Identification and prediction of functional protein modules using a bi-level community detection algorithm

Suely Oliveira and Rahil Sharma*

Department of Computer Science,
The University of Iowa,
Iowa City, IA 52242, USA
Email: suely-oliveira@uiowa.edu
Email: rahil-sharma@uiowa.edu
*Corresponding author

Abstract: Identifying functional modules is believed to reveal most cellular processes. There have been many computational approaches to investigate the underlying biological structures. We shall use community detection algorithm which we present in a bi-level algorithmic framework to accurately identify protein complexes in less computational time. We call this algorithm bi-level label propagation algorithm (BLLP). Using this algorithm, we extract 123 communities from a protein–protein interaction (PPI) network involving 2361 proteins and 7182 interactions in *Saccharomyces cerevisiae* i.e. yeast. Based on these communities found, we make predictions of functional modules for 57 uncharacterised proteins in our dataset, with 80%+ accuracy. We also perform a comparative study by applying various well-known community detection algorithms on the PPI yeast network. We conclude that, BLLP algorithm extracts more accurate community structures from PPI yeast networks in less computational time.

Keywords: community detection; label propagation; bi-level algorithms; computational proteomics.

Reference to this paper should be made as follows: Oliveira, S. and Sharma, R. (2016) 'Identification and prediction of functional protein modules using a bi-level community detection algorithm', *Int. J. Bioinformatics Research and Applications*, Vol. 12, No. 2, pp.129–148.

Biographical notes: Suely Oliveira is a Professor in the Computer Science Department at the University of Iowa. She received her PhD from the University of Colorado in 1993. She was at the Australian National University from 1993 to 1994, and at Texas A&M University from August 1994 to May 1998. She has published over 50 articles in the areas of numerical analysis, parallel algorithms, scientific computing, combinatorial scientific computing, and is co-author of book *Writing Scientific Software* published by Cambridge Press University and *Building Proofs: A Practical Guide* published by World Scientific.

Rahil Sharma received his BE degree in Computer Engineering from University of Mumbai and MS degree in Computer Science from University of Iowa, USA. He is currently working towards his PhD degree in Computer Science at the University of Iowa. His research interests include areas of algorithm development, high performance parallel computing, and big data analytics.

1 Introduction

Most cellular processes are believed to be carried out by groups of highly interacting proteins called functional modules, protein complexes, or molecular complexes. Recent large-scale high-throughput experiments, and integration of published data, have generated large protein–protein interaction (PPI) networks. Even one of the simplest eukaryotic organism, yeast, has more than 5000 proteins. Protein complexes can be detected by identifying highly connected sets of proteins in PPI networks. Computational identification of functional modules or protein complexes can provide an inexpensive guideline for biological experiments. Protein–protein interactions can alter kinetic properties of enzymes, create new binding sites for small effector molecules, destroy or inactivate the protein, exhibit a new functionality which a single protein cannot exhibit alone, etc.

There have been many recent computational approaches to disclose the underlying biological structures (Bader and Hogue, 2003; Ding et al., 2006; Krogan, 2006; Ramadan et al., 2005; Xiong et al., 2005; Zhang et al., 2004). These approaches are divided into two groups. One group uses machine learning approaches to construct weighted networks by integrating existing data sets to predict protein complexes (Krogan, 2006; Zhang et al., 2004). Another group tries to extract highly connected subgraphs or dividing a whole network into groups of clusters on a protein–protein interaction (PPI) network (Bader and Hogue, 2003; Ding et al., 2006; Ramadan et al., 2005; Xiong et al., 2005; Oliveira and Seok, 2008).

There are a number of challenges in treating protein–protein interaction data. One is that many high-throughput experiments have high error rates, which results in a great many false positives for interactions between proteins. Another challenge is that some proteins are interaction mediating proteins that interact with very large numbers of other proteins; these might, for example, provide common services to many different parts of the cell. The former challenge can make accurate identification of functional modules difficult, while the latter challenge tends to make the entire proteome appear to be a single indivisible functional module.

Protein complexes correspond to modules, which can be viewed as dense sub-networks or communities in PPI networks. Community detection algorithms can be used extract these dense sub-networks/communities. Modules can be defined in many ways with, for example, densely connected sub-networks with more intra-node edges as compared to inter-node edges. Some of these definitions are present in the literature (Radicchi et al., 2004). Functional module detection in PPI networks is computationally very hard. Based on different definitions of modules, there are many community detection algorithms in the current state of the art, which is used to identify functional modules in PPI networks. Trivial algorithms are not well suited for this job (Yook et al., 2004).

One class of the community detection technique relies on finding completely connected sub-networks (*cliques*) in PPI network (Spirin and Mirny, 2003). This technique is very inefficient for detecting modules for large PPI networks, and also proteins participating in a complex, rarely have interactions with all other proteins in that complex. Another class of community detection algorithms rely on finding dense sub-networks in PPI network (not essentially cliques) (Altaf-Ul-Amin et al., 2006). Such algorithms mis-classify low-shell proteins into distinct clusters, where as they could have been classified in a same core cluster. Due to this weak connectivity, many biological interactions are ignored. Similar pitfalls are also been observed, when hierarchical community detection algorithms are used to find modules in PPI networks (Li et al., 2008; Rahman and Ngom, 2013). In this paper, we present a bi-level community

detection algorithm based on label propagation, which eliminates the above pitfalls in identifying modules in PPI networks, and also helps to control large communities which dominate the network. Hence we retrieve communities with high modularity, high accuracy of matching with ground truth functional modules and in less computational time.

2 Related work

2.1 Multi-level spectral algorithms

Ramadan et al. (2005) proposed a two-level architecture for a yeast proteomic network. They construct small networks from a PPI network by removing proteins which interact with too many or too few proteins. Removing proteins with too few interactions can eliminate many effects of false positives. These proteins have a tendency to be classified in larger clusters to which it interacts, even if the interaction has low confidence value. Such low-shell proteins generally group with other low-shell proteins. There are specific proteins that function as substrates or agonists of protein receptors, such as G-protein coupled receptors (GPCR's). This is a specific interaction between molecules that have concordant configurations. We can also apply clustering again to these low-shell proteins which are removed, to avoid ignoring important biological processes. On the other hand, removing proteins with the largest numbers of interactions can make the finer structure of the interactions more evident. A clustering algorithm is applied to this residual network. Validation of clusters is performed by comparing the clustering result with a protein complex database, the Munich Information Center for Protein Sequences (MIPS). A spectral clustering method plays a critical role for identifying functional modules in the PPI network in their research.

