Network Querying [*]

Lecturer: Roded Sharan Scribers: Naama Amir and Arnon Mazza

May 28, 2013

1 Introduction

In recent years the amount of available data on protein-protein interaction (PPI) networks have increased rapidly, spanning different species such as yeast, bacteria, fly, worm and human. The rapid growth is shown in Figure 1. Besides availability of the data, other incentives to analyze several PPI networks at once include biological validation over several networks and prediction of unknown protein function and interactions.

A fundamental problem in molecular biology is the identification of cellular machinery, that is, protein pathways and complexes. PPI data present a valuable resource for this task. But there is a considerable challenge to interpret it due to the high noise levels in the data and the fact that no good models are available to pathways and complexes. Comparative analysis is used to tackle these problems, and improve the accuracy of the predictions. The main paradigm behind comparison of PPI networks is that evolutionary conservation implies functional significance. Conservation of protein subnetworks is measured both in terms of *protein sequence* similarity, and in terms of similarity in *interaction topology*.

In this scribe we focus on the network querying problem: Given a PPI network G of one species and a subnetwork S of another species, we wish to find subnetworks in G that are similar to S, in terms of both sequence similarity and topological similarity. In Section 3, we describe an algorithm that assumes that the query is simply a path. In Section 4 we show an algorithm that can handle tree queries and is extensible under certain conditions to general graph queries. Finally, in Section 5, we discuss the common scenario where the query topology is unknown, and show how to find a high confidence connected subnetwork that best matches the query proteins. We accompany the description of all algorithms by biological validations, demonstrating the power of network queries in identifying conserved modules across different species.

2 Preliminaries

As mentioned before, molecular interaction databases can be used to study the molecular pathways across species. One can look for similar patterns between a query that reflects some pathway in a PPI network of a well studied organism, and some, less studied, network of interest, thus discovering new pathways in the latter network. We proceed to a formal definition of the problem.

2.1 Problem Definition

Let $G = (V, E, w)$ be an undirected weighted graph, representing a PPI network, where $w : E \to R$ is a weight function representing interaction reliabilities. Let $G_Q = (V_Q, E_Q)$ denote a query graph with k vertices. Let $h(q, v)$ denote a similarity score between $q \in V_Q$ and $v \in V$, representing sequence homology.

[*]Based on scribes by Irit Levy, Oved Ourfali, Ofer Lavi and Lev Ferdinskoif (2005-6)

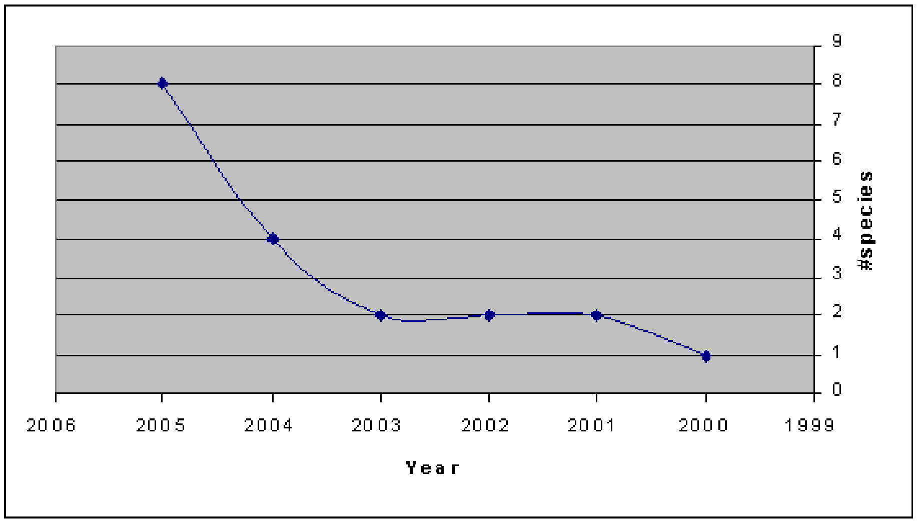


Figure 1: The amount of species for which large-scale PPI networks have been constructed. We can clearly see a rapid growth in the number of species since 2003.

A subdivision of an edge (u, v) in a graph $H = (U, F)$ replaces it with two edges (u, w) and (w, v), where $w \notin U$. H is *extendable* to a graph G if G can be obtained from H by a series of subdivisions.

An alignment of the query graph G_Q to G is defined as: (i) a subgraph $G_A = (V_A, E_A)$ of G, referred to as the alignment subgraph; (ii) a subgraph $S_Q = (V_Q^S, E_Q^S)$ of G_Q and a subgraph $S_A = (V_A^S, E_A^S)$ of G_A; (iii) a bijection $\sigma : V_Q^S \rightarrow V_A^S$ such that there is an edge $(u, v) \in E_Q^S$ if and only if there is an edge $(\sigma(u), \sigma(v)) \in E_A^S$. It is further required that S_Q is extensible to G_Q and S_A is extensible to G_A.

Intuitively, we look for a mapping between query nodes (allowing deletions) and graph vertices (allowing insertions) that is interaction-preserving.

The weight of an alignment is the sum of (i) similarity scores of aligned vertices, (ii) weights of edges in the aligned subgraph, (iii) a (negative) penalty score for each node deletion or insertion.

The graph query problem is therefore defined as follows: Given G_Q, G, h and penalty scores for insertions and deletions, find an alignment of G_Q in G with maximal weight.

2.2 Problem Complexity

The problem is in general equivalent to subgraph isomorphism, which is computationally hard [4]. A naive algorithm can solve this problem in $O(n^k)$. Reduction in complexity can be achieved using fixed parameter algorithms by constraining the query. The algorithms that will be described here combine a DP approach and the color coding method presented in [1]. The idea is to randomly assign k colors to the vertices of the graph and then search for colorful alignments in which each color is used exactly once.

Using dynamic programming, in each step the algorithms extend optimal sub-alignments. Adding a graph vertex is allowed only if it is not already in the alignment. Naively, each potential sub-optimal alignment should maintain the list of at most k vertices already matched, resulting in $O(n^k)$ possibilities. Using color coding, it is sufficient to remember the used colors rather than the vertices, resulting in $O(2^k)$ possibilities, which significantly reduces the computation time. However, the algorithm returns a correct answer only if the optimum alignment is colorful, which happens with probability $\frac{k!}{k^k} \simeq e^{-k}$. Therefore, if we repeat the algorithm $ln(\frac{1}{\epsilon})e^k$ times, we get the optimum alignment with probability at least $1 - \epsilon$ for any desired value of ϵ.

3 Path Queries

Sequence comparison is a basic tool in biological research, widely used for nucleotide sequences comparison and search (as in RNA and DNA molecules), and for amino acids sequence comparisons (as in protein homology discovery). It is used both for evolutionary and functional research of both genes and proteins. The availability of PPI networks allows us to extend the use of sequence comparison methods to more complex functional units, such as protein pathways and modules, and thus elevate homology detection from the level of single protein homology to the level of functional protein pathways and modules homology.

This section describes a method for path querying, called QPath ([6]), which overcomes some fundamental drawbacks of the PathBLAST algorithm ([5]):

1. In a PathBLAST result, a matched pathway may contain the same protein more than once, which is biologically implausible.
2. The resulting matched pathways must be very close to each other, while we might want to allow a higher degree of freedom, and support more than a single consecutive insertion or a single consecutive deletion difference between the paths, which is the maximum PathBLAST allows.
3. The running time of the algorithm involves a factorial function of the pathway length, limiting its applicability to short pathways (in practice, it was applied to paths of up to 5 proteins).

3.1 The path query problem

The problem setting is defined as follows: the input is a target network, represented as an undirected weighted graph $G = (V, E)$ with a weight function on the edges $w : E \to R$, and a path query $Q = (q_1, \ldots, q_k)$. Additionally, a scoring function $H : Q \times V \to R$ is given. The output is a set of best matching pathways $P = (p_1, \ldots, p_k)$ in G, where a good match is measured in two aspects:

1. Each node in the matched pathway and its corresponding node in the query are similar with respect to the given scoring function H.
2. The reliability of edges in the matched pathway is high.

If we don't force the size of the query and matching paths to be equal, we can still measure the match between a query $Q = (q_1, \ldots, q_k)$ and a pathway $P = (p_1, \ldots, p_l)$ by introducing *dummy nodes* which allow for deletions, if inserted in the matching path and for insertions, if inserted in the query.

In this framework, the target graph is a PPI network of species 1, where vertices are proteins and edge weights represent interaction probabilities between two proteins. The query pathway Q is a pathway extracted from a PPI network of species 2, and the function H is a similarity measure between proteins in the two species.

3.2 The QPath algorithm

First, in order to allow more flexibility in deletions and insertions, deletions of nodes in the target network are allowed by introducing a mapping M form Q to $P \cup \{0\}$ where deleted query nodes are mapped to 0 by M. The total score of an alignment reflects the measures of protein homology and the interaction

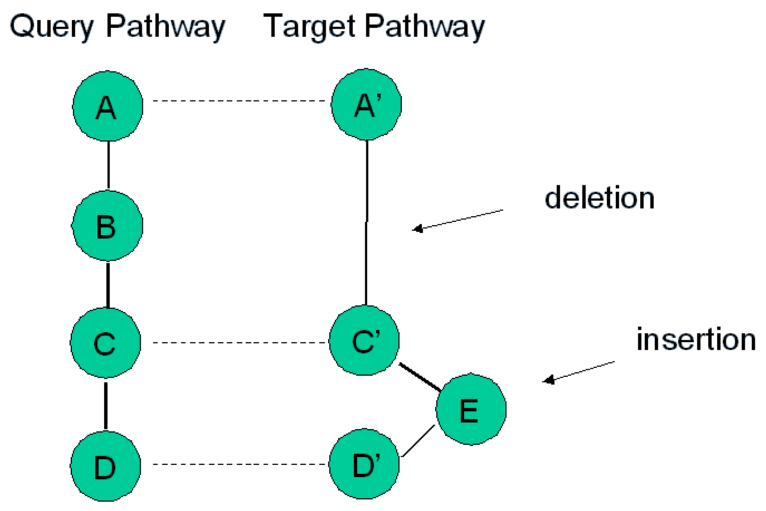


Figure 2: An example of an alignment that induces insertions (F') and deletions (C).

probabilities of the path, while keeping the path similarity with a certain degree of freedom for insertions and deletions, and is set to be:

$$\sum_{i=1}^{l-1} w(p_i, p_{i+1}) + \sum_{i=1, p_i \neq 0}^{k} h(q_i, M(p_i))$$

Where the first summation is the *interaction score* and the second is the *sequence score*. Edge weights represent the logarithm of reliability of interaction between two proteins, and the protein similarity scoring function H is set to be the BLAST E-value for the two proteins, normalized by the maximal E-value over all pairs of proteins from the two networks.

3.3 Avoiding cycles (non-trivial paths)

In order to find only simple paths, QPath uses the color coding technique (Alon et al. [1]). The method allows finding simple paths of size k by randomly choosing a color out of k colors for every vertex in the graph, and looking only for subgraphs that do not contain more one vertex of the same color. Since a particular path may be assigned non-distinct color, the method requires choosing many random colorings, and running the search for each of them separately.

3.4 Finding the best matching paths

QPath sets in advance two parameters - N_{ins} and N_{del}, which are the number of insertions and deletion allowed in the matched path. When looking for a path of size k, QPath assigns $k + N_{ins}$ colors for the vertices.

The following dynamic programming recursion is then used for dynamically building the best path:

$$W(i,j,S,\Theta_{del}) = max_{m \in V} \begin{cases} W(i-1,m,S-c(j),\theta_{del}) + w(m,j) + h(q_i,j) & (m,j) \in E \\ W(i,m,S-c(j),\theta_{del}) + w(m,j), & (m,j) \in E \\ W(i-1,m,S,\theta_{del}-1), & \theta_{del} < N_{del} \end{cases}$$

$W(i, j, S, \Theta_{del})$ is the maximum weight of an alignment for the first i nodes in the query that ends at vertex $j \in V$, induces θ_{del} deletions, and visits a vertex of each color in S. The first case is the case where q_i is aligned with vertex j, and thus we add to the best alignment so far the score $h(q_i, j)$, and remove the color of j from the set of available colors S. In the second case, q_i is not aligned with j, meaning q_i is an insertion, and the score does not change. The third case is a deletion case, and therefore we decrease the number of allowed deletions from this point on by one.

The best alignment score will be $max_{j \in V, S \subseteq C, \theta_{del} < N_{del}} W(k, j, S, \theta)$, and the alignment itself can be find by backtracking. The running time for each coloring choice is $2^{O(k+N_{ins})mN_{del}}$. For a choice of $\varepsilon \in (0, 1)$ such that the probability to find the optimal match is at least $1 - \varepsilon$ we would need to choose $ln(n/\varepsilon)$ random colorings, which will give a total running time of $ln(n/\varepsilon)2^{O(k+N_{ins})mN_{del}}$.

In order to use QPath for searching homologous paths between two given PPI networks, it is first required to extract good candidates from the first network, and then search for these paths in the target network. QPath can find good candidates by searching the first network for a dummy path query, consisting of dummy proteins that have the same similarity score H to all vertices in the network. Such a search yields pathways with high interaction scores in the first network, regardless of the path query itself.

3.5 Running QPath on yeast and fly PPI networks

The yeast (S. cerevisiae) PPI network contains 4,726 proteins and 15,166 known interaction between them. The fly (D. melanogaster) PPI network contains 7,028 proteins and 22,837 interactions, but in spite of its larger size, it is much less complete than the yeast network.

The algorithm was tested first on the more complete yeast PPI network, finding good candidates for querying the fly PPI network next. It discovered 271 pathways which were better than 99% of randomly chosen pathways obtained by setting all interaction scores to be equal and running the query on the tweaked data. The 271 pathways were then used as queries for the fly PPI network.

The results of running the algorithm on the yeast PPI network were assessed by looking at the functional enrichment of the found paths. 80% of the paths found were functional enriched, implying their biological significance. In comparison, running dummy queries on the less complete fly network resulted with only 39% of the 132 fly paths found to be functionally enriched.

Running the 271 paths found in the yeast PPI network as queries on the fly network discovered that 63% of them had a match in the fly network (Figure 3).

The results show that pathway similarity can be used for identification of functionally significant pathways, and that those query pathways can help us to infer the actual function of matched pathways. a first annotation map of protein pathways in fly that are conserved from yeast was obtained this way by QPath.

3.6 Scoring the paths

After setting the scoring framework, there is a need to set the weighs parameters, and define the actual contribution of the different scoring components - the *interaction score*, the *sequence score* and the cost of insertions and deletions.

The target is to find a weight function that will maximize the probability that a path with high score is indeed functionally enriched. This was done by using logistic regression on the path attributes - interactions reliability, sequences similarity, number of insertions, and number of deletions, using known functionally enriched paths in the yeast network for training.

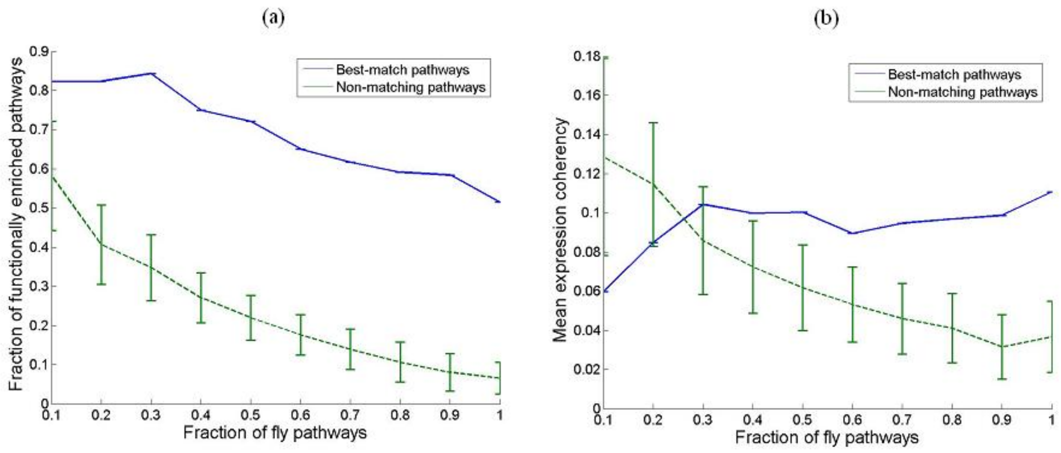


Figure 3: Source [6]. Functional significance of best-match pathways in fly. Functional enrichment (a) and expression coherency (b) of fly best-match pathways obtained by QPath compared to fly pathways that are not the result of a query

3.7 Is the insertion and deletion flexibility really required?

As mentioned before, one of the most important features QPath introduced is the ability to align sequences with a high number of subsequent insertions or deletions. Figure 4 illustrates that this feature is indeed important, as most of the conserved paths between the yeast and the fly, required more than one insertion and deletion.

