Xu *et al. BMC Systems Biology* 2017, **11**(Suppl 7):135
DOI 10.1186/s12918-017-0504-3

BMC Systems Biology

RESEARCH Open Access

# CPredictor3.0: detecting protein complexes from PPI networks with expression data and functional annotations

Ying Xu[1], Jiaogen Zhou[3], Shuigeng Zhou[2,4] and Jihong Guan[1*]

*From* 16th International Conference on Bioinformatics (InCoB 2017)
Shenzhen, China. 20-22 September 2017

## Abstract

**Background:** Effectively predicting protein complexes not only helps to understand the structures and functions of proteins and their complexes, but also is useful for diagnosing disease and developing new drugs. Up to now, many methods have been developed to detect complexes by mining dense subgraphs from static protein-protein interaction (PPI) networks, while ignoring the value of other biological information and the dynamic properties of cellular systems.

**Results:** In this paper, based on our previous works CPredictor and CPredictor2.0, we present a new method for predicting complexes from PPI networks with both gene expression data and protein functional annotations, which is called CPredictor3.0. This new method follows the viewpoint that proteins in the same complex should roughly have similar functions and are active at the same time and place in cellular systems. We first detect active proteins by using gene express data of different time points and cluster proteins by using gene ontology (GO) functional annotations, respectively. Then, for each time point, we do set intersections with one set corresponding to active proteins generated from expression data and the other set corresponding to a protein cluster generated from functional annotations. Each resulting unique set indicates a cluster of proteins that have similar function(s) and are active at that time point. Following that, we map each cluster of active proteins of similar function onto a static PPI network, and get a series of induced connected subgraphs. We treat these subgraphs as candidate complexes. Finally, by expanding and merging these candidate complexes, the predicted complexes are obtained.
We evaluate CPredictor3.0 and compare it with a number of existing methods on several PPI networks and benchmarking complex datasets. The experimental results show that CPredictor3.0 achieves the highest F1-measure, which indicates that CPredictor3.0 outperforms these existing method in overall.

**Conclusion:** CPredictor3.0 can serve as a promising tool of protein complex prediction.

**Keywords:** PPI network, Protein complex, Gene expression, GO annotation

*Correspondence: jhguan@tongji.edu.cn
[1]Department of Computer Science and Technology, Tongji University, Shanghai 201804, China
Full list of author information is available at the end of the article

## Background

Proteins, as the material basis of life, are the ultimate controller and direct performer of life activities, participate almost all of biological functions. Most proteins do not perform biological functions alone, but form protein complexes with others [1]. So to have a more comprehensive and deep understanding of cell compositions and life processes, the identification of protein complexes is very important.

Although biological techniques such as Tandem Affinity Purification with Mass Spectrometry (TAP-MS) [2] can detect protein complex directly, the accuracy is not high. In addition, biological techniques are usually time-consuming and costly. These make biological techniques cannot meet the requirement of post-genome research for handling big biological data.

With the development of high-throughput experimental technologies, PPI data rapidly increase, which provides chance for using computational methods to detect protein complexes. Moreover, computational methods can overcome drawbacks of experimental technologies. PPI networks can be constructed by using PPI data, where nodes and edges represent proteins and interactions between proteins respectively. Empirical studies on PPI networks indicate that there are modular components in these networks [3]. From the view of network topography, these modules are made up of closely related proteins; From the view of biology, these modules aggregate proteins that perform functions together. Thus, protein complexes can be detected by mining the modular structures (i.e., dense subgraphs or subnetworks) from PPI networks.

So far, there have been many researches that put forward different graph clustering methods to detect local dense subgraphs to detect protein complexes from PPI networks [4–9]. These methods are intuitive and straightforward. To overcome the high false-positive and false-negative problems in PPI networks, many studies have attempted to improve the reliability of PPI data by exploiting gene expression data [10, 11] and protein functional annotations [12, 13] to improve the accuracy of protein complex prediction. In addition to dense subgraph mining based approaches, in the past decade some other method have also developed, including the core-attachment structure based methods [14, 15], methods for non-dense junction complexes and small complexes [16, 17], and methods using dynamic PPI networks [18]. In next section, we will present a relatively comprehensive survey on complex prediction.