One author of this paper has successfully applied a multilevel spectral algorithm to cluster a group of documents using similarity matrices which are mostly dense with entries between 0 and 1 (Oliveira and Seok, 2005) and has developed a matrix-based multilevel approach to identify functional protein modules (Oliveira and Seok, 2008). Like large-scale networks, the vertex connectivities of proteomic networks follow a scale-free power-law distribution (Bornholdt and Schuster, 2006). That means that, the proteomic network consists of a small number of high degree nodes and a majority of low degree nodes. However, the proteomic network has no edge weights. Multilevel algorithms have a long history, mostly for partial differential equations in numerical analysis but also for network partitioning, such as METIS (Karypis and Kumar, 1995). Multilevel schemes have been applied to network clustering too (Dhillon et al., 2005; Oliveira and Seok, 2005).

2.2 Community detection algorithms

A community in a network is a set of nodes that are densely connected with each other and sparsely connected to the other nodes in the network. A group of proteins in a same community within a PPI network, frequently coincide with known functional modules or protein complexes. Many proposed algorithms to detect community structures in complex networks are based on graph theory. Most of the work in this area is focused on enhancing the modularity, that is to increase the number of intra-cluster edges as opposed to inter-cluster edges. In our comparative study, we use modularity as a metric to determine the quality of our results over other well known algorithms, when applied to

PPI network. Some of these algorithms use techniques like betweenness centrality (De Meo et al., 2011), hierarchical clustering (Vieira et al., 2014), and label propagation (Kothapalli et al., 2013). A thorough review of community detection algorithms for networks is given in Fortunato (2010). It consists of various techniques, methods and datasets for detecting communities in biology, computer science and other disciplines, where the system is represented as a network.

In this paper we present the algorithm of Soman and Narang (2011) as a bi-level algorithm. We use it to identify functional protein modules in PPI networks with high accuracy. The BLLP algorithm involves a pre-processing stage where the edge weights of the network are computed based on its topological features, a step similar to coarsening of the original network, followed by a label propagation algorithm (Raghavan et al., 2007) and a post-processing step to improve the quality of the communities detected. Even though it is possible for a protein to have multiple functional modules, our algorithm does not detect overlapping communities, i.e., does not identify multiple functional module for a single protein. Instead it identifies one strong functional module for a protein under consideration.

The remainder of the paper is organised as follows. In Section 3, we will talk about important features of PPI networks. In Section 4, we describe the PPI yeast network that is used in this paper for all simulations. In Section 5, we describe the bi-level label propagation algorithm followed by Section 6, where we show the computational results along with a comparative study with other well known community detection algorithms (De Meo et al., 2011; Kothapalli et al., 2013; Blondel et al., 2008). We also show the accuracy of BLLP algorithm in predicting functional modules of uncharacterised proteins. We finally end the paper by giving implementation details and conclusion in Section 7.

3 Features of PPI networks

Graph theory is commonly used as a method for analysing PPIs in Computational Biology. Each vertex represents a protein, and edges correspond to experimentally identified PPIs. Proteomic networks have two important features (Born-holt and Schuster, 2003). One is that the degree distribution function $P(k)$ (the number of nodes with degree k) follows a power law $P(k) \approx$ constant $k^{-\alpha}$ and so is considered a scale-free network. This means that, most vertices have low degrees, called low-shell proteins (defined in Section 3), and a few are highly connected, called hub proteins. The other feature is the *small world* property which is also known as *six degrees of separation.* This means the diameter of the network is small compared with the number of nodes.

The standard tools to understand these networks are the clustering coefficient (Cc), the average path length, and the diameter of the network. The clustering coefficient (Cc_i) is defined in terms of $E(G(v_i))$, the set of edges in the neighbourhood of v_i. The clustering coefficient is the probability that a pair of randomly chosen neighbours of v_i are connected. That is,

$$Cc_i = \frac{2}{deg(v_i)[deg(v_i)-1]} \cdot |E(G(v_i))| \tag{1}$$

where $G(v_i)$ is the neighbourhood of v_i, and $E(G(v_i))$ is the set of edges in $G(v_i)$.

Since the denominator is the maximum possible number of edges between vertices connected to v_i, $0 \le Cc_i \le 1$. The global clustering coefficient can be simply the average of all individual clustering coefficients (Cc_i) like

$$\overline{Cc} = \sum_{i=1}^{n} Cc_i / n \tag{2}$$

But this 'average of an average' is not very informative (Bornholt and Schuster, 2003); one alternative is to weight each local clustering coefficient

$$Cc = \sum_{i=1}^{n} \frac{deg(v_i)}{MaxDeg} Cc_i / n \tag{3}$$

where *MaxDeg* is the maximum degree in the network. We use the latter *Cc* as our clustering coefficient for the rest of the paper.

The path length of two nodes v_i and v_j is the smallest sum of edge weights of paths connecting v_i and v_j. For an unweighted network, it is the smallest number of edges connecting v_i and v_j. The average path length is the average of path lengths of all pairs (v_i,v_j). The diameter of the network is the maximum path length. Some other important features about PPI networks are:

- The hub proteins have interactions with many other proteins, so it is hard to limit them to only one cluster and the computational complexity increases when they are included.
- There are many low-shell proteins, which increases the size of the network. These nodes are easy to cluster when the nodes they are connected to are clustered first.
- Proteomic networks are mostly comprised of one big component and several small components.

In Section 4, we talk about these features with our model of yeast network in more details.

4 Model PPI network for yeast

There are databases which contain protein–protein interactions as well as cellular localisation, gene regulation and the context of these interactions. The best known are KEGG (www.genome.ad.jp/kegg), BIND (www.bind.ca), MIPS (mips.gsf.de), PROnet (www.pronet.doubletwist.com) and DIP (dip.doe-mbi.ucla.edu) which are used to test the algorithms.

Over the last decade, high throughput interaction detection approaches like yeast two-hybrid system (Uetz et al., 2000), protein complex purification technique using mass spectrometry (Ho et al., 2002), correlated messenger RNA expression and genetic interaction data (Hughes et al., 2000), etc., have created number of datasets of PPI interactions for several eukaryotic organisms, like yeast. The interactions produced by using the above techniques have many false positives. In order to measure the accuracy of these interactions, Von Mering et al. (2002) checked 80,000 interactions amongst 5400 yeast proteins and assigned each interaction a confidence value. To eliminate the effect of false positives in our predictions, we focus on 7182 interactions with high confidence

value (above 75%), among 2361 proteins. This proteomic network for yeast is given in Pajek Datasets (http://vlado.fmf.uni-lj.si/pub/networks/data/bio/Yeast/Yeast.htm). To evaluate the clustering results, we compare with the functional modules given in the Munich Information Center for Protein Sequence (MIPS), which has a list of experimentally trusted functional modules in yeast. Each protein is given a short label for identification (called PIN on MIPS) and a functional module. We select 57 proteins randomly from 2361 proteins, having the following functional modules: 20 r-RNA, 11 Ribosome biogenesis, 7 Splicing, 6 Lipid oxidation, 2 Protein binding, 3 t-RNA, 3 m-RNA, 2 Tricarboxylic acid pathway, 2 Nuclear degradation and 1 Catabolism. Then we mask these functional modules for the 57 proteins and term them as uncharacterised proteins. We then make predictions on which functional modules they belong to, based on the BLLP algorithm's outcome and compare it with the ground truth functional module. We also conduct network analysis to determine the following features of PPI Yeast network:

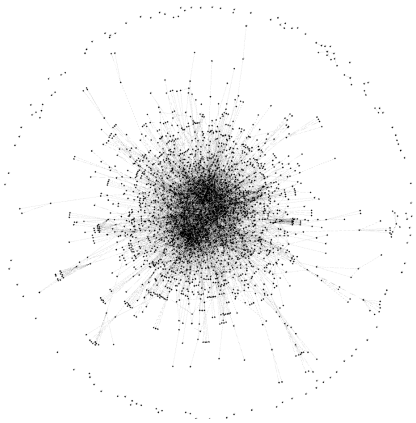
4.1 Betweenness centrality distribution

Betweenness centrality is the number of shortest paths between a pair of vertices that pass through a node. There is a pair of proteins in the PPI yeast network, through which ≈ 950 shortest paths pass. These proteins are considered as a hub-proteins. In our data set, these proteins are labelled YMR093W and YHR052W. Their functional module is r-RNA processing. These proteins have majority of interactions with other proteins within the largest community we detected in our dataset, being the functional module for r-RNA processing. They also have considerable interaction with proteins in the ribosome biogenesis functional module.

4.2 Clustering coefficient

The clustering coefficient of the PPI yeast network, i.e., the measure of the degree to which nodes in this network tend to cluster together is 0.271. There are 1300 nodes whose clustering coefficient is 0; i.e., low-shell proteins.

Figure 1 PPI yeast network in which 2361 proteins are linked by 7182 interactions and 536 self interactions



4.3 Closeness centrality distribution

The closeness centrality C_i of a node i in a network is the inverse of the mean shortest path distance from i to every other node in the network

$$C_i = \frac{n-1}{\sum_{i \neq j} dist(i,j)} \tag{4}$$

where $dist(i,j)$ is the shortest path distance between nodes i and j, and n is total number of nodes in the network. If there exists no path between i and j then, n nodes are used in equation 4, instead of the path length.

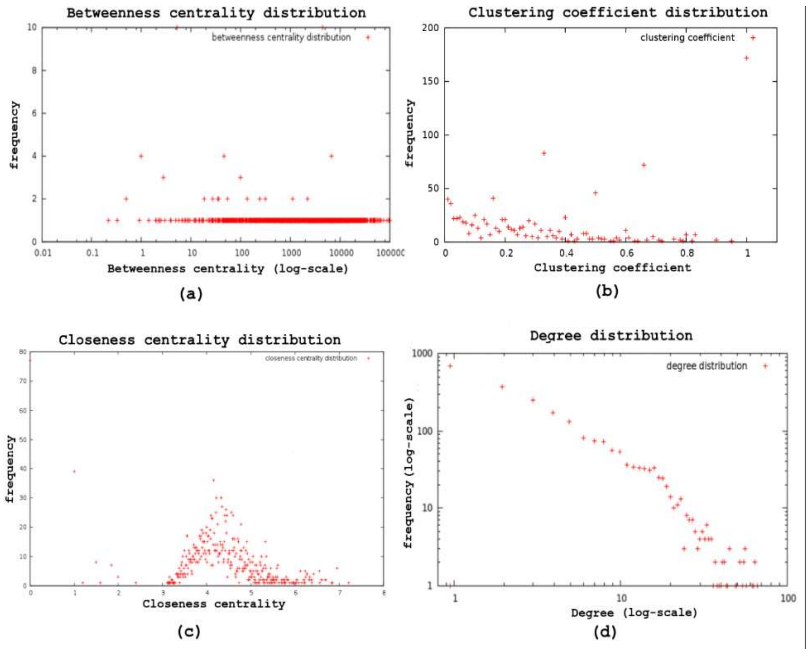
4.4 Degree distribution

In Figure 2(d), we can see that the degree distribution function for PPI yeast network $P(k)$ (the number of nodes with degree k) follows a power law $P(k) \approx$ constant $k^{-\alpha}$ and hence is a scale-free network (Bornholdt and Schuster, 2006; Pržulj, 2007; Pržulj et al., 2004).

4.5 Diameter, average path length, number of shortest paths and connected components

The density of the network is 0.003. The diameter of the yeast network is 11. So the diameter is much smaller as compared to the number of nodes in the PPI yeast network network, thus it exhibits the *small world property*. The average path length is 4.3762. There are 78 small connected components.

Figure 2 Properties of PPI yeast network (see online version for colours)



5 BLLP algorithm

In this section, we present the algorithm of (Soman and Narang, 2011) in a bi-level algorithmic framework, involving a pre-processing stage where the edge weights of the network are computed based on its topological features, a step similar to coarsening of the original network (level 1), followed by applying label propagation algorithm (Raghavan et al., 2007) once on coarsened network (level 2) and then iteratively in level 1, incorporating a post-processing step to improve the quality of the communities detected.

5.1 Pre-processing: topological weight assignment (Level 1)

The BLLP algorithm finds communities in a network $G(V, E)$ where V represents the nodes/vertices/proteins and E represents the edges/interactions between the nodes, by assigning weights to the edges and tracking the propagation of the label through the network. It is desirable to assign edge weights that most accurately represent the topological structure of the network in the BLLP algorithm. Since we do not have any prior knowledge of the community structure, we assign weights to each edge based on the significance of that edge to the other nodes in the network, and to the nodes at the end points of that edge. For each edge $e(i, j)$ (where i and j are nodes) in the fine network G, the topological edge weight $w_{top}(i,j)$ assigned to it is the ratio of number of triangles that edge $e(i, j)$ participates in to the total number of triangles containing node i. If the weight of the edge $e(i, j)$ is greater than other edges in the 1-neighbourhood of i then, node i and node j are more likely to be in the same community. Whereas on the contrary, if edge $e(i,j)$ has lower weight than most other edges in the 1-neighbourhood of i, then node i and node j are less likely to be in the same community.

Mathematically,

$$w_{top}(i, j) = \frac{t_{(i,j)}}{\sum_{(i,k)}(t_{(i,k)})};\ k \in N_i \tag{5}$$

where N_i is the 1-neighbourhood of i, and $t_{(i, j)}$ is the total number of triangles whose sides contain edge (i, j).

The BLLP algorithm also works well with weighted networks, where the edges are assigned weights w_{input} as an input. To get the total weight of an edge, we simply have to take product of the topological weight of that edge, with its input weight.

$$w_{total}(i, j) = w_{top}(i, j) \times w_{input}(i, j) \tag{6}$$

We initialise the label of each node in the fine network G to their corresponding node id, which is also the label weight for that node. The propagation function used to transfer labels between nodes in the network is given by:

$$L(i) = argmax_{j \in N_i}(L_j) \tag{7}$$

where L_j is the label (also referred as weight of the label) of node j in N_j (nodes one edge away from i). In the later subsections, we modify this propagation function before using it in the label propagation step.

5.2 Coarsening

In this step of the BLLP algorithm, we apply *coarsening* to the fine network *G.* For each node *i* of network *G,* we find the maximum weighted edge $e(i, j)$ in its 1-neighbourhood. Now we find all such edges in the fine network *G* and copy them to a new network $G' = (V', E')$. That is, the set of edges E' is the set of edges $e(i, j)$ where $e(i, j)$ has the maximum weight in the 1-neighbourhood of *i.* The set of vertices is the set of all *i* and *j* where $e(i,j)$ is in E'. Note that $|V| > |V'|$.

There always exists some node pairs say n_1 and n_2 such that (n_1,n_2) is connected with a maximum weight edge and conversely so is (n_2,n_1). It is very important to label these nodes with the common label, to avoid *oscillatory behaviour*, where these 2 nodes will keep on exchanging labels and algorithm will not converge, thus degrading the quality of the results.

5.3 Labelling and interpolation

Now in *level 2* we find the connected components in this coarse network G' and then perform a label propagation algorithm until all the nodes in each component have common labels. This phase of finding locally relevant communities ensures good quality results. By doing this we also eliminate all the overlapping pairs. These components are local communities in the network. We then transfer the labels of the nodes in coarse network G' back to the fine network *G.* These are the new labels which will be used when the label propagation step begins.

Now before we go ahead let us take a small example to understand the pre-processing, coarsening, and interpolation steps of the algorithm. We apply pre-processing step to the fine network *G* in Figure 3(a), where weights are assigned to all the edges based on the topological structure of *G.* Assume the weights are assigned as shown in Figure 3(b). All six nodes are given initial labels corresponding to their node identifier. In the coarsening step of BLLP algorithm, for each node in the fine network *G,* we find the maximum weighted edge in its 1-neighbourhood. For example in Figure 3(b), *edge*(1*,* 2) is the maximum weighted edge for node 1 as well as node 2. Similarly, *edge*(3*,* 5) and *edge*(5*,* 6) are maximum weighted edges in the 1-neighbourhoods of nodes 4 and 6 respectively. We copy all such edges and corresponding nodes in the new network G'. In the interpolation step we find connected components in this coarse network G'. Then we apply one iteration of label propagation, where all the nodes in G' send their label to every other node in their 1-neighbourhood and each node assigns itself the lowest label it receives. This way all the nodes in each component have a common label. In Figure 3(d), we have two connected components in G' with label 1 and label 3. We then transfer these labels from coarse network G' to fine network *G* shown in Figure 3(e).

5.4 Label propagation (Level 1)

Now we shall apply the label propagation algorithm to the fine network *G*. Each iteration of the label propagation algorithm includes the following steps;

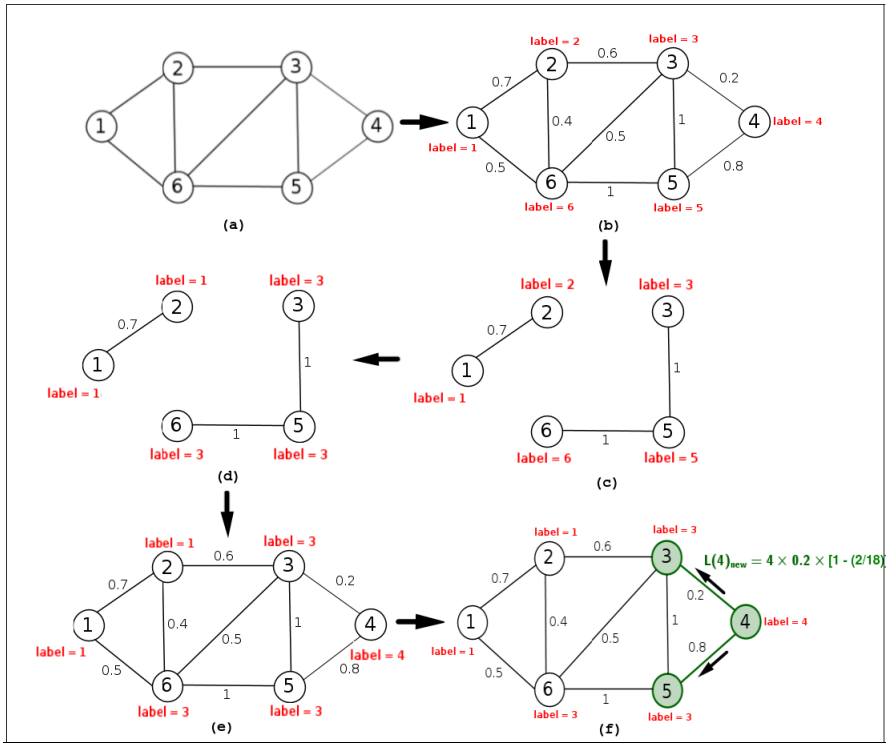
1 every node i in the fine network G sends a label $L(i)_{new}$ given in equation 9, to every other node in its 1-neighbourhood

2 each node then assigns itself the maximum value label it receives

3 if the label of some node changed, repeat step 1

The algorithm terminates when all the nodes are stably labelled. It takes 6 iterations for the algorithm to terminate on the PPI yeast network. The nodes having the same labels after the termination of the label propagation algorithm, belong to the same community.

Figure 3 (a) Network G (b) Pre-processing: Topological weight Assignment (Level 1) (c) Coarse graph G' (d) Labelling : Find connected components and give common label to nodes in same component (e) Interpolation: Transfer labels from G to G' (f) Label Propagation (Level 1) (see online version for colours)



We use the label given in equation (9) and not the label $L(i)$ given in equation (7) because, if there is a large dominating community in the network, it will dominate all the other communities leading to the scenario where all the nodes in the network have same label. This is similar to spread of disease and is also referred to as *epidemic spread* in the network. BLLP *controls the community size* by assigning weight to each label, in each iteration of the label propagation, based on the following formula;

$$W_{label}(L_i) = 1 - \frac{d_C}{2m} \tag{8}$$

where d_C is the sum of degrees of all the nodes in community C with label L_i and m is total number of edges in the network. The modified label propagation function is

$$L(i)_{new} = argmax_{j \in N_i}[(L_j) \times w_{total}(j,i) \times W_{label}(L_i)] \tag{9}$$

where L_j is the label of node j, and w_{total} (j,i) is the total weight of an *edge*(i,j). So we can see that, if the size of the community increases the value of $W_{label}(L_i)$ decrease and so the chances of this label to propagate further in the network decreases. Conversely, if the size of the community decreases the value of W_{label} (L_j) increases and so the chances of this label to propagate further in the network increases.

For more clarity on how labels are propagated, let us consider the example given in Figure 3(f), where node 4 propagates its label to its neighbour node 3. The label propagation function used by node 4 to propagate its label to node 3 is given by $L(4)_{new}$ = (current label of node 4) × (edge weight of *edge*$(4, 3)$) $\times W_{label}(L_4)$, where W_{label} (L_4) is determined by equation (8), which prevents the epidemic spread in the network. Node 3 receives labels from nodes 2, 4, 5 and 6 and then assigns itself the maximum value label it receives.

We determine the quality of the detected communities using the modularity metric, denoted by,

$$Q = \frac{1}{2m}\left[\sum_{i,j}[A_{(i,j)}\delta(C_iC_j)] - \sum_i \frac{dC_i}{d_i}\right] \tag{10}$$

where $A_{(i,j)}$ is the adjacency matrix. C_i and C_j denotes the communities in which nodes i and j belong respectively. δ (C_i,C_j) is the Kronecker function such that

$$\delta(C_{i,}C_j) = \begin{cases} 1; & C_i = C_j \\ 0; & C_i \neq C_j \end{cases}$$

d_{C_i} denotes the degree of community C_i and d_i is the degree of node i.

6 Computational results

Computational results for BLLP algorithm when applied to PPI yeast network is shown in this section. We show that, using our algorithm highly accurate predictions are made to identify functional modules for uncharacterised proteins. We also conduct a comparative study, comparing three different community detection algorithms (De Meo et al., 2011; Kothapalli et al., 2013; Blondel et al., 2008) on PPI yeast network, based on modularity of the communities discovered and computational time.

6.1 BLLP on PPI yeast network

The PPI yeast network we studied involves, 2361 proteins and 7182 interactions with 536 self-interactions and 78 small components. BLLP extracts 123 different communities from this data set. The modularity of the extracted communities is 0.592. The MIPS Comprehensive Yeast Genome Database (CYGD) provides the catalogue of protein–protein interactions, the protein complex catalogue and the protein localisation catalogue

which stores information related to the proximity of proteins in yeast (Monien et al., 2000). The protein complexes include more than 200 manually extracted protein complexes. We check the communities obtained by BLLP against the MIPS database to determine the percentage of correct matching.

Figure 4 (a) PPI yeast network with 123 communities: red = largest community with 1279 nodes, blue = second largest community with 602 nodes (b) Correctness of groupings made by BLLP (see online version for colours)

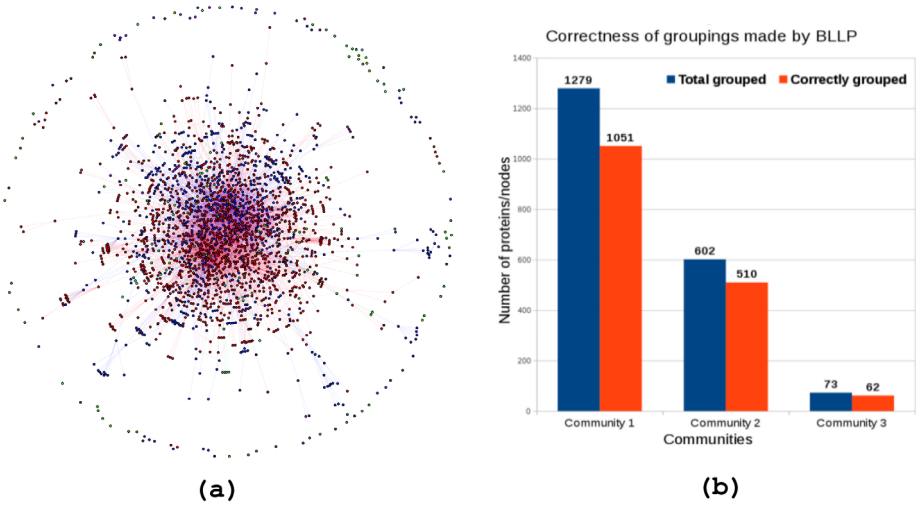


Table 1 Jaccard Index to quantify the distance between protein complexes in MIPS database and functional module partitions by BLLP algorithm. (1205, 580, and 64 in rows 1, 2, and 3 are total number of grouped proteins in largest 3 ground truth communities respectively).

Communities	*Jaccard Coefficient*	*Jaccard Distance*
Community 1	$\frac{1051}{1279+1205-1051}$	1 – 0.7334 = 0.267
Community 2	$\frac{510}{602+580-510}$	1 – 0.759 = 0.241
Community 3	$\frac{62}{73+64-62}$	1 – 0.827 = 0.173

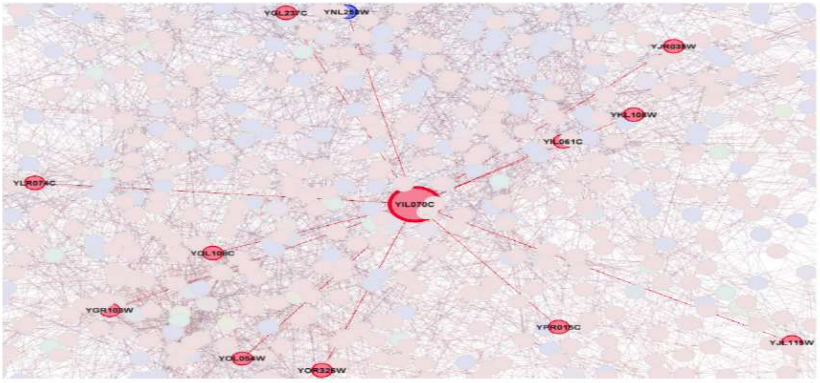
In Table 1, Jaccard coefficient and Jaccard distance for the largest 3 communities detected by BLLP compared with MIPS protein complexes is shown. Jaccard similarity coefficient, is a statistic used for measuring similarity between finite sample sets, and is defined as the size of the intersection divided by the size of the union of the sample sets. The Jaccard distance, which measures dissimilarity between sample sets, is complementary to the Jaccard similarity coefficient (Downton and Brennan, 1980; Sah, et al., 2014). As shown in Figure 4, in the largest community having 1279 proteins; the BLLP algorithm matches 1051 proteins correctly to their functional module i.e.82.20% correctness. In the second largest community having 602 proteins of which 510 proteins have been correctly matched i.e. 84.71% correctness and in the third largest community

having 73 proteins of which 62 proteins have been correctly matched i.e. 84.93% correctness. All the components with ≤ 25 nodes and > 1 node are grouped with 92+% accuracy.

Table 2 Number of detected communities and corresponding sizes, in PPI yeast network

No. of communities	*Nodes*	*% of whole network*
1	1279	54.21
1	602	25.5
1	73	3.13
1	44	1.86
1	38	1.61
1	27	1.14
1	25	1.06
1	18	0.76
1	12	0.64
1	10	0.51
2	9	0.34
2	7	0.3
5	6	0.25
6	4	0.21
5	3	0.13
17	2	0.08
76	1	0.04

Figure 5 YIL070C uncharacterised protein interacts with other known proteins (see online version for colours)



We observe that, the largest community with 1279 proteins is dominated by proteins with functional module of r-RNA processing; of which 1051 proteins are classified correctly and 91% of mis-classified proteins belong to functional module of ribosome biogenesis. The second largest community with 602 proteins is dominated by proteins within the ribosome biogenesis functional module; of which 510 proteins are classified correctly

and 87% of mis-classified proteins belong to the functional module for r-RNA processing. From this observation we infer that there are many proteins in the largest two communities detected, which may have multiple functional modules i.e., ribosome biogenesis as well as r-RNA processing. The third largest community with 73 proteins is dominated by proteins in the splicing functional module, of which 62 proteins are classified correctly and remaining 11 proteins are mis-classified to functional module for r-RNA processing and oxidation of fatty acids, lipids, and bio-synthesis. Some of these 11 mis-classified proteins are classified into larger communities, which are dominated by proteins in the r-RNA and ribosome biogenesis functional modules. This effect is much less as compared to same seen, when other community detection algorithms (in Section 6.3) are applied to the PPI yeast network.

Table 3 Incorrect prediction of functional modules made by the BLLP algorithm, for uncharacterised proteins

No.	*Protein Pin*	*BLLP assigned modules*	*Correct modules*
1	YKL155C	r-RNA processing	Ribosome biogenesis
2	YNL177C	r-RNA processing	Ribosome biogenesis
3	YMR158W	r-RNA processing	Ribosome biogenesis
4	YPL012W	Ribosome biogenesis	r-RNA processing
5	YPR144C	Ribosome biogenesis	r-RNA processing
6	YNL002C	Ribosome biogenesis	r-RNA processing
7	YJL069C	Ribosome biogenesis	r-RNA processing
8	YGL111W	Ribosome biogenesis	r-RNA processing
9	YLR222C	Ribosome biogenesis	r-RNA processing
10	YDR428C	Splicing	Lipid oxidation
11	YLR186W	Splicing	r-RNA processing

6.2 Functional module prediction for uncharacterised proteins

We have 57 uncharacterised proteins in our data set and we shall make predictions based on the communities detected by BLLP algorithm. We check the correctness of these predictions against the MIPS database, to determine the accuracy of our results. We determine the functional modules of uncharacterised proteins, by checking its interactions with known proteins. We assign the uncharacterised protein, the functional module of that of majority of the known proteins that it interacts with, belonging to the same community. For example, YIL070C interacts with YGL237C, YIL061C, YKL108W, YJR035W, YOL108C, YOL054W, YPR015C, YJL115W, etc., where majority of these have a functional module r- RNA, so we assign YIL070C the functional module r-RNA. In case of a tie, we assign any one of the functional module randomly, to the uncharacterised protein. It is possible for a protein to have multiple functional modules, but our algorithm does not resolve the issue of overlapping communities in the network. We mainly focus on finding one strong functional module for an uncharacterised protein. We follow the same procedure for all the uncharacterised proteins and assign them a functional module based on same rule. We then verify our functional module assignment using the MIPS database/ground-truth dataset, making 46 out of 57 correct predictions i.e. accuracy of 80% for PPI yeast dataset. Functional

modules of 11 out of 57 uncharacterised proteins are incorrectly predicted, for similar reasons mentioned in section 6.1, paragraph 3. In Table 3, we have shown 11 uncharacterised proteins whose functional modules are incorrectly predicted, and Table 4 contains 46 uncharacterised proteins whose functional modules are correctly predicted, by BLLP.

Table 4 Correct prediction of functional modules made by the BLLP algorithm, for uncharacterised proteins

No.	*Protein Pin*	*BLLP assigned modules*
1	YGR263C	Catabolism
2	YBL004W	r-RNA processing
3	YPR034W	r-RNA processing
4	YDR449C	r-RNA processing
5	YIL070C	r-RNA processing
6	YER126C	r-RNA processing
7	YGR128C	r-RNA processing
8	YGR145W	r-RNA processing
9	YHR196H	r-RNA processing
10	YHR088W	r-RNA processing
11	YHR052W	r-RNA processing
12	YKR060W	r-RNA processing
13	YMR093W	r-RNA processing
14	YOR001W	r-RNA processing
15	YDR517W	Lipid oxidation
16	YOR093C	Lipid oxidation
17	YBL055C	Lipid oxidation
18	YDL193W	Lipid oxidation
19	YNL026W	Lipid oxidation
20	YGL211W	Protein binding
21	YGL059W	Protein binding
22	YMR310C	Ribosome biogenesis
23	YMR117C	Ribosome biogenesis
24	YMR074C	Ribosome biogenesis
25	YIL093C	Ribosome biogenesis
26	YDR036C	Ribosome biogenesis
27	YGL129C	Ribosome biogenesis
28	YJR014W	Ribosome biogenesis
29	YGR156W	Ribosome biogenesis
30	YGR285C	Splicing
31	YDL209H	Splicing
32	YKL018W	Splicing

Table 4 Correct prediction of functional modules made by the BLLP algorithm, for uncharacterised proteins (continued)

No.	*Protein Pin*	*BLLP assigned modules*
33	YKL059C	Splicing
34	YLR424W	Splicing
35	YPL151C	Splicing
36	YGR278W	Splicing
37	YNL123W	t-RNA synthesis
38	YDR428C	t-RNA synthesis
39	YJR008W	t-RNA synthesis
40	YJL046W	Tricarboxylic acid pathway
41	YNL168C	Tricarboxylic acid pathway
42	YLR421C	Nuclear degradation
43	YPL252C	Nuclear degradation
44	YDR018C	m-RNA processing
45	YLR074C	m-RNA processing
46	YKR081C	m-RNA processing

6.3 Comparative study

In this section, we compare the results obtained by applying the following 3 community detection algorithms to PPI yeast network, to that obtained by BLLP.

1 Fast unfolding of communities in large networks (FC) (Blondel et al., 2008), which is a heuristic method based on modularity optimisation.

2 Label Propagation Algorithm for Community Detection (LPA) (Kothapalli et al., 2013), uses label propagation method, without the bi-level frame-work used in BLLP.

3 Generalised Louvain Method for Community Detection in Large Networks (LM) (De Meo et al., 2011) also uses concept of network modularity optimisation, while exploiting the measure of edge centrality.

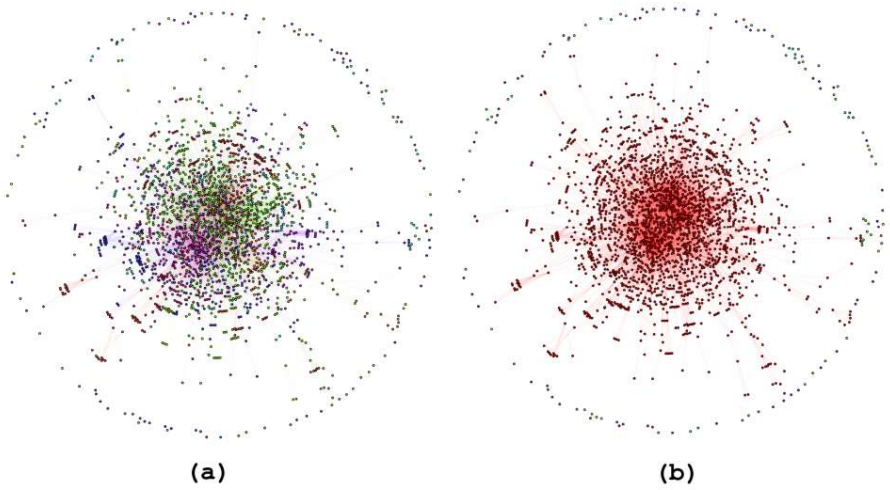
FC algorithm gives 115 communities with the largest community having 250 proteins (Figure 6(a)). The PPI networks has many low-shell proteins, which increases the size of network. They generally comprise of one big component and several small components. So the idea to divide the networks based on modularity optimisation doesn't yield good results for the PPI yeast network. The grouping made by FC is highly inaccurate. In the largest community with 250 proteins, 60% of proteins has functional module of r-RNA processing, 22% has functional module of ribosome biogenesis, and the remaining proteins has multiple mixed functional modules. LM algorithm behaves quite similar to FC, since both these algorithms are based on the same concept of modularity optimisation. Whereas LPA algorithm gives 94 communities (Figure 6(b)), but there is an *epidemic spread* which gives rise to a community of size 2224 proteins. This results in low modularity communities and incorrect grouping of proteins. To avoid this situation BLLP takes care that *epidemic spread* does not take over the network.

In order to detect functional modules in PPI yeast network, one should not just focus on modularity maximisation community detection algorithms. More novel community detection techniques are required for correct groupings of proteins in these networks. Novel techniques involving pre-processing the original network based on its topological features. Also multi-level algorithmic frame-works have a better scope to achieve accurate clustering results on biological networks, and thus making accurate predictions.

Table 5 Comparing various community detection algorithms on PPI yeast network

Algorithm	*Modularity*	*Time*	*No. communities*
FC	0.581	2.56 sec	115
LPA	0.10	24.23 sec	94
LM	0.546	7.72 sec	111
BLLP	0.592	6.23 sec	123

Figure 6 (a) FC clustering with largest community having 250 proteins (b) LPA clustering with largest community having 2224 proteins (see online version for colours)



Since we are just comparing the various algorithms on PPI yeast dataset, which is not a huge dataset, computational time does not play an important role. In case of huge datasets, BLLP is the only algorithm which is designed to be easily scalable, that is, can be parallelised for multi-core and GPU architecture.

7 Implementation and conclusion

We implemented the BLLP algorithm using C++ and graph libraries. All the simulations are done on Processor-Intel Core i7 3770 3.4GHz and Turbo Boost enabled Memory-16GB DDR3-1600 RAM 500G 3GB/s 7200 RPM; Linux machines. All the plots are done using Gephi and Gnuplot.

This research focuses on matching groups of proteins which are more likely to be part of the same functional modules. Using bi-level community detection algorithms with label propagation we achieve more accurate groupings of proteins in less computational time. Our computations show that the predictions made to determine the functional module of uncharacterised protein is also highly accurate. Our computational analysis also proves that, BLLP algorithm extracts higher modularity communities when compared to other well known community detection algorithms. Comparatively, the computational time is also close to the best.

Further applications of community detection BLLP algorithm includes other complex networks such as genetic networks, the World Wide Web, citation networks, biological networks and social networks.

Acknowledgement

We would like to thank the reviewers for their time and helpful advice.

References

Altaf-Ul-Amin, M., Shinbo, Y., Mihara, K., Kurokawa, K. and Kanaya, S. (2006) 'Development and implementation of an algorithm for detection of protein complexes in large interaction networks', *BMC bioinformatics*, Vol. 7, No. 1, p.207.

Bader, G.D. and Hogue, C.W. (2003) 'An automated method for finding molecular complexes in large protein interaction networks', *BMC Bioinformatics*, Vol. 57, No. 1.

Blondel, V.D., Guillaume, J-L., Lambiotte, R. and Lefebvre, E. (2008) 'Fast unfolding of communities in large networks', *Journal of Statistical Mechanics: Theory and Experiment*, Vol. 2008, No. 10.

Bornholdt, S., and Schuster, H.G. (2006) *Handbook of Graphs and Networks: From the Genome to the Internet*, John Wiley and Sons.

De Meo, P., Ferrara, E., Fiumara, G. and Provetti, A. (2011) 'Generalized Louvain method for community detection in large networks', *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference*, pp.88–93.

Dhillon, I., Guan, Y. and Kulis, B. (2005) 'A fast kernel-based multilevel algorithm for graph clustering', *Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pp.629–634.

Ding, C., He, X., Xiong, H. and Peng, H. (2006) 'Transitive closure and metric inequality of weighted graphs: detecting protein interaction modules using cliques', *International Journal of Data Mining and Bioinformatics*, Vol. 1, No. 2, pp.162–167.

Downton, M. and Brennan, T. (1980) 'Comparing classifications: an evaluation of several coefficients of partition agreement', *Class. Soc. Bull 4*, Vol. 4, pp.53–54.

Fortunato, S. (2010) 'Community detection in graphs', *Physics Reports*, Vol. 486, No. 3, pp.75–174.

Ho, Y., Gruhler, A., Heilbut, A., Bader, G.D., Moore, L., Adams, S.L. and Tyers, M. (2002) 'Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry', *Nature*, Vol. 415, No. 6868, pp.180–183.

Hughes, T.R., Marton, M.J., Jones, A.R., Roberts, C.J., Stoughton, R., Armour, C.D. and Friend, S.H. (2000) 'Functional discovery via a compendium of expression profiles', *Cell*, Vol. 102, No. 1, pp.109–126.

Karypis, G. and Kumar, V. (1995) *MeTis: Unstructured Graph Partitioning and Sparse Matrix Ordering System and Version 2.0*.

Kothapalli, K., Pemmaraju, S.V. and Sardeshmukh, V. (2013) 'On the analysis of a label propagation algorithm for community detection', *Distributed Computing and Networking*, pp.255–269.

Krogan, N. (2006) 'Global landscape of protein complexes in the yeast *Saccharomyces cerevisiae*', *Nature*, Vol. 440, No. 7084, pp.637–43.

Li, M., Wang, J. and Chen, J.E. (2008) 'A fast agglomerate algorithm for mining functional modules in protein interaction networks', *Biomedical Engineering and Informatics*, Vol. 1, pp.3–7.

Monien, B., Preis, R. and Diekmann, R. (2000) 'Quality matching and local improvement for multilevel graph-partitioning', *Parallel Computing*, Vol. 26, pp.1609–1634.

Oliveira, S, and Seok, S-C. (2005) 'A multi-level approach for document clustering', *Computational Science-ICCS*, Springer, Berlin Heidelberg, pp.204–211.

Oliveira, S, and Seok, S-C. (2008) 'A matrix-based multilevel approach to identify functional protein modules', *International Journal of Bioinformatics Research and Applications*, Vol. 4, No. 1 pp.11–27.

Pržulj, N. (2007) 'Biological network comparison using graphlet degree distribution', *Bioinformatics*, Vol. 23, No. 2, pp.e177–e183.

Pržulj, N., Corneil, D.G. and Jurisica, I. (2004) 'Modeling interactome: scale-free or geometric?' *Bioinformatics*, Vol. 20, No. 18, pp.3508–3515.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V. and Parisi, D. (2004) 'Defining and identifying communities in networks', *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 101, No. 9, pp.2658–2663.

Raghavan, U.N., Albert, R. and Kumara, S. (2007) 'Near linear time algorithm to detect community structures in large-scale networks', *Physical Review E*, Vol. 76, No. 3.

Rahman, M.S. and Ngom, A. (2013) 'A fast agglomerative community detection method for protein complex discovery in protein interaction networks', *Pattern Recognition in Bioinformatics*, Springer, Berlin Heidelberg, pp.1–12.

Ramadan, E., Osgood, C. and Pothen, A. (2005) 'The architecture of a proteomic network in the yeast', *Proceedings of CompLife2005, Lecture Notes in Bioinformatics*, Vol. 3695, pp.265–276.

Sah, P., Singh, L.O., Clauset, A. and Bansal, S. (2014) 'Exploring community structure in biological networks with random graphs', *BMC Bioinformatics 15*, Vol. 1, p.220.

Soman, J. and Narang, A. (2011) 'Fast community detection algorithm with GPUs and multicore architectures', *Parallel & Distributed Processing Symposium (IPDPS), 2011 IEEE International*, pp.568–579.

Spirin, V. and Mirny, L.A. (2003) 'Protein complexes and functional modules in molecular networks', *Proceedings of the National Academy of Sciences*, Vol. 100, No. 21, pp.12123–12128.

Uetz, P., Giot, L., Cagney, G., Mansfield, T.A., Judson, R.S., Knight, J.R. and Rothberg, J.M. (2000) 'A comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae*', *Nature*, Vol. 403, No. 6770, pp.623–627.

Vieira, V., Xavier, C.R., Ebecken, N.F.F. and Evsukoff, A.G. (2014) 'Modularity based hierarchical community detection in networks', *Computational Science and its Applications-ICCSA 2014*, pp.146–160.

Von Mering, C., Krause, R., Snel, B., Cornell, M., Oliver, S.G., Fields, S. and Bork, P. (2002) 'Comparative assessment of large-scale data sets of protein-protein interactions', *Nature*, Vol. 417, No. 6887, pp.399–403.

Xiong, H., He, X., Ding, C., Zhang, Y., Kumar, V. and Holbrook, S. (2005) 'Identification of functional modules in protein complexes via hyperclique pattern discovery', *Proceedings of Pacific Symposium on Biocomputing (PSB)*.

Yook, S.H., Oltvai, Z.N. and Barabasi, A.L. (2004) 'Functional and topological characterization of protein interaction networks', *Proteomics*, Vol. 4, No. 4, pp.928–942.

Zhang, L., Wong, S., King, O. and Roth, R. (2004) 'Predicting co-complexed protein pairs using genomic and proteomic data integration', *BMC Bioinformatics*, Vol. 5, No. 38.