Introducing the Concept of Second Neighbours to FPNC algorithm for Improving the Functional Modules Detection

Mohammad Shamsur Rahman and Nafisa Chowdhury

University of Windsor, Ontario, Canada, N9B 3P4.
Military Institute of Science and Technology, Dhaka-1216, Bangladesh.
rahman1v@uwindsor.ca,
nafisamou2004@gmail.com

Abstract. Proteins are biological polymers of amino acid residues. Proteins perform various functions within living organisms. Multiple proteins carry out these tasks by forming functional modules. Each functional module possesses community structure. For identifying functional modules, a lot of community detection or clustering algorithms were designed, but most of those algorithms suffer by inappropriate clustering results which do not make any sense biologically. Though some of the algorithms came out with better results but too high time complexity was not of great help. Recently an efficient algorithm was designed which outperformed other existing algorithms, named *Fast Protein Network Clustering* or *FPNC* algorithm. We have worked on that algorithm and improved its performance by introducing the concept of second neighbours (neighbours of neighbours of any vertex), named as *Second order Fast Protein Network Clustering* algorithm or *2nd order FPNC* algorithm. By coming up with the concept of 2nd neighbours, *2nd order FPNC* algorithm has better scoring function and better functional module mapping results indicating efficient identification of functional modules from any protein-protein Interaction network. These results have also shown that *2nd order FPNC* algorithm identifies the functional modules more accurately than existing algorithms. According to computational results *2nd order FPNC* algorithm put an important pace in the field of functional modules detection from protein-protein interaction networks.

Keywords: Second neighbours, Direct Neighbours, 2nd order FPNC algorithm, Protein-protein interaction networks.

1 Introduction

Protein plays a very crucial role in the cause of diseases, evolution of species, change of behaviour of any living organisms and other functionalities inside the genome including maintaining high degree order, catalyzing metabolic reactions, replicating DNA, stimulation, binding and releasing molecules etc. A single protein cannot carry out a specific task by itself. To carry out a specific task or a part of a task, two or more proteins interact with each other by forming either protein complexes or functional modules. Protein complexes are a set of proteins that interact with each other at the same time and place to perform a specific task in living organism[11]. On the other hand, functional modules are groups of proteins that participate in a particular cellular process while binding each other at a different time and place[11]. Basically a functional module is formed by the combination of single or multiple protein complexes. To predict the cause of diseases, evolution of species or other functionalities of living organism, it is very important to identify protein complexes and functional modules efficiently. In the field of bioinformatics, all proteins and their interactions are represented by a network, called "Protein-protein interaction (PPI)" network. Proteins of a functional module form a dense sub graph in any PPI network [11, 12]. It means proteins in a module possess community structure. Finding community structure provides useful information to predict the functional mod- ules or protein complexes of non-characterized proteins in the highly connected graph. So, Community detection in a PPI network is an important step to detect functional modules in a PPI network.

A lot of community detection algorithms were designed for detecting functional modules from PPI networks. These algorithms can be categorized into three groups [7]: a) density based, b) graph partitioning and c) hierarchical method. First two methods suffer a common problem- separately clusters a protein from its neighbour if its degree is one. Hierarchical based method is a method which represents the whole detection as a tree. At first, it considers all proteins are individual communities which are known as singletons. Singletons are merged together to form large communities based on some metrics or measures. This process continues till whole PPI network is represented as one community. It is known as agglomerative hierarchical method [2]. There is another hierarchical method available- divisive method which follows exactly inverse steps of agglomerative method [2].

In the paper, we modify the existing fastest agglomerative hierarchical algorithm: *Fast Protein Network Clustering* or *FPNC* algorithm by introducing the concept of second neighbours with direct neighbours of any vertex. The rest of the paper is organized as follow. In Section 2, we discuss a few hierarchical algorithms to which *2nd order FPNC* is based on. Section 3 introduces our proposed method. Computational experiments and discussions of results are given in Section 4 before we conclude with possible directions of research.

2 Related works

Many hierarchical clustering approaches (both agglomerative and divisive techniques) have been introduced in literature, since the original publication of Girvan and Newman in [3] for clustering networks. So, we represent few methods that are directly related to our proposed agglomerative approach.