In this paper, based on our previous works CPredictor [19] and CPredictor2.0 [16, 17], we propose a new method called CPredictor3.0, which considers both dynamic PPI and functional information. First, we use expression data of different time points to detect active proteins at the same time point, meanwhile we cluster proteins by functional annotations such that each cluster contains proteins of similar function(s). Then, we compute protein clusters of similar function(s) and being active at the same time point by set intersection operation with one set corresponding to an active protein set generated by expression data and the other set corresponding to a protein cluster generated from functional annotations. Following that, we map the resulting clusters onto a static PPI network and obtain a series of induced connected subgraphs, which are treated as candidate complexes. Finally, we identify protein complexes by expanding and merging the candidate complexes. Our experimental results validate the effectiveness of CPredictor3.0, which outperforms the existing methods in overall.

## Related work

So far, a variety of computational methods for complex prediction have been proposed. Here, we present a brief survey on the related works by roughly classifying the existing methods into the following types: methods based on local dense subgraphs, methods based on the Core-Attachment Model, methods based on dynamic PPI networks, methods based on supervised learning. Among them, methods based on local dense subgraphs constitute the most part of existing works. Note that this method hierarchy only reflects our view of existing works. There may be other hierarchies of existing works in the literature. And a brief survey, we cannot cover all existing works, but we try our best to present the major existing works.

### Methods based on local dense subgraphs

As one of earliest computational methods of complex prediction, MCODE [4] first weights each protein based on its core-clustering density in the PPI network, then the protein (say $p$) with the largest weight is selected to be a seed node of a primary complex, which is expanded by including other proteins whose weights exceed a pre-set threshold recursively, till there are no more nodes to be added. If there are unprocessed nodes, new complexes will be generated in the way above. Finally, the neighbors of each complexes generated above are included into the complexes if their weights is higher than a pre-set "fluff" parameter. MCL [5] predicts complexes based on random walk in a PPI network, it is a fast and highly scalable clustering method. To simulate random walk, two operators, expansion and inflation are used to manipulate the adjacency matrix iteratively. The aim of those two operators is to separate dense subgraphs out from the network. Protein complexes predicted by this method are non-overlapped. ClusterONE [6] uses a new measure to compute the cohesiveness of one subgraph, and works by seeding and expanding with neighboring nodes. This method performs better than the other methods when it was developed.

As there exists high false-positives and false-negatives in PPI networks. Some methods weight edges of PPI network using PPI network topology, gene expression data and protein function to improve the reliability of PPI networks. DPClus [20] weights each edge according to the number of shared neighbors of the node pair, then the weight of each node is computed by summing the weights of all its edges. Cao et al. [21] treated complex prediction as an optimization problem and built the objective function by considering a variety of topology characteristics, then the genetic algorithm was employed to detect complexes from PPI networks.

In general, proteins that form functional groups have similar gene expression, so some methods weight edges of PPI networks using expression data. MATISEE [10] measures the intensity of interaction of a pair of proteins using the correlation of expression data. Ou-Yang et al. [11] detected protein complexes from signed PPI networks, the sign of each edge is computed using the pearson correlation coefficient of gene expression of the two proteins.

Except for expression data, protein function provides important clue for protein complex detection. SWEMODE [12] proposed by Lubovac weights each edge based on the semantic similarity of the function(s) of two proteins, the weight of each node is given by the weighted clustering coefficients of the nearest neighbors. Cho et al. [13] weighted each edge according to the functional similarity of two nodes, and the weight of each node is the sum of weights of its edges. The flow simulation algorithm is then used to split the flow from the nodes with larger weights. As each flow goes along edges and its influence decay according to the similarity of each node pair it passes, it stops when its influence is less than a certain threshold. Thus, the PPI network is divided into a plurality of subgraphs consisting of proteins connected by flow from the same source protein.