In the same manner, discovering functionally enriched paths was also found to be strongly depended on the fact that a high number of insertions and deletions is required (See Figure 5)

3.8 Functional conservation

Results of running QPath on the yeast and fly PPI networks, yielded that for 64% of the conserved paths, the matched paths in the fly network conserved one or more functions of the yeast query pathways. In

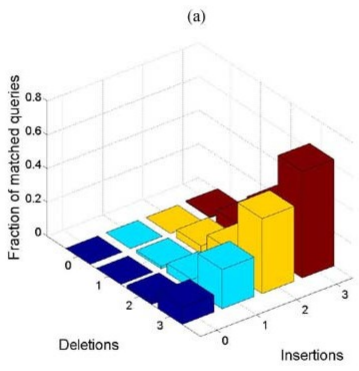


Figure 4: Source [6]. Fraction of matched queries between yeast and fly networks in respect to the number of deletions and insertions in the conserved paths

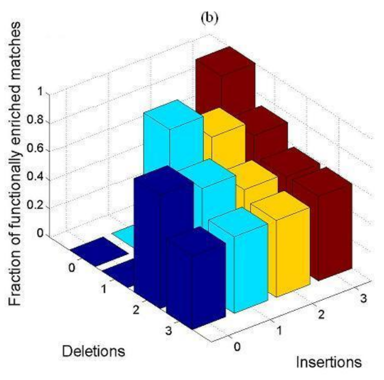


Figure 5: Source [6]. Fraction of functionally enriched matches in respect to the number of deletions and insertions in the conserved paths

contrast, a random shuffling of the matches was tested and resulted to in conservation rate of only 31%. Interestingly, the functional conservation was much lower when limiting the protein homology only to the best pairs, one from each species. This implies that pathway homology can be used to predict function. More explicit methods for such a prediction, that make use of the networks homology on top of the straight forward sequence alignment will be presented in the next section.

4 Graph Queries

Here we present the QNet algorithm for Querying in Molecular interaction databases. The algorithm solves the subgraph isomorphism problem by constraining the graph queries to trees, achieving a fixed parameter algorithm. Also it combines a DP approach and the color coding method [3].

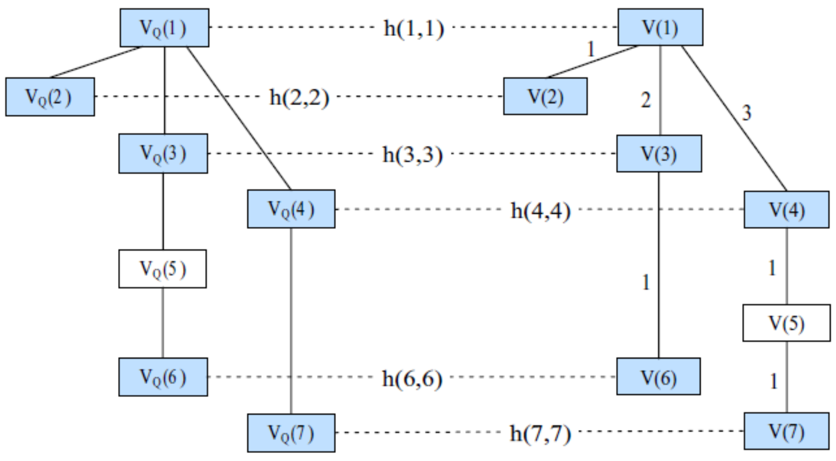
4.1 The QNet Algorithm

For each query node q with n_q children denote its children by $q_1, ..., q_{n_q}$. The alignment is built gradually covering the subtrees rooted by the first j children of $q : q_1, ..., q_j$.

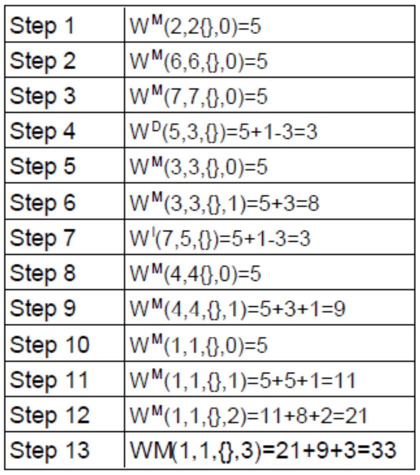
Let $W^M(q, v, S, j)$ denote the maximal score of an alignment between (i) the query subtree that is rooted at node q and contains its first j children along with their subtrees, and (ii) the graph subtree rooted at vertex v and uses the color set S.

$$W^M(q,v,S,j) = \max_{\substack{u:(u,v)\in E \\ S'\subset S}} \begin{cases} (*Match\ child\ j*) \\ W^M(q,v,S',j-1) + W^M(q_j,u,S-S',n_{q_j}) + w(u,v) \\ \\ (*Insertion\ vertex\ u*) \\ W^M(q,v,S',j-1) + W^I(q_j,u,S-S') + w(u,v) \\ \\ (*Deletion\ child\ j*) \\ W^M(q,v,S',j-1) + W^D(q_j,v,S-S') \end{cases}$$

$W^I(q, v, S)$ denotes the optimal score of an alignment of the query subtree that is rooted at q and the graph subtree that is rooted at v such that q is aligned with some vertex u that is a descendant of v. In this case, v is not aligned to any node in the query, in other words v is inserted.



(a) Query Graph (b) Alignment subgraph



Step 1	$W^M(2,2\{\},0)=5$
Step 2	$W^M(6,6,\{\},0)=5$
Step 3	$W^M(7,7,\{\},0)=5$
Step 4	$W^D(5,3,\{\})=5+1-3=3$
Step 5	$W^M(3,3,\{\},0)=5$
Step 6	$W^M(3,3,\{\},1)=5+3=8$
Step 7	$W^I(7,5,\{\})=5+1-3=3$
Step 8	$W^M(4,4\{\},0)=5$
Step 9	$W^M(4,4,\{\},1)=5+3+1=9$
Step 10	$W^M(1,1,\{\},0)=5$
Step 11	$W^M(1,1,\{\},1)=5+5+1=11$
Step 12	$W^M(1,1,\{\},2)=11+8+2=21$
Step 13	WM(1,1,{},3)=21+9+3=33

(c) Dynamic Programming steps

Figure 6: Execution example of the QNet DP algorithm.

$$W^I(q,v,S) = \max_{u\,:\,(u,v)\in E} \begin{cases} W^M(q,u,S-\{c(v)\},n_q) + w(u,v) + \delta_i \\ W^I(q,u,S-\{c(v)\}) + w(u,v) + \delta_i \end{cases}$$

$W^D(q,v,S)$ denotes the optimal score of the alignment of the query subtree that is rooted at q and the graph subtree that is rooted at v, such that q is deleted and v is aligned with an ancestor of q.

$$W^D(q,v,S) = \max_{u\,:\,(u,v)\in E} \begin{cases} W^M(q_1,u,S,n_{q1}) + w(u,v) + \delta_d \\ W^I(q_1,u,S) + w(u,v) + \delta_d \\ W^D(q_1,v,S) + \delta_d \end{cases}$$

Deletions are allowed for query nodes of degree two. Inserted vertices in the alignment subgraph must also have degree two.

The recursion is initialized by setting $W^M(q,v,S,0) = h(q,v)$ for each $q \in V_Q$ and $v \in V$.

The maximal score of the alignment is $\max_{(v,S)} W^M(r,v,S,n_r)$, where r is some root of the query graph with degree 1. An application of the DP recursions to a sample query is demonstrated in Figure 6.

The running time of each trial is $2^{O(k+N_{ins})}|E|$, where N_{ins} is the maximal number of insertions allowed. The probability of receiving distinct colors for the vertices of the optimal matching tree is at least $e^{-k-N_{ins}}$. Thus, the running time of the algorithm is $2^{O(k+N_{ins})}|E|ln(\frac{1}{\epsilon})$, for any desired success probability $1-\epsilon$. Therefore, the algorithm is tractable for realistic values of $|E|$ and k. (For example, the running time for a graph with $|V| \sim 5000$ and query tree with $k=9$ nodes was 11 seconds).

4.2 General graph queries by consensus match construction

It is possible to use the QNet algorithm to execute general graph queries. Given a query graph Q, one can extract several spanning trees of Q and execute the QNet algorithm on each of the trees. The resulting matches can be used to construct a consensus match, consisting of all proteins that appeared in at least half of the matches.

4.3 Bounded tree-width graph queries

In this section we describe in high-level a graph query algorithm, proposed by [3], that is not restricted to tree queries. We first define the notions of *tree decomposition* and *tree-width* and then proceed to describe the intuition of the algorithm.

Tree decomposition. The tree decomposition of a graph $G = (V, E)$ is a pair $T = (X, F)$, where $X = X_1, ..., X_s$ is tree whose nodes represent subsets of V (named *tree nodes* or *super nodes*), satisfying the following properties:

1. The union of all sets X_i covers V.

2. For every edge $(v, w) \in E$, there is a subset X_i that contains both v and w.

3. If X_i and X_j both contain a vertex v, then all nodes X_k of the tree in the (unique) path between X_i and X_j contain v as well. That is, the tree nodes associated with each vertex v form a connected subtree of T.

   This requirement can also be stated as follows: if X_i, X_j, X_k are tree nodes and X_k is on the path between X_i and X_j, then $X_i \cap X_j \subseteq X_k$.

The *width* of a tree decomposition is the size of its largest set X_i minus one. The *tree-width* of a graph G is the minimum width among all possible tree decompositions of G. Note that by this definition, the tree-width of a tree is equal to one.

The tree decomposition of a graph is not unique. For example, a trivial tree decomposition of any graph G contains all vertices of G as a single tree node.

Lemma. If W is a clique in a graph G then in every tree decomposition of G there is a tree node X_i such that $W \subseteq X_i$. Thus, the tree-width of a clique of size n is $n - 1$.

Proof. Let T be some tree decomposition of G. For every $w \in W$ let i_w be the tree node with minimum height in T that contains w and let r_w be this height. Examine the vertex w^* such that r_{w^*} is maximal. Let z be some node in W other than w^*. We prove by contradiction that $z \in i_{w^*}$. Since there is an edge (z, w^*), there must be some tree node i that contains both z and w^*. Since w^* is not contained by any tree node above i_{w^*}, i must be a descendant of i_{w^*}, or else the connectivity constraint for w^* would be violated. But from $z \in i$, $z \notin i_{w*}$ and $r_z \leq r_{w*}$, connectivity is violated for z, a contradiction.

Intuition of the algorithm. In order to solve the graph query problem for the general case, given a query graph Q, a tree decomposition T of Q is computed, and then the QNet algorithm is executed using T as the query. This requires adaptation of the algorithm to address two main challenges:

1. A super-node X that is composed of t query nodes may have any topology, requiring an exhaustive search of $O(n^t)$ time to find its matches in the input graph. This is addressed by limiting the tree-width that the algorithm can handle.

2. A query node may appear in multiple super-nodes, requiring the algorithm to be consistent when different super-nodes are aligned. The key idea is to introduce a new parameter σ to the DP formula, denoting an assignment from query nodes to vertices in the input graph. The best alignment of a super-node X in the tree is then computed over all possible assignments for X that are consistent with the assignments of its children.

4.4 Results

In order to estimate the accuracy and quality of the QNet algorithm, two types of tests were performed:

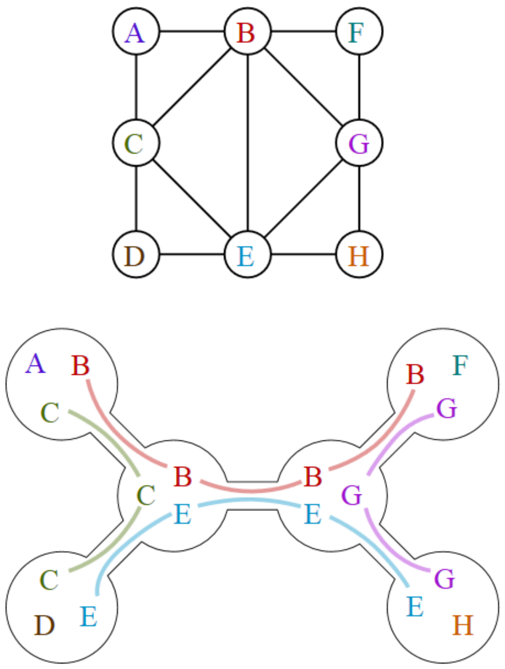


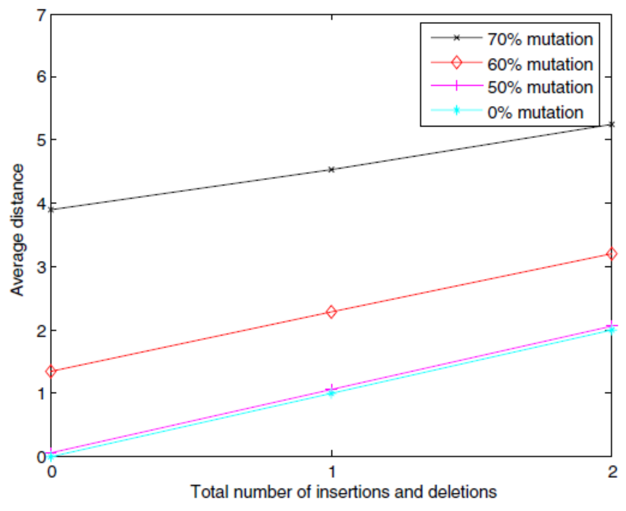
Figure 7: A graph with eight vertices, and a tree decomposition of it onto a tree with six nodes. Note how the occurrences of every graph node form a connected subtree.

Synthetic query trees. In the first test, QNet was used to query the PPI network of yeast with a set of synthetic query trees. This set consisted of 20 randomly chosen subtrees of sizes ranging from $k = 5$ to $k = 9$ from the yeast PPI network. Each query tree was perturbed with up to 2 node insertions and deletions, and by a pre-specified amount of point mutations in protein sequences. QNet was applied to identify a match for each query tree.

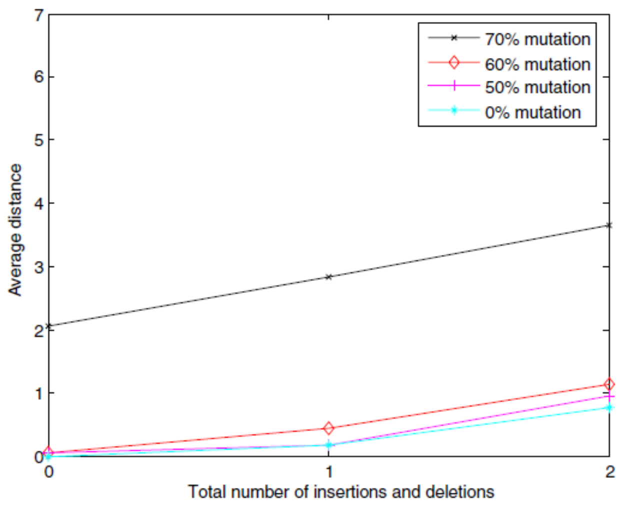
The accuracy of a matched tree was evaluated by the symmetric difference between the protein set of the query and the protein set of its match (termed *distance*). The results show that when perturbing protein sequences in up to 60% of the residues, the average distance is lower than 1 (Figure 8b). As illustrated in Figure 8a, these results are significantly more accurate than those achieved by a sequence-only based method (BLAST), showing that the topology of the query tree carries important signal. The advantage of QNet over a sequence-based approach becomes more pronounced when the mutation rate increases.

Cross-species comparison of protein complexes. As a second validation test, QNet was applied to query known yeast protein complexes from the MIPS database within the fly network. Overall, 94 complexes were considered, each consisting of at least 4 proteins. The query subnetworks for each complex were formed by extracting an average of 40 random trees of size $3 - 8$ from the yeast network that is induced by the proteins in the complex. The resulting match of a query complex was achieved by applying QNet to each of the trees and then constructing a consensus match, as described in Section 4.2.

The biological plausibility of an obtained consensus match was tested based on functional enrichment of its member proteins with respect to the fly GO process annotation. The significance of the enrichment was computed based on a combination of hypergeometric distribution, random set analysis and false discovery rate correction. Of the 94 yeast complexes, 36 resulted in a consensus match with more than one protein in fly. Of these, 72% were significantly enriched ($p < 0.05$). For comparison, randomly chosen trees from the



(a)



(b)

Figure 8: The average distance of the matched tree from the original tree is plotted against the total number of insertions and deletions introduced to the query for 4 different mutation levels. (a) Performance of a sequence-based approach. (b) Performance of QNet.

fly PPI network that have the same distribution of sizes and interaction scores as the consensus matches were examined. Only 17% of the random trees were functionally enriched; also, the mean enrichment p-value was much lower for the true consensus matches (Wilcoxon rank test p-value $< 6.5 \times 10^{-9}$).

Figure 9 illustrates the result of querying the Cdc28p complex. This complex is composed of cyclin-dependent kinases involved in regulating the cell cycle in yeast. The consensus match obtained in fly consists solely of cyclin-dependent kinases and significantly overlaps the cyclin-dependent protein kinase holoenzyme complex (GO:0000307).

5 Topology-free queries

In the previous algorithms discussed here, given a protein complex or pathway of species A and a PPI network of species B, the goal was to identify subnetworks of B that are similar to the query in terms of sequence, topology, or both. In practice, the topology of the query is often not known. The Topology-Free Querying of Protein Interaction Networks (TORQUE) suggested by [2] ignores the topology of A. Given a query, represented as a set of proteins, it seeks a matching set of proteins that are sequence-similar to the query proteins and span a connected region of the network.

5.1 The TORQUE algorithm

The input to this method is a set of proteins, representing a complex or pathway of interest and a network in which the search is to be conducted. The goal is to find matching sets of proteins that span connected regions in the network. The corresponding theoretical problem is searching a colored graph for connected subgraphs whose vertices have distinct given colors. The method uses alternatively dynamic programming

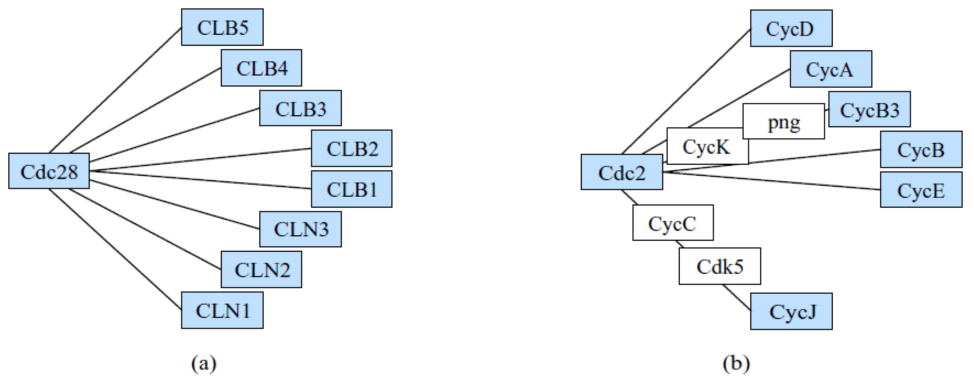


Figure 9: (a) The yeast Cdc28p complex. (b) The consensus match in fly. Matched nodes appear on the same horizontal line. Inserted proteins appear in white.

and integer linear programming for the task. The fixed-parameter dynamic programming (DP) algorithm utilizes the color coding paradigm [1].

Preprocessing. Each query protein is colored differently. Each vertex in the target network is associated with a subset of colors corresponding to the query proteins it is sequence-similar to. The goal is to find the best match such that for each target protein a different query protein is associated to and all target proteins are connected, i.e choosing for each target vertex a different color from its subset of colors and finding a colorful subgraph. An example is shown in figure 10.

Problem Definition. Let k be the number of proteins in the query. Let $G = (V, E)$ be a PPI network where vertices represent proteins and edges correspond to PPIs. For a vertex v, let $N(v)$ denote the set of its neighbors. For two disjoint sets S_1 and S_2, we write $S_1 \uplus S_2$ for their union $S1 \cup S2$. We denote by $G[K]$ the subgraph of G induced by the vertex set K. Given a set of colors $C = 1, 2, .., k$, a coloring constraint function $\Gamma : V \rightarrow 2^C$ associates with each $v \in V$ a subset of colors $\Gamma(v) \subseteq C$. This subset of colors represents the subset of proteins in the query that are similar to the protein represented by v.

Given a graph $G = (V, E)$, a color set C, and a coloring constraint function $\Gamma : V \rightarrow 2^C$, the goal is to find a connected subgraph of G that is C-colorful.

Dynamic programming approach. In this section we show how to solve the problem using a randomized DP approach. This approach considers only coloring constraint functions that associate each $v \in V$ with a single color. In this case, the input is a graph where each vertex is assigned a color from C, and we aim to find a connected subgraph having exactly one vertex of each color. First, note that every connected subgraph has a spanning tree. Therefore, every colorful connected subgraph will have a colorful spanning tree. Thus, instead of looking for a colorful subgraph we look for a colorful tree.

We show here the weighted case, where each edge is assigned a weight, and the heaviest tree is sought. The algorithm finds, for each vertex, the heaviest colorful subtree rooted at it.

Define $T(v, S)$ as the weight of the heaviest tree that is (i) rooted at v and (ii) S-colorful.

$$T(v, S) = \max_{u \in N(v)\ ,S_1 \uplus S_2 = S\ ,\Gamma(v) \in S_1, \Gamma(u) \in S_2} T(v, S_1) + T(u, S_2) + w(u, v)$$

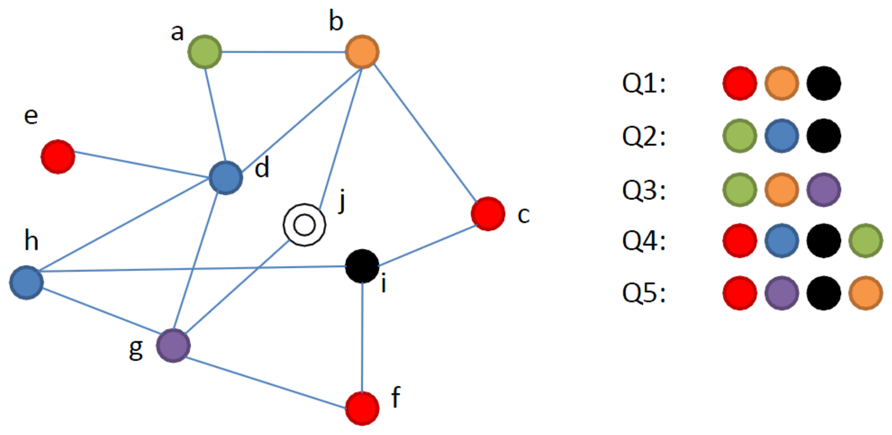


Figure 10: Topology free quering example. Left: the network. Right: queries. Q1 is solved by $\{c, b, i\}$.

Initialization: $T(v, \gamma) = 0$ iff $\Gamma(v) = \gamma$, $T(v, \gamma) = -\infty$ otherwise.
The weight of the optimum match is given by $\max_v T(v, C)$. The algorithm runs in $O(3^k|E|)$ time.

A color coding based solution exists also for the case that each vertex has an associated set of colors instead of one. This is solved in practice by randomly choosing a single color for each vertex and bounding the expected number of iterations needed (at most $k!$ but often less than 100).

ILP approach. Another formulation of this problem uses ILP and is generally faster than the DP algorithm. The main challenge is the formulation of the connectivity requirement. This is solved by simulating a flow system over the proposed match, choosing one node as a sink, $k-1$ nodes as sources of one unit of flow, and testing whether the sink drains $k-1$ units of flow,while disallowing flow between non-selected vertices.

5.2 Results

In order to assess the biological contribution of TORQUE, the algorithm was applied to query protein complexes within the PPI networks of yeast (5430 proteins, 39936 interactions), human (7915 proteins, 28972 interactions) and fly (6650 proteins, 21275 interactions). Two types of query complexes were selected: complexes with known topology (yeast, human, fly) and complexes with unknown topology (mouse, rat, bovine). The complexes were collected from publicly available databases (yeast - from the SGD project, fly - from the GO category ”protein complex” annotation, and all mammals - from the CORUM website). The first type of queries served to validate the algorithm and compare it to the QNet algorithm, whereas the second type was used to explore the usefulness of the algorithm when query topology is not available.

To evaluate the quality of the matches, two measures were used:

- **Functional coherence**: This measure reports the percent of matches that are significantly functionally coherent with respect to the GO annotation. (Note that while the query is functionally coherent, the reported matches may not be so due to permissive homology matching and noise in the PPI data.)

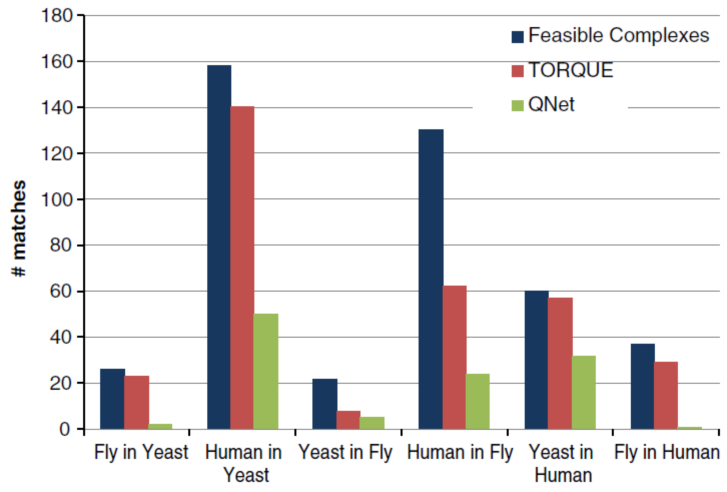


Figure 11: Comparison of number of matches for Torque and QNet.

The functional coherence of a match, represented as a set of proteins, was computed using the GO TermFinder tool.

- **Specificity**: This measure reports the fraction of matches that significantly overlap with a known protein complex. The significance of the overlap was evaluated by combining the hypergeometric distribution and an empirical p-value obtained by looking for protein complexes in random sets of the same size. The specificity computation was applied to the matches that had a non-zero overlap with the collection of complexes to which they were compared. As shown in Figure 12, a high percentage of the matches were specific. This suggests that matches that had no overlap with known complexes are candidates to be novel complexes.

A comparison of the number of detected matches by both the TORQUE and QNet algorithms is depicted in Figure 11. (Protein complex queries were executed in QNet using the consensus match method described in Section 4.2).

To analyze the quality of the results, the functional coherence and specificity measures were computed for the set of identified matches. A match was considered ”functionally coherent” or ”specific” based on a significance threshold of 0.05. These results are summarized in Figure 12.

Figure 13 summarizes the results of the second type of queries, that involves complexes with unknown topologies.

References

[1] N. Alon, R. Yuster, and U. Zwick. Color–coding. *J. ACM*, 42(4):844–856, 1995.

[2] S. Bruckner, F. Huffner, R.M. Karp, R. Shamir, and R. Sharan. Topology-free querying of protein interaction networks. *Journal of Computational Biology*, 17:237252, 2010.

[3] B. Dost, T. Shlomi, N. Gupta, E. Ruppin, V. Bafna, and R. Sharan. Qnet: A tool for querying protein interaction networks. *Journal of Computational Biology*, 15:913925, 2008.

		Functional coherence		*Specificity*	
Network	*Complex*	TORQUE	*QNet*	TORQUE	*QNet*
Yeast	Fly	23 (100%)	2 (100%)	19 (82%)	2 (100%)
	Human	134 (95%)	49 (98%)	119 (85%)	47 (94%)
Fly	Yeast	8 (100%)	3 (60%)	8 (100%)	4 (80%)
	Human	56 (90%)	21 (87%)	62 (100%)	23 (95%)
Human	Yeast	48 (84%)	25 (78%)	43 (75%)	23 (71%)
	Fly	21 (72%)	0 (—)	21 (72%)	0 (—)
Total		290	100	272	99

Figure 12: Comparison of the functional coherence and specificity measures between the matches captured by QNet and TORQUE.

Network	*Complex*	*No. feasible*	*No. matches*	*Functional coherence*	*Specificity*
Yeast	Bovine	4	4	4	4
	Mouse	17	17	16	13
	Rat	23	20	19	9
Fly	Bovine	3	0	—	—
	Mouse	14	7	0	1
	Rat	34	21	17	7
Human	Bovine	4	4	2	1
	Mouse	48	46	32	24
	Rat	44	43	32	24
Total		191	162	122	83

Figure 13: Statistics of querying of protein complexes with unknown topology using TORQUE.

[4] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman & Co., 1979.

[5] B.P. Kelley, B. Yuan, F. Lewitter, R. Sharan, B.R. Stockwell, and T. Ideker. Pathblast: a tool for alignment of protein interaction networks. *Nucleic Acids Res.*, 32, 2004.

[6] T. Shlomi, D. Segal, E. Ruppin, and R. Sharan. Qpath: a method for querying pathways in a protein–protein interaction network. *BMC Bioinformatics*, 7:199–208, 2006.