In the year 2002, Girvan and Newman designed a hierarchical approach to obtain the communities based on edge-betweenness [3]. It is known as GN method. In this method, the algorithm finds the edge which is mostly used in all shortest path, called edge-betweenness. It removes this edge (having highest betweenness) to create a new cluster. This process follows divisive method. It was experiencing same fate as density and graph partitioning based approach. It also suffered the computational time issue and its time complexity is $O(nm^2)$ where n is the number of vertices and m is the number of edges of any graph G. One year later, Newman improvised the hierarchical community detection method by introducing ΔQ where Q is the modularity measure to calculate the compactness of a community [6]. It also improved the computational time and it is $O(n(m+n))$. Later Clauset and his professors modified Newman's ΔQ method [1], made the hierarchical method even faster (Computational complexity is $O(mk \log_2 n)$, where k is the depth of dendogram). But both of those two methods suffer same difficulty as GN method. In 2011, Wang et al. [12] designed a newer and faster hierarchical method for identifying communities from any PPI network named: *HC-PIN* based on *edge clustering values* or $ECV(u,v)$. ECV is designed for both weighted and unweighted PPI networks. The time complexity of *HC-PIN* algorithm is $O(\bar{d}^2ml)$ where $\bar{d}$ is the average degree of any PPI network; m is the number of edges; l is the number of repeated computations, depends on the number of the set of the values of λ. This method was also experiencing same problem like GN method.

Preferential attachment is a property of PPI networks where a low/ small degree vertex v (a protein) of a PPI network has higher tendency or motivation to attach with higher/ large degree vertex u (a protein) to form cluster [7]. None of the algorithms above were designed considering *preferential attachment* property PPI networks. Recently, Rahman et al. [7, 8], has designed a new approach in hierarchical method known as ***FAC-PIN*** algorithm based on *relative vertex-to-vertex clustering value* or $R(u \dashrightarrow v)$. *Relative vertex-to-vertex clustering value* has been designed by considering preferential attachment property.

$$R(u \dashrightarrow v) = \frac{|N_u^+ \cap N_v^+|}{|N_u^+|} \tag{1}$$

where N_u^+ is the neighbour list of vertex u where u itself is a neighbour of u. Basically, their relative vertex-to-vertex clustering values is designed for unweighted PPI network. It calculates the likelihood value of a vertex u to form cluster with another vertex v. Though it improves the performance of the agglomerative hierarchical method for detecting functional modules in a PPI network, the outcomes of the *FAC-PIN* algorithm also depend on another parameter α. It also affects the computational time. The computational time complexity of the algorithm for each α is $O(\bar{d}^2n)$ where $\bar{d}$ is the average degree of any PPI network and n is the number of vertices. So far, *FAC-PIN* is one of existing fastest hierarchical methods for clustering PPI networks, though it depends on the set of values of α and the time complexity is $O(\bar{d}^2nl)$ where l is the number of the set of values of α. Later Rahman et al. [9] redesigned *FAC-PIN* algorithm by introducing the concept of ΔQ, which removes the effect of α, with modified $R(u \dashrightarrow v)$:

$$R_w(u \dashrightarrow v) = \frac{\sum_{k \epsilon I^+_{u,v}} w(u,k)}{\sum_{s \epsilon N^+_u} w(u,s)} \tag{2}$$

Here, $I^+_{u,v}$ is the list of common neighbours between vertices u and v including u and v; N^+_u is the list of neighbour of vertex u including vertex u; $w(u,s)$ is the weight of the edge between vertices u and s. *Relative vertex-to-vertex clustering value* function is designed for both weighted and unweighted PINs. Equation- 1 is the special case when the weight of all edges of an unweighted PPI network is considered as one. ΔQ follows the same formula designed by Clauset et al.[1]:

$$\Delta Q_{u,v} = \begin{cases} \frac{1}{2m} - \frac{d_u * d_v}{(2m)^2} & \text{if } u \text{ and } v \text{ are connected} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Where d_u is the degree of the vertex u, but if u is in any cluster, d_u is the out degree of the cluster. $\Delta Q_{u,v}$ is the modularity increment/ decrement measure after adding a vertex v or a cluster of a vertex v with a vertex u or a cluster of vertex u. The newly designed algorithm is known as *Fast Protein Network Clustering Algorithm* or *FPNC* algorithm. After introducing ΔQ and $R_w(u \dashrightarrow v)$, the time complexity of *FPNC* algorithm has improved to $O(\bar{d}^2 n)$

In the following section, we discuss about "second neighbour" which is introduced to improve the performance of *FPNC* algorithm.

3 Second order FPNC Algorithm

All previous algorithms were designed using the direct neighbouring information. Specially the metrics/ measures of the latest three algorithms were designed based on direct neighbours. These three algorithms find the common direct neighbours between two vertices u and v to calculate either $ECV(u,v)$ or $R(u \dashrightarrow v)$ or $R_w(u \dashrightarrow v)$. But cluster cannot be formed by any vertex and its direct neighbours. There is also presence of indirect neighbours like second neighbours (neighbours of neighbours of any vertex) in any communities which are ignored in last three algorithms. Considering second neighbours in the computation of $R_w(u \dashrightarrow v)$ will enhance the performance of community detection process.

In this paper, we modify the $R_w(u \dashrightarrow v)$ by introducing the concept of 2nd neighbours with direct neighbours. $R_w(u \dashrightarrow v)$ is calculated by using neighbour list which contains only the list of direct neighbours of any vertex. We redesign our neighbour list by adding the second neighbours of any vertex. The modified neighbour list is as followed:

$$N^{2+}_u = N^{2+}_u \bigcup (N_v - N_u) \bigcup \{u\} \quad \forall v \;\epsilon\; N_u \tag{4}$$

Here N^{2+}_u is the neighbour list of any vertex u which contains the list of direct and 2nd neighbours including itself; N_v is the neighbour list of any vertex v. Initially N^{2+}_u contains only the list of direct neighbours of any vertex u i.e. N_u.

Modified neighbour list is also modified $R_w(u \dashrightarrow v)$ and renamed as $R_{w_2}(u \dashrightarrow v)$:

$$R_{w_2}(u \dashrightarrow v) = \frac{\sum_{k \epsilon I^{2+}_{u,v}} w(u,k)}{\sum_{s \epsilon N^{2+}_u} w(u,s)} \tag{5}$$

Here, $I^{2+}_{u,v}$ is the list of common direct and second neighbours between vertices u and v including u and v. According to the *preferential attachment* property, in any PPI network, any small degree vertex v has higher tendency to add and form cluster with any large degree vertex u. So, two cases are possible to happen between two vertices u and v where u and v are connected by an edge.

– vertex v has exactly one edge that is connected with vertex u where $deg(u, G) > 1$.
– vertex v is connected with multiple edges. The vertices v and u
  - share most (at least equal) of the direct neighbours of u and v.
  - share few number of direct neighbours of u and v.

Before discussing the possible conditions, let us consider the weights of all vertices are one. In first case, the value of $R_{w_2}(v \dashrightarrow u)$ is 1, because the $N^{2+}_v = \{\{u\}, \{v\}, N_u - \{v\}\}$, $N^{2+}_u = \{\{u\}, \{v\}, N_v - \{u\}, N_{N_u}\}$ and $I^{2+}_{u,v} = \{\{u\}, \{v\}, N_u - \{v\}\}$ where N_{N_u} is the list of only 2nd neighbours of vertex u. It means v has full possibilities to form cluster with u (the likelihood value is one). This is case, our newly designed $R_{w_2}(v \dashrightarrow u)$ is working as $R_w(v \dashrightarrow u)$ and maintains the preferential attachment property of any PPI network.

First condition of second case tells us, vertex v shares at least 50% of direct neighbours to vertex u. In this condition two possible situations can happen: either (a). the second neighbours among vertices v and u are at least 50% common or (b). less. In first situation, The value of $R_{w_2}(u \dashrightarrow v)$ is greater or equal to 0.5, which indicates vertex v has at least 50% possibilities to form cluster with u. In the second situation, the number of shared second neighbours is less than 50%, it tells that the direct neighbours of v and u belong to separate clusters, not in same. If the direct neighbours are in different or separate clusters, there is less possibilities to form a single cluster by vertices v and u. Our modified $R_{w_2}(u \dashrightarrow v)$ also produces the likelihood value which is less than 0.5. It will help our algorithm to detect more compact and accurate clusters.

In last situation, v has few possibilities to attach with u to make a cluster as per $R_w(u \dashrightarrow v)$. But if the direct neighbours of the vertices v and u shares maximal second neighbours, the direct neighbours of the v and u are belonging to same cluster; u and v should be in same cluster. Whereas $R_w(u \dashrightarrow v)$ wrongly misclassifies the vertices u and v into separate clsuters. In this situation, New $R_{w_2}(u \dashrightarrow v)$ gives higher likelihood value than $R_w(u \dashrightarrow v)$ and performs well. On the other hand, the neighbours of direct neighbours of the vertices v and u are few common to each other, the $R_{w_2}(u \dashrightarrow v)$ gives smaller likelihood value and tells v and u should not be in same cluster. By theoretically our modified $R_{w_2}(u \dashrightarrow v)$ performs efficiently than existing $R_w(u \dashrightarrow v)$ and can help *FPNC* algorithm to detect more accurate clusters.

In the algorithm part, accept the *initialization phase*, the *community detection phase* remain unchanged. We can call it as *Second order FPNC* algorithm.

Algorithm 1 The ***2nd Order FPNC*** Algorithm

```
Input: $G = (V, E)$: undirected PPI network
Output: $P_k = \{C_1, \dots, C_k\}$: identified collection of modules
  {Initialization phase}
  for every $v_i \in V$ do
    {Outer loop}
    $C(v_i) \leftarrow \{ \{v_i\}, \emptyset \}$; {each vertex is a singleton cluster}
    $N_v^{2+} \leftarrow N_v$
    for all $u \in N_v$ do
      {Inner loop-1}
      $N_v^{2+} \leftarrow N_v^{2+} \bigcup (N_v - N_u) \bigcup \{u\}$
    end for
    for all $u \in N_v$ do
      {Inner loop-2}
      Calculate and store $R_{w_2}(u \dashrightarrow v)$ into $R$.
    end for
  end for
  {Community detection phase}
  repeat
    $R_{w_2}(u \dashrightarrow v) \leftarrow R$; {select next highest $R_{w_2}(u \dashrightarrow v)$ in $R$}
    if $[R_{w_2}(u \dashrightarrow v) = 1]$ OR $[\Delta Q_{u,v} > 0]$ then
      $C(v) \leftarrow C(v) \cup \{ \{u\}, \{(u, v)\} \}$;
      $C(u) \leftarrow C(v)$;
    end if
  until $R = \emptyset$
  {Compute the partition $P_k$}
  $U \leftarrow V$;
  $i \leftarrow 1$;
  while $U \neq \emptyset$ do
    $v \leftarrow$ randomly select a vertex from $U$;
    $C_i \leftarrow C(v)$;
    $U \leftarrow U \setminus \{u \mid C(u) = C(v)\}$;
    $i \leftarrow i + 1$;
  end while
  return $P_k \leftarrow \{C_1, \dots, C_k\}$;
  Evaluate modularity $Q(P_k)$ of partition $P_k = \{C_1, \dots, C_k\}$;
```

Computational complexity analysis is very important because any PPI network contains more than million interactions and hundred thousands of proteins. Let $n = |V|$ be the number of vertices (proteins), $m = |E|$ be the number of edges (interactions), $\bar{d}$ be the average degree of all vertices ($\bar{d} = \frac{1}{n} \sum_{v \in V} deg(v, G)$) and the maximum vertex degree in G be $d_{\max} = \max_{v \in V} deg(v, G)$. The computation time complexity of "inner loop-1" is $O(d_{max})$ if we consider all nodes (vertices) possess maximum degree. The complexity of computing a single $R_{w_2}(u \dashrightarrow v)$ for given vertices u and v in the "inner loop-2" of *2nd order FPNC* algorithm is $O(d_{\max})$. The complexity of the "inner loop-2" is then $O(d^2_{\max})$, and hence, the total time complexity of the "outer loop" is $O(nd^2_{\max}) \ll O(n^3)$. Since PPI networks possess power-law property, the majority of the proteins interact with only very few proteins. So, we can use the average degree $\bar{d}$ instead of d_{max} and the average degree $\bar{d}$ is generally small which can be considered as constant [12]. In community detection phase, *2nd order FPNC* requires $O(m)$ times which is $O(n\bar{d}^2) > O(m)$. As such, then the complexity of *2nd order FPNC* algorithm is $O(n\bar{d}^2)$. It also indicates that after introducing the 2nd neighbour concept in *FPNC* algorithm, the computation time complexity does not alter.

4 Results and Discussion

We implemented our *2nd order FPNC* algorithm on *Saccharomyces cerevisiae* (Baker's yeast). The protein-protein interaction network of baker's yeast of 5697 proteins and 50675 interactions was downloaded from BioGrid database[1]. PPI network of baker's yeast was an unweighted network. So, we assigned confidence score to the interactions using logarithmic regression based scheme of [10] to convert the unweighted PPI network to weighted PPI network.

We tested our algorithm on both weighted and unweighted PPI networks of baker's yeast. We carried out the experiments by following several steps. At first we clustered PPI network using *2nd order FPNC*, *FPNC*[9], *FAC-PIN*[8], *HC-PIN*[12] and *CNM*[1] algorithms. Then we mapped the clustered output of each algorithm into the original functional modules of baker's yeast. Functional module mapping was performed for individual ontology (*Biological Process(BP)*, *Molecular Function(MF)* and *Cellular Component(CC)*) and all ontologies together. In third step, we also analyzed the efficiency of *2nd order FPNC* algorithm by measuring modularity Q [6], $w\text{-}\log\text{-}v$ [5]. In functional module mapping phase, we considered the functional modules of *Yeast Genome* database[2] as gold standard. All functional module mapping and efficiency analysis results are shown in next subsequent sub sections.

4.1 Functional Module Mapping

We carried out the functional module mapping using a variable called *P-value*[9, 12] which was calculated from cumulative distribution function *cdf* of *hypergeometric distribution.*

$$P\text{-}value = 1 - \sum_{j=0}^{k-1} \frac{\binom{|F_i|}{j}\binom{|V|-|F_i|}{|M|-j}}{\binom{|V|}{|M|}} \tag{6}$$

[1] downloaded from `http://thebiogrid.org` version 3.2.97 accessed on August 2013

[2] `http://downloads.yeastgenome.org/curation/literature/go_slim_mapping.tab` accessed on July 2013

Where F_i is a functional category mapped to module M. The proteins in functional category F_i are physical functional modules, the proteins in module M are computer generated clusters and the common proteins of F_i and M are considered as true predictions k. It is understood as the probability that at least k proteins in M are included in F_i. Lower *P-value* indicates that the cluster closely corresponds to the functional module, because the network has a lower probability to produce separate community by chance [4].

To detect significant modules, a term called *cutoff* value had been used. If the *P-value* of a community is less than cutoff value, we considered it as significant module. This process was also used by both Wang et al.[12] and Rahman et al.[8, 9] for validating their *HC-PIN* and *FAC-PIN* algorithms. Wang et al.[12], Rahman et al.[8, 9] and we considered the cutoff value is 0.05. Besides the *P-value*, we also used another two important aspects- *Recall* and *Precision* to estimate the performance of the algorithms for detecting functional modules [12]. Recall is the fraction of the true prediction out of all known modules and *Precision* is the fraction of true prediction out of all communities (computer generated clusters)[12]. The formulas of Recall and Precision follow as below-

$$Recall = \frac{|M \cap F_i|}{|F_i|} \tag{7}$$

$$Precision = \frac{|M \cap F_i|}{|M|} \tag{8}$$

Recall indicates how effectively proteins with the same functional category in the network are extracted and *Precision* illustrates how consistently proteins in the same module are annotated [12]. Thus, a harmonic mean of *Recall* and *Precision*, *f-measure* is defined as follow-

$$f\text{-measure} = \frac{2 \times Recall \times Precision}{Recall + Precision} \tag{9}$$

The accuracy of an algorithm is defined as the average of *f-measure* of the significant modules generated by the algorithm [12]. In the experiment, we considered the communities/ clusters should have three or more proteins which was also considered by Wang et al.[12] and Rahman et al.[8, 9]. The results of functional modules validation are shown in Table 1. In Table 1, k_p, k_s, $\overline{-\log P}$ and $\overline{fm}$ shows the number of communities identified by algorithms, the number of significant modules, average - log(*P-value*) and average f-measure respectively.

CNM algorithm is only designed for unweighted PPI network. So we were unable to compute *CNM* algorithm for weighted PPI network of baker's yeast and put blank in the columns 7-10. In Table 1, it is found that *2nd order FPNC* algorithm identified more proportionate significant modules than other algorithms. Average - log(*P-value*) of the *2nd order FPNC* algorithm is lower than other which indicates it produced almost accurate communities/ modules. The average f-measure of *2nd order FPNC* algorithm is higher than other, means *2nd order FPNC* algorithm outperforms *FPNC*, *FAC-PIN*, *HC-PIN* and *CNM* algorithms

Table 1. Functional enrichment of the identified Modules which comprises of three or more proteins

Algo	Ont	Unweighted PPI				Weighted PPI			
		k_p	k_s	-log P	fm	k_p	k_s	-log P	fm
2nd order FPNC	All	413	198	2.83	0.27	418	214	3.74	0.46
	BP		187	2.51	0.22		188	3.67	0.51
	CC		187	2.73	0.29		187	3.92	0.48
	MF		188	2.97	0.24		187	3.55	0.42
FPNC	All	432	217	3.06	0.21	398	253	4.18	0.41
	BP		203	3.11	0.28		216	4.22	0.32
	CC		203	2.83	0.15		216	4.02	0.42
	MF		203	2.99	0.11		221	4.14	0.39
FAC-PIN	All	522	241	4.66	0.16	522	276	5.59	0.39
	BP		168	4.14	0.17		203	4.98	0.27
	CC		168	4.52	0.13		197	5.67	0.42
	MF		168	3.94	0.11		218	5.43	0.36
HC-PIN	All	159	72	3.17	0.02	353	176	4.28	0.33
	BP		61	2.52	0.03		118	4.52	0.34
	CC		61	3.09	0.01		152	3.97	0.29
	MF		61	3.21	0.02		109	4.12	0.32
CNM	All	226	19	4.45	0.03	-	-	-	-
	BP		12	3.93	0.04		-	-	-
	CC		12	4.33	0.04		-	-	-
	MF		12	3.90	0.03		-	-	-

in respect to accuracy. Table 1 shows *2nd order FPNC* algorithm outsails other three algorithms for finding functional modules experimentally.

4.2 Efficiency Analysis

Given a clustering result (i.e. a partition) $P_k = \{C_1, \ldots, C_k\}$ with k clusters, we used the popular scoring functions: *modularity measure*, introduced by Newman [6] and modified by Rahman et al. [9] for both weighted and unweighted PPI networks, defined as

$$Q_w(P_k) = \sum_{i=1}^{k} \left(\frac{\sum_{\forall \{v,u\} \epsilon \bar{G}_i} w(u,v)}{\sum_{\forall \{a,b\} \epsilon G} w(a,b)} - \left(\frac{\sum_{\forall \{v,a\} \epsilon G \ and \ v \epsilon \bar{G}_i} w(v,a)}{\sum_{\forall \{a,b\} \epsilon G} w(a,b)} \right)^2 \right) \tag{10}$$

Here $\bar{G}_i$ is the cluster i, k is the number of clusters of a graph G. But modularity measure faces resolution problem. For this reason, we also used another scoring function which was known as $w\text{-}\log\text{-}v$ and proposed in [5]. Later Rahman et al.[9] modified it for both weighted and unweighted PPI networks.

$$w\text{-}\log\text{-}v = \sum_{i=1}^{k} \left(\frac{\sum_{\forall \{v,u\} \epsilon \bar{G}_i} w(u,v)}{\sum_{\forall \{a,b\} \epsilon G} w(a,b)} \times \log \frac{\sum_{\forall \{v,a\} \epsilon G \ and \ v \epsilon \bar{G}_i} w(v,a)}{\sum_{\forall \{a,b\} \epsilon G} w(a,b)} \right) \tag{11}$$

Here $\bar{G}_i$ is the cluster i, k is the number of clusters in a graph G.

As said above, we ran *2nd order FPNC* algorithm then evaluated the modularity structure of the communities obtained for six different (containing at least more than 3000 proteins and 34000 interactions) species which are shown in Table - 2.

We also implemented the best-performing *FPNC*[9], *FAC-PIN*[8], *HC-PIN*[12] and *CNM*[1] algorithms. The *FPNC*, *FAC-PIN*, *HC-PIN* and *CNM* methods

Table 2. Dataset used in evaluating *FAC-PIN*, *HC-PIN* and *CNM* algorithms

Species	**Scientific name**	**Proteins**	**Interactions**
Baker's yeast	*Saccharomyces cerevisiae*	5697	50675
Frog	*Xenopus Tropicalis*	5473	122706
Human	*Homo sapiens*	8997	34935
Rice	*Oryza sativa*	3778	320570
Boar	*Sus Scrofa*	5303	119920
Fish	*Danio rerio*	8188	274358

Table 3. Scoring functions: Q results of *FAC-PIN*, *HC-PIN* and *CNM* for different species

Algorithm	Unweighted PIN						Weighted PIN					
	Yeast	Boar	Frog	Human	Fish	Rice	Yeast	Boar	Frog	Human	Fish	Rice
2nd order FPNC	**0.602**	**0.811**	**0.796**	**0.825**	**0.780**	**0.783**	**0.983**	**0.981**	**0.974**	**0.986**	**0.979**	**0.932**
FPNC	0.511	0.753	0.743	0.782	0.769	0.740	0.976	0.973	0.951	0.986	0.976	0.916
FAC-PIN	0.436	0.704	0.431	0.592	0.521	0.591	0.866	0.753	0.763	0.896	0.910	0.897
HC-PIN	0.141	0.674	0.229	0.285	0.229	0.521	0.897	0.936	0.927	0.543	0.621	0.837
CNM	0.038	0.518	0.752	0.012	0.752	0.179	-	-	-	-	-	-

Table 4. Scoring functions: - (w- log -v) results of *FAC-PIN*, *HC-PIN* and *CNM* for different species

Algorithm	Unweighted PIN						Weighted PIN					
	Yeast	Boar	Frog	Human	Fish	Rice	Yeast	Boar	Frog	Human	Fish	Rice
2nd order FPNC	**1.486**	**3.011**	**2.708**	**2.099**	**2.286**	**2.022**	**3.477**	**6.852**	**6.651**	**6.531**	**3.997**	**5.129**
FPNC	1.327	2.634	2.525	1.849	1.778	1.615	3.063	6.053	5.167	5.932	3.267	4.555
FAC-PIN	1.301	1.561	2.021	1.518	0.718	1.149	2.835	6.138	4.617	1.972	2.762	3.751
HC-PIN	1.299	1.805	1.809	0.760	0.997	1.585	2.672	5.967	3.223	1.908	1.288	3.641
CNM	1.299	1.530	1.699	1.283	0.516	0.236	-	-	-	-	-	-

had been run on the same PPI network data that used in *2nd order FPNC* approach. For *HC-PIN*, we set the parameter λ (0.5, 1.0, 1.5, 2.0, 2.5 and 3.0) as in [12]. For *FAC-PIN*, we selected the five different values for α (0.5, 0.25, 0.125, 0.0625 and 0.03125) mentioned in [7], whereas CNM has no parameters. Scoring functions results of the five methods are given in Tables 3 and 4

As we see in Tables 3 and 4, *2nd order FPNC* algorithm outperformed *FPNC*, *HC-PIN*, *FAC-PIN* and *CNM* methods in respect to Q and w- log -v.

5 Conclusion

In the paper, we modified the *FPNC* algorithm by introducing the concept of second neighbours for detecting functional modules which made the algorithm more efficient and robust. *2nd order FPNC* algorithm can find functional modules more accurately than other four comparisons. Our *2nd order FPNC algorithm* has shown an excellent improvement in the result of finding functional modules both theoretically and experimentally. The idea of introducing the concept of second neighbours gives us more compact clusters resulting into better average *f-measure* means higher accuracy and lower average - log(P-*value*) which gives us less chance of misclassification. There is still an open window for modifying *2nd order FPNC* algorithm by introducing the concept of cluster density, separability, conductance etc. to improve its efficiency even more. The concept of shortest path technique or network kernel can also be implemented to make it more efficient and effective. We have a plan to continue working on modifying *2nd order FPCN* algorithm by introducing network kernel to improve its functionality. Besides, we have another plan to use resampling techniques to perform several executions of each algorithm. While performing several executions of each compared method, statistical tests can be used to extract sound conclusions with a statistics base. We have implemented our algorithm to detect functional modules from any protein-protein interaction network. But a protein complex performs a specific task than functional modules. We want to perform protein complex validation testing for the results of our 2nd order FPNC algorithm to find its working efficiency over protein complexes.

Acknowledgment

We would like to give special thanks to Dr. Alioune Ngom and his student Mohammad S. Rahman for helping us about *FAC-PIN* algorithm and research concept. Dr. Aron Clauset and Dr. Min Li also helped us regarding their designed *CNM* and *HC-PIN* algorithms respectively.

References

1. Clauset, A., Newman, M. E. J., Moore, C.: "Finding community structure in very large networks", In *Nature*, Vol. 453, No. 7191, 2005.
2. Fortunato, S.: "Community detection in graphs", In *Elsevier Physics Reports*, Vol. 486, 2010.
3. Girvan, M., Newman, M.E.: "Community Structure in Social and Biological Networks", In *Proceedings of Natural Academy of Science USA*, Vol. 99, 2002.
4. Kenley, E.C., Cho, Y.-R.: "Detecting protein complexes and functional modules from protein interaction networks: A graph entropy approach", In *Proteomics*, Vol. 11, pp. 1116-1121, 2011.
5. Laarhoven, T.V., Marchiori, E.: "Robust Community Detection Methods with Resolution Parameter for Module Detection in Protein Protein Interaction Networks", In *Springer-Verlag Berlin*, Vol. 7632, 2012.
6. Newman, M.E.J.: "Fast algorithm for detecting community structure in networks", In *Physical Review*, Vol. 69, No. 066133, 2003.
7. Rahman, M.S., Ngom, A.: A Fast Agglomerative Community Detection Method for Protein Complex Discovery in Protein Interaction Networks, In *8th IAPR International Conference on Pattern Recognition in Bioinformatics*, 2013.
8. Rahman, M.S., Ngom, A.: Relative Vertex-to-Vertex Clustering Value — A New Criterion for the Fast Detection of Functional Modules in Protein Interaction Networks, In *9th International Symposium on Bioinformatics Research and Applications (ISBRA), USA*, 2013.
9. Rahman, M.S., Islam, M.S., Chowdhury, N., Karim, M.M.: Fast Protein Network Clustering Algorithm: A New Approach for Clustering Protein-protein Interaction Networks to Detect Functional Modules, In *2nd International Conference on Advances in Electrical Engineering (ICAEE), Bangladesh*, 2013 (Accepted).

10. Sharan, R., Suthram, S., Kelley, R.M., Kuhn, T., McCuine, S., Uetz, P., Sittler, T., Karp, R.M., Ideker T.: "Conserved Pattern of Protein Interaction in Multiple Species", In *Proceedings of Natural Academy of Science, USA*, Vol. 102, No. 6, 2005.
11. Spirin, V., Mirny, L.A.: "Protein Modulees and Functional Modules in Molecular Networks", In *Proceedings of Natural Academy of Science USA*, Vol. 100, No. 21, 2007.
12. Wang, J., Li, M., Chen, J., Pan, Y.: "A Fast Hierarchical Clustering Algorithm for Functional Modules Discovery in Protein Interaction Networks", In *IEEE/ACM Transaction on Computational Biology and Bioinformatics*, Vol. 8, No. 3, 2011.