PCD-GED: Protein complex detection considering PPI dynamics based on time series gene expression data

Amir Lakizadeh a, Saeed Jallili a, *, Sayed-Amir Marashi b

a Computer Engineering Department, Tarbiat Modares University, Tehran, Iran
b Department of Biotechnology, College of Science, University of Tehran, Tehran, Iran

HIGHLIGHTS

• A novel method for detecting protein complex is proposed.
• It uses gene expression data to model dynamicity of PPI networks.
• It generates a series of subnetworks according to each column of expression data.
• Detection in every subnetwork is based on the weighted clustering coefficient values.
• It finds protein complexes with high weighted density core.

ARTICLE INFO

Article history:
Received 26 October 2014
Received in revised form
14 April 2015
Accepted 17 April 2015
Available online 29 April 2015

Keywords:
Protein–protein interaction
Protein complex
Clustering coefficient
Weighted density

ABSTRACT

Detection of protein complexes from protein–protein interaction (PPI) networks is essential to understand the function of cell machinery. However, available PPIs are static, and cannot reflect the dynamics inherent in real networks. Our method uses time series gene expression data in addition to PPI networks to detect protein complexes. The proposed method generates a series of time-sequenced subnetworks (TSN) according to the time that the interactions are activated. It finds, from each TSN, the protein complexes by employing the weighted clustering coefficient and maximal weighted density concepts. The final set of detected protein complexes are obtained from union of all complexes from different subnetworks. Our findings suggest that by employing these considerations can produce far better results in protein complex detection problem.

1. Introduction

Protein complexes are key cellular entities in cellular organization and function. They are as building blocks for many biological processes. Therefore, detecting these complexes can provide a better understanding of basic components and organization of cell machinery from a system level point (Gavin et al., 2002; Barabasi and Oltvai 2004, Gavin et al., 2006; Srihari and Leong 2012). Experimental methods such as TAP-MS and co-immunoprecipitation (Bader and Hogue 2002) for protein complex detection are both expensive and time-consuming (Von Mering et al, 2002). Furthermore, in TAP-MS, transient low-affinity protein complexes may not be detected (Gavin et al, 2002). In the past decade, one recent advance in high-throughput experimental techniques that are complemented by computational methods, resulting in a large amount of protein–protein interaction (PPI) data. These networks make it possible to develop computational methods for protein complex prediction. So, computational methods can be considered as an alternative to find protein complexes (Li et al., 2010).

Detection of protein complexes of PPI networks plays a significant role in unfolding the structure of PPI networks, predicting protein functions and explaining particular biological processes. It is essential not only in understanding complex formations, but also to find its higher level cellular organization. Clustering is the main approach to detect protein complexes from PPI networks and it is defined as categorizing data objects into groups (clusters) such that objects in each cluster are more inter-similar compared to objects from other clusters.

Over the past decade, many computational methods have been proposed for clustering PPI networks (Li et al., 2010, Srihari and Leong 2013). Although PPIs are constructed based on physical contact between proteins, it does not mean that all possible interactions occur in any cell at any time (Chen et al., 2014). The cellular systems are highly dynamic and PPI networks are changing over time, environments and different stages of the cell cycle.
computational methods that are presented to detect protein complexes. For example, many computational methods focus on detecting highly-connected subgraphs in PPI networks and ignore their inherent core-attachment organization (Srihari and Leong 2013). Another problem is the considerable number of false positive and false negative interactions, i.e. noise, in current PPI networks. It is obvious that using weighted networks in which each interaction is scored by a “reliability” value, one can reduce the effect of this noise.

PCD-GED, as we rolled the name for our proposed method, begins with computing a simple active threshold for each gene, based on its mean of expression profile. The algorithms continues with generating a series of subnetworks of active proteins according to each column of gene expression matrix. PCD-GED then, finds protein complexes in every subnetwork in two phases. In the first phase, it computes the weighted clustering coefficient value for each protein and based on a threshold, it selects a seed protein for each core. It finds the complex core by expanding the seed. Expanding a core is continued (in a greedy approach) until no increment in the weighted density of core is obtained. In the second phase, each core complex is grown by addition of attachment proteins. The final set of detected protein complexes are obtained after removing redundancies from unions of all complexes derived from different subnetworks. Experimental results show that exploiting this dynamicity, and searching the cores with maximal weighted density, enables PCD-GED to improve the accuracy of protein complex detection. Hence, it is a more reliable method for protein complex detection. A list of symbols used in the paper is presented in Appendix 1.

2. Material and methods

2.1. Datasets and evaluation measures

To evaluate the proposed method, PCD-GED is compared with some recent methods. The PPI network, DIP (Xenarios et al., 2002) and gene expression data, GED1 are used in this comparison. A more thorough comparison between PCD-GED and the best of previous methods, TSN-PCD is performed on “Consolidated” (Collins et al., 2007) PPI network and gene expression data, GED2 (Pramila et al., 2007). The properties of all used networks are shown in Table 1. We compute a simple active threshold for each gene in expression matrix. Based on these thresholds, we generate time-sequence subnetworks; i.e. for each protein in a time point, if its gene expression value is greater than its active threshold, it is selected (as active) from static PPI network. Time-point subnetwork for each column is a subgraph of static PPI network that consists of all active proteins of that specific time. Note that the used gene-expressing profiles must well-cover proteins of static PPI network. In order to evaluate the detected

Table 1
properties of used datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consolidated</td>
<td>PPI network</td>
<td>9073 interactions between 1622 proteins</td>
</tr>
<tr>
<td>DIP</td>
<td>PPI network</td>
<td>21,788 interactions between 4950 proteins</td>
</tr>
<tr>
<td>GED1 (Tu, Kudlicki et al., 2005)</td>
<td>Gene expression</td>
<td>Expression of 6777 genes in 36 time points</td>
</tr>
<tr>
<td>GED2 (Pramila, Wu et al., 2007)</td>
<td>Gene expression</td>
<td>Expression of 4774 genes in 25 time points</td>
</tr>
</tbody>
</table>
protein complexes, we use 408 known protein complexes as benchmark from CYC2008 (Pu et al., 2009).

2.2. The PCD-GED main algorithm

Algorithm 1 indicates the main function of PCD-GED. Inputs of algorithm are a weighted PPI network, G(V,E), and a matrix of gene expression values. The matrix must well-cover the proteins in G. PCD-GED, has four pareametr: α, β, γ and δ that are cut, seed, set and refinement parameters repsectively. The cut parameter α is used to generate an active threshold for each gene in expression matrix. The larger value for α parameter causes to have fewer active proteins in every subnetwork and as a result we have less interactions in each subnetwork. The seed parameter β is used in SeedGeneration sub-function in order to generate initial seeds for the cores of protein complexes. By tunning β, we can control the number of predicted complexes such that the larger value for β parameter leads to fewer protein complexes. Also, the set parameter γ, is used as a coefficient in CoreExpanding sub-function to determine whether a neighbor protein belongs to current predicted core. The larger value for γ parameter leads to detect smaller complexes. The refinement parameter δ is used in Refinement sub-function to merge very similar predicted complexes. Here, we explain the main function of PCD-GED, Algorithm 1, in more details.

In lines 2–3, the mean of expression values for each protein in expression matrix M is computed as m. In an iteration (line 4), TSNi is extracted for each time point (column) Ti of the expression matrix in the first step (line 4.1). TSNi consists of active proteins at the time Ti. Recall that a protein is selected as active in time point Ti if its expression value is greater than α × m, where α is the cut parameter. Then in three steps (lines 4.2–4.4) the protein complexes in TSNi, “Complexes”, are detected. In line 4.2, Seeds are found. The concept of weighted density of a subgraph is used to expand the seeds to determine the candidate cores (line 4.3). Note that each candidate core is a high weighted density subgraph around the seed. The protein complexes are found by addition of attachment proteins to the cores (line 4.4). At the end of loop (line 4.5) current detected protein complexes are accumulated in “Complexes”, the set of all recently detected protein complexes. Some resulting complexes from step 4 may have a high level of similarity; for example a complex may be detected from several subnetworks. So, we apply a refinement procedure in order to filter out redundencies and generate the final protein complexes set.

Algorithm 1. PCD-GED main function

Input: G(V, E): A weighted (static) PPI network, M: matrix of gene expression levels of proteins that well-cover proteins in G and α, β, γ : cut, seed, and set parameters

Output: Complexes: the detected protein complexes
1. Complexes ← an empty set
2. for each row i in M (as protein pi) do
3. set m as mean of all expression values of pi in all time points (columns)
4. for each time point Ti in M do
5. TSNi ← subgraph of G consists of active proteins pi in Ti that its expression is greater than (α × m)
6. 2. Seedsi ← SeedGeneration(TSNi, β)
7. 3. Coreis ← CoreFinding(TSNi, Seedsi)
8. 4. Complexesi ← CoreExpanding(TSNi, Coreis, γ)
9. 5. Complexes ← Complexes ∪ Complexesi
10. Complexes ← Refinement(Complexes)
2.3. Seed generation

“SeedGeneration” sub-function tries to candidate a protein seed for each complex core. It is based on the assumption that each protein complex has a core that characterizes its main function. In proposed method, we suppose that each complex core grows around a central protein as its seed. It reads a weighted PPI subnetwork TSNj(Vj, E) and a coefficient β as inputs and generates a set of proteins as seeds. It computes \( wccj(pi) \), the weighted local clustering coefficient, defined in eq. (1) for each vertex pi in the subnetwork TSNj(Vj, E).

\[
\text{wccj}(pi) = \sum e \in E(pi), \frac{(w(e))/|Nj| \times (Nj - 1)}
\]

In Eq. (1), E(pi) is the set of all edges between neighbors of pi, w(e) is the weight of edge e, Nj is the number of neighbors of pi, Vj is the set of vertices (proteins) in TSNj. Those vertices pi, from TSNj, which their \( wccj(pi) \) values are greater than β are considered as preliminary protein seeds.

2.4. Core finding

Based on observations, each complex core includes several proteins to perform key function of complex. “CoreFinding” sub-function employs weighted density to find functional cores around the seeds. In “CoreFinding” sub-function (Algorithm 2), Cores (the final cores to be returned) is set to an empty set (line 1). In an iterative procedure, (line 2–11) for each seed protein pi, the core ci is initialized to pi (line 2). In line 3, the induced ci sub graph, ICi, is computed. Induced sub graph ICi is the direct neighborhood subgraph of pi. By calling “MaxWdeg” sub-function (Algorithm 3), in line 4; a vertex in ICi with the largest sum of weighted interactions with vertices in current core ci is selected as pj. Next, through an inner loop, (lines 6-9) while there is such pj in ICi that its addition to ci cannot decrease the weighted density, it is added to the core. The “Wdensity” sub-function computes the weighted density of the subgraph consisting [ci ∪ pj] vertices by Eq. (2). Detected core, cj is added to Cores in line 10. The preliminary cores detected up to now may be the same regarding to an isomorphism on network topology. Hence, redundant protein cores are removed (in line 12). Note that according to the greedy characteristic of “CoreFinding” sub-function, it does not necessarily leads to an optimal weighted dense subgraph; instead it determines a subgraph IC with high weighted density.

The weighted density (DW) of a subgraph IC in G is computed by Eq. (2)

\[
\text{DW} = \sum e \in E(w(e)/|V| \times (|V| - 1))
\]

Where |V| is the number of vertices in G.

Algorithm 2. CoreFinding sub-function

Input: G(V,E); A weighted PPI network, Seeds: the set of candidate proteins as seeds of cores

Output: Cores: the set of detected protein cores
1. Cores ← an empty set
2. for each protein seed pi ∈ Seeds do
3. ci ← pi // an initial core
4. ICi ← induced sub graph of pi
5. pj ← MaxWdeg(G, ICi)
6. while there is pj and Wdensity(G, [ci ∪ pj]) > Wdensity(G, ci) do
7. ci ← ci ∪ pj
8. pj ← MaxWdeg(G, ICi)
9. end while
10. Cores ← Cores ∪ ci
2.5. Core expanding

After core detection for each protein complex in the previous steps, “CoreExpanding” sub-function (Algorithm 4), adds the attachment proteins to the core to construct a protein complex. In this Algorithm, InitComplexes, the set of protein complexes that must be returned, is initialized to an empty set (line 1). In the main loop (lines 2-11), \( L \), the set of all neighbors of protein in \( c_i \) is obtained by iteration (lines 3-7). Within another iteration (lines 8-11) \( \gamma \) is set to all neighbors of \( p_x \), for each proteins \( p_j \) of \( L \). If \(|c_i \cap \gamma| \geq \gamma, \) then we add \( p_x \) as an attachment to \( c_i \). In line 12, we add the updated core \( c_i \) to InitComplexes. Note that as expected each protein can belong to more than one protein complex, which means that overlapping complexes are allowed to be found.

Algorithm 3. MaxWdeg sub-function

Input: \( G(V,E) \): A weighted PPI network, \( c_i \): currently detected core, \( IG \): the induced subgraph of seed of \( c_i \).
Output: \( px \): a vertex in \( IG \) with the largest sum of weighted interactions with vertices in current core \( c_i \)
1. Cmax\( \leftarrow 0 \)
2. index\( \leftarrow -1 \)
3. for each protein \( p_x \) in \( IG \)
4. if \( c_i \) is not consist \( p_x \)
5. cur\( \leftarrow 0 \)
6. for each protein \( p_x \) in \( c_i \)
7. cur\( \leftarrow cur+weight \) of interaction between \( p_x \) and \( p_y \)
8. end for
9. if cur > Cmax
10. Cmax\( \leftarrow cur \)
11. index\( \leftarrow p_x \)
12. end if
13. end if
14. end for

Algorithm 4. CoreExpanding sub-function

Input: \( G(V,E) \): A weighted PPI network, Core, the detected complexes cores, \( \gamma \): set threshold
Output: InitComplexes: the set of protein complexes
1. InitComplexes\( \leftarrow \) an empty set.
2. for each core \( c_i \) \( \epsilon \) Core do
3. \( L \leftarrow \) an empty set
4. for each protein \( p_x \) \( \epsilon \) \( c_i \) do
5. \( N_i \leftarrow \) the set of all direct neighbors of \( p_i \)
6. \( L \leftarrow L \cup N_i \)
7. end for
8. for each protein \( p_j \) \( \epsilon \) \( L \) do
9. \( N_j \leftarrow \) the set of all direct neighbors of \( p_j \)
10. if \(|c_i \cap N_j| \geq \gamma,|L|\) then \( c_i \leftarrow c_i \cup p_j \)
11. end for
12. InitComplexes\( \leftarrow \) InitComplexes \( c_i \)
end for

2.6. Refinement

Since some of the predicted protein complexes may have a large commonality, “Refinement” sub-function (Algorithm 5) is used to integrate highly similar (overlapping) protein complexes. Complexes, the final protein complexes to be returned, is initialized to InitComplexes, all recent complex detected (line 1). In line 2, Sim is set as an \( n \times n \) lower triangular overlapping matrix with zero values above diagonal (\( n \) is the number of complexes in input set InitComplexes). In lines 3-5, for each element \( sim_i \) that \( i<j \), the similarity score of complexes \( C_i \) and \( C_j \) is computed as eq. (3) (Srihari and Leong 2013).

\[
sim_{ij} = |C_i \cap C_j|/(|C_i \cup C_j|)
\]  

Next, the all Complexes in InitComplexes are marked as unvisited in line 6. Max is defined as the maximum value in Sim which is located in row \( i \) and column \( j \) which \( C_i \) and \( C_j \) are not visited (line 7). Through an iteration (lines 8-14) while Max > \( \delta \), either \( C_i \) or \( C_j \) which has smaller size is marked as visited and deleted from Complexes. This loop will be repeated until the maximum similarity value of SimMx gets no larger than the refinement threshold \( \delta \). We observed that different settings for \( \delta \) have a little effect on \( f_1 \)-measure. However, the best result is achieved by \( \delta =0.7 \), so, we used this value as default.

Algorithm 5. Refinement sub-function

Input: \( G(V,E) \): InitComplexes, the initial detected protein complexes, \( \delta \): refinement parameter
Output: Complexes: the final protein complexes to be returned.
1. Complexes\( \leftarrow \) InitComplexes.
2. Sim\( \leftarrow \) an \( n \times n \) lower triangular overlapping matrix with zero values above diagonal.
3. for each pair complex \( C_i \) and \( C_j \) in InitComplexes do\( / i<j \)
4. \( sim_{ij}\)\( \leftarrow \) similarity score of complexes \( C_i \) and \( C_j \) as eq. (3)
5. end for
6. mark all complexes in InitComplexes as unvisited.
7. Max\( \leftarrow \) maximum value in Sim which is located in row \( i \) and column \( j \) which \( C_i \) and \( C_j \) are not visited
8. while Max > \( \delta \)
9. \( C_i \)\( \leftarrow \) Either \( C_i \) or \( C_j \) which has smaller size
10. Mark \( C_i \) as visited.
11. Delete \( C_i \) from Complexes
12. Max\( \leftarrow \) maximum value in Sim which is located in row \( i \) and column \( j \) which \( C_i \) and \( C_j \) are not visited
13 end while

3. Experiments and results

In this section, we first introduce the evaluation metrics and parameter settings. Then, we present detailed experimental results. Next, PCD-GED is evaluated by additional analysis. Comparison of PCD-GED with other methods and PCD subroutine without gene expression data shows the effect of expression profile in true prediction of protein complex. The proposed method is simply based on the fact that proteins in the true protein complex must be simultaneously expressed in one cell. The consistency of gene expression pattern supports the true estimation of protein complexes. Especially, using the time series data of gene expression is an appropriate approach from the viewpoint of the temporal resolution.

3.1. Evaluation measures

Precision, recall and \( f_1 \)-measure are commonly-used evaluation metrics in information retrieval and machine learning. In the context of protein complex detection, precision measures the fraction of predicted complexes that matches any real complex. Recall, is also the fraction of real complexes that can be detected correctly. Let \( R = \{r_1, r_2, \ldots, r_n\} \) denotes the set of real (benchmark)
protein complexes and \( D = \{ d_1, d_2, ..., d_n \} \) indicates the set of detected complexes. To assess a method, we need to define how effectively a predicted protein complex matches a real one. According to the literature, the overlapping score \( OS(d_i, r_j) \) (Bader and Hogue 2003) is used to determine the overlap between a predicted complex \( d_i \) and a real benchmark complex \( r_j \).

\[
OS(d_i, r_j) = \frac{|d_i \cap r_j|^2}{|d_i||r_j|}
\]

(4)

Let \( \delta > 0 \) be a predefined least similarity threshold; such that \( OS(d_i, r_j) \geq \delta \) implies that \( d_i \) and \( r_j \) are matched. Typically, \( \delta \) is set to 0.2 and precision \( (Pr) \) and recall \( (Re) \) are defined according to Eqs. (5) and (6) respectively (Gavin et al., 2002; Jin, McCallen et al., 2009; Tang et al., 2011).

\[
Re = \frac{|\{ r_i \in R: \exists d_j \in D; OS(r_i, d_j) > \delta \}|}{|R|}
\]

(5)

\[
Pr = \frac{|\{ d_j \in D: \exists r_i \in R; OS(d_j, r_i) > \delta \}|}{|D|}
\]

(6)

\( f_1 - \text{measure} \) (defined in Eq. 7) is basically the harmonic mean of precision and recall, and it is used as an overall measure of performance.

\[
f_1 - \text{measure} = 2 \times Pr \times Re / (Re + Pr)
\]

(7)

### 3.2. Parameter sensitivity analysis

In order to assess the effect of cut and seed parameters \( \alpha, \beta \), we run PCD-GED with different settings for \( \alpha \) and \( \beta \) parameters and compute precision, recall and \( f_1 - \text{measure} \) in each case. Our method is capable of maximizing precision or recall on demand by fine tuning \( \alpha, \beta \) parameters. Figs. 1–3 illustrate the effect of different values for parameters on above measures. For example as shown in Fig. 3, the best result based on \( f_1 - \text{measure} \) is obtained for \( \alpha = 0.8 \) and \( \beta = 0.3 \). Hence, we suggest these values as defaults. Note that in all experiments, we fix \( \gamma \), the set threshold to 0.5.

### 3.3. Comparison with other methods

Here, we compare PCD-GED (considering its best parameter setting) with several recent dynamic methods proposed in the literature, e.g. TSN-PCD (Li et al., 2012), TC-PIN (Tang et al., 2011), APPIN (Wang et al., 2013) and NF-APIN (Xiao et al., 2013). In all these experiments, the same input and benchmark datasets to ours are used to compute precision, recall and \( f_1 - \text{measure} \) values. Experimental results are shown in Fig. 4. In cases of TC-PIN, APPIN and NF-APIN, the results are reported from (Xiao et al., 2013) and for TSN-PCD, the values are computed based on authors reported output complex set (Li et al., 2012). Note that in these researches, precision and recall are reported as sensitivity and specificity respectively.

As Fig. 4 shows, in PCD-GED, (w.r.t its best parameter settings) \( f_1 - \text{measure} \) has 29% improvement compared to TSN-PCD, the best of other methods. In order to make our analysis precise to the best of our effort, we compared PCD-GED with TSN-PCD on “Consolidated” PPI network and gene expression data, GED2. The results (Fig. 5) confirm the superiority (47% improvement) of PCD-GED.

These experimental results proved that taking our contributions into account in PCD-GED, can improve the accuracy of protein complex detection.

### 3.4. Analytical discussion

We discuss the effect of our main contributions, i.e. employing gene expression data (dynamic detection of protein complexes) and using weighted clustering coefficients to find complex seed.

We try to examine the effect of these features by switching them on and off. In order to evaluate the effect of employing dynamic considerations, we ran PCD subroutine on static PPI network only (without gene expression data) and compared it with PCD-GED. The results (Fig. 6) show that using adaptability can significantly increase both precision and recall with more effect on recall. This implies that PCD-GED has 27% improvement based on \( f_1 - \text{measure} \) compared with PCD. Indeed, the results confirm that running a detection method on different sub-networks with distinct topologies that are derived from one static PPI network based on gene expression data can retrieve extra real protein complexes. Therefore, it causes the method has more recall.

Next, to evaluate the effect of using weighted clustering coefficients values, we ran PCD-GED in two modes: with and without “SeedGeneration” sub-function. Note that in the case “SeedGeneration” is absent, all of proteins in PPI sub-networks are selected as seeds. So, it causes PCD to predict lots of complexes which precision is fallen. The results (Fig. 7) based on “DIP”, GED1 datasets and CYC2008 as benchmark, confirm using weighted clustering coefficients significantly increases the precision with negligible change in running time. It causes 17.5% improvement on \( f_1 - \text{measure} \).
Fig. 4. NF-APIN (Xiao et al., 2013), APPIN (Wang et al., 2013), TC-PIN (Tang et al., 2011), TSN-PCD (Li et al., 2012) and PCD-GED methods comparison (considering their best parameter values) based on DIP (Xenarios, Salwinski et al., 2002), PPI network and GED1 (Tu, Kudlicki et al., 2005), gene expression data. Best parameters setting for PCD-GED are $\alpha = 0.8$, $\beta = 0.3$. In cases of NF-APIN, APPIN and TC-PIN, the results are reported from (Xiao, Wang et al., 2013) and in the case of TSN-PCD, the values are computed based on authors reported output complex set (Li, Wu et al., 2012).

Fig. 5. PCD-GED in comparison to TSN-PCD on “Consolidated” PPI network and gene expression data, GED2.

Fig. 6. PCD-GED in comparison to PCD. Results show that incorporating dynamicity can increase recall and precision. This improvement implies that PCD-GED has far better $f_1$-measure in comparison to PCD.
4. Conclusion

Regarding the role of protein complexes in many biological processes, detection of protein complexes from the experimental PPI networks is an essential problem in computational biology. Over the past decade, most research on biological networks has been focused on static topological properties, describing networks as collections of nodes and edges. But, in reality, cellular systems are highly dynamic and responsive to cues from environment (Jin et al., 2009).

Recently some research works (Tang et al., 2011; Li et al., 2012; Wang et al., 2013; Xiao et al., 2013) have concentrated on dynamics in computational analysis. A challenging task in post-genomic era is how to use the systematic and dynamic organization of PPI networks to detect biologically significant protein complexes. On the other hand, it is well-known that current PPI networks contain both halloffalse positive and false negative interactions. By using weighted networks in which each interaction is scored by a "reliability" value, one can reduce the effect of this noise. Also, according to the high degree of functional similarity between proteins in a protein complex core, it is expected that a complex core has high weighted density and not necessarily k-clique structure but this concept is not employed in the recent proposed methods.

Here, we presented a novel protein complex detection method called PCD-GED, which based on a simple active threshold, generates a series of subnetworks of active proteins according to each column of gene expression matrix to employ cell dynamicity in protein complex detection. Furthermore, it finds protein complexes based on core-attachments from a weighted PPI network. Considering the role of proteins in a complex core, PCD-GED finds the cores with high weighted density based on weighted clustering coefficient of each protein in PPI network. PCD-GED is also highly adaptable to maximize precision, recall or f1-measure by fine tuning of α, β parameters. Experimental results revealed that in PCD-GED no criterion is sacrificed for other such that precision, recall have same value and f1-measure has at least 29% and 47% improvements compared to other methods. As a future work, we propose extending PCD-GED to make use of more dynamic information. Using other data sources is another research candidate that can improve the accuracy results for protein complex detection method. Data from various sources enhances the accuracy of protein complex detection method.

Acknowledgments

The authors are grateful to the anonymous referees for their constructive and insightful comments.

Appendix 1. list of used symbols.

\[ G(V,E) \]
A PPI graph; V is the set of proteins, E is the set of interaction between proteins in V.

\[ p_i \]
Each protein in PPI network.

\[ c_i \]
A protein core.

\[ \alpha \]
Cut parameter, it is used for selecting the active proteins in each time-sequence subnetwork.

\[ \beta \]
Seed parameter, it is used in “SeedGeneration” function.

\[ \gamma \]
Set parameter, it is used in “CoreExpanding” function.

\[ m_i \]
The mean of expression values for each protein \( p_i \) in expression matrix.

\[ T_j \]
The \( j \)th column of gene expression matrix.

\[ TSN_j \]
The time-sequence Subnetwork consists of active proteins are selected based on \( j \)th column of gene expression matrix.

References