In our previous works CPredictor [19] and CPredictor2.0 [16, 17], we also used protein functional information. But different from the existing methods, we first used protein functional information to cluster proteins, then mapped the clusters onto PPI networks. The difference between CPredictor and CPredictor2.0 lies in the usage of functional information. In this paper, we follow the same idea of CPredictor and CPredictor2.0, but we also use expression data. That is, we consider the dynamic property of PPI networks.

## Methods based on Core-Attachment model

Gavin et al. [1] studied the structures of yeast protein complexes and found that each protein complex consists of two parts: the core is made of proteins connected tightly, and attachments that have relatively sparse interactions with the core .

Following the core-attachment structure, two methods CORE [14] and COACH [15] were proposed. CORE assesses the probability that two proteins belong to the same core using their common neighbors. Then, larger cores are produced by merging cores of sizes two, three and so on repeatedly. Finally, a protein can be added into one core as attachment if it has interactions with more than half proteins in the core. COACH first identifies small dense subgraphs around proteins of high weight, and then generates cores by merging those dense subgraphs. It uses the same way to add attachments as CORE does. Later, Peng et al. [22] porposed the WPNCA method, which divides a weighted PPI network into multiple closely connected subgraphs by using the PageRank-Nibble algorithm, and then, protein complexes are generated in each subgraph based on the Core-Attachment structure.

## Methods based on dynamic PPI networks

Earlier methods detect complexes from static PPI networks. Actually, the interactions among proteins are dynamic and change over time at different biological stages. In recent year, there are some works on detecting protein complexes from dynamic PPI networks. Tang et al. [18] applied a fixed threshold to cluster proteins using expression data such that each cluster consists of proteins active at the same time point. Since the expression levels of different proteins are quite different, it is unreasonable to use a fixed threshold for all proteins. Later, Wang et al. [23] proposed the three-sigma model to calculate active threshold for proteins, and achieved better performance of complex prediction. Zhang et al. [24] first identified transient and stable protein interactions to construct dynamic PPI networks based on the three-sigma model, then predicted protein complexes from the dynamic PPI networks. Lei et al. [25] constructed dynamic PPI networks using the same method as in [24], and then optimized the parameters of Markov clustering by the firefly algorithm to detect protein complexes.

## Methods based on supervised learning

Some works use supervised learning to detect protein complexes. Qi et al. [26] classified the topological properties of protein complexes into four categories, and used these properties as features to train probability bayesian network, which was used to predict complexes from the subgraphs generated from PPI networks randomly. Yong et al. [27] used true complexes as training data, and a variety of information such as interactions, functions, text and topology as features, to train Bayesian model to predict the probabilities of protein interactions included in small complexes, large complexes and non-complexes, and then small complexes of size 2,3 were extracted.

## Methods

Here, we first give an overview of CPredictor3.0, then describe the major components of CPredictor3.0 in detail.

### Overview

Figure 1 shows the flowchart of CPredictor3.0, which consists of six major steps: 1) Detecting active proteins; 2) Clustering proteins by function; 3) Computing active proteins of similar function; 4) Extracting candidate complexes from PPI networks; 5) Expanding candidate complexes; 6) Merging candidate complexes.

The rationale behind our method is that proteins of a complex performs some function(s) by interacting with each other at the same time and the same place in cellular systems [28]. CPredictor3.0 works like this: First, it detect active proteins from gene expression data for different time points, then it cluster proteins according to functions by using functional annotations. With the results of the above two steps, it computes active protein clusters of similar function. Following that, these clusters are mapped onto a PPI network to extract induced connected subgraphs, which are taken as candidate complexes. Finally, we expand the resulting candidate complexes and merge overlapping ones to get the final predicted complexes. In what follows, we describes these steps in detail.

### Detecting active proteins

Gene expression data reveal the dynamic properties of proteins in their lifetime. As a protein is not always active, its expression level changes with its activity degree. Concretely, higher gene expression level means higher activity. To get the active time points of each protein, Tang et al. [18] set a global fixed threshold for all proteins. There are two drawbacks with a global threshold. On the one hand, there is noise in biological data. On the other hand, the gene expression curve for each protein is different. To solve these problems, Wang et al. [23] proposed the three-sigma model to compute active threshold for each protein. In this paper, we use the three-sigma model to calculate the active threshold for each protein.

Suppose the expression data are measured at $n$ time points. For a protein $p$, $V_k(p)$ represents protein $p$'s expression value at time point $k$, $\mu(p)$ and $\sigma(p)$ are the mean and the standard deviation of expression values over the period from 1 to $n$. The active threshold of protein $p$ is evaluated as follows:

$$Active(p) = \mu(p) + \beta * \sigma(p) * \left(1 - \frac{1}{1 + \sigma(p)^2}\right), \quad (1)$$

Above, $\beta$ is an adjustable parameter that helps us to get the most optimal threshold. Usually, we set $\beta$ =0, 1, 2, 3.

After obtaining the active thresholds for all proteins, we can collect all active proteins at each time point. That is,

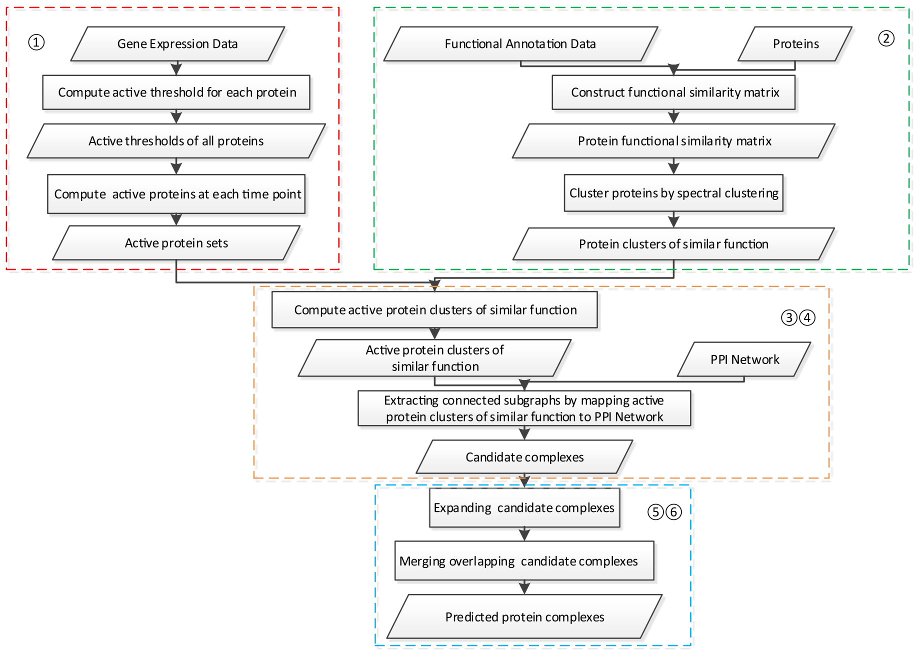
①
Gene Expression Data
Compute active threshold for each protein
Active thresholds of all proteins
Compute active proteins at each time point
Active protein sets
Functional Annotation Data
Proteins
②
Construct functional similarity matrix
Protein functional similarity matrix
Cluster proteins by spectral clustering
Protein clusters of similar function
Compute active protein clusters of similar function
③④
Active protein clusters of similar function
PPI Network
Extracting connected subgraphs by mapping active protein clusters of similar function to PPI Network
Candidate complexes
Expanding candidate complexes
⑤⑥
Merging overlapping candