucleic acids research 32: 5539–5545.
22. He X, Zhang J (2006) Toward a molecular understanding of pleiotropy. Genetics 173: 1885–1891.
23. Kohler S, Bauer S, Horn D, Robinson PN (2008) Walking the interactome for prioritization of candidate disease genes. Am J Hum Genet 82: 949–958.
24. Erten S, Bebek G, Koyutürk M (2011) Vavien: an algorithm for prioritizing candidate disease genes based on topological similarity of proteins in interaction networks. Journal of computational biology: a journal of computational molecular cell biology 18: 1561–1574.
25. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, et al. (2000) Gene ontology: tool for the unification of biology. Nature genetics 25: 25–29.
26. Darnell SJ, Page D, Mitchell JC (2007) An automated decision-tree approach to predicting protein interaction hot spots. Proteins: Structure, Function, and Bioinformatics 68: 813–823.
27. Deng M, Tu Z, Sun F, Chen T (2004) Mapping gene ontology to proteins based on protein–protein interaction data. Bioinformatics 20: 895–902.
28. Deng M, Sun F, Chen T (2003) Assessment of the reliability of protein-protein interactions and protein function prediction. Pacific Symposium on Biocomputing: 140–151.
29. Day-Richter J, Harris MA, Haendel M, Lewis S, et al. (2007) Obo-editan ontology editor for biologists. Bioinformatics 23: 2198–2200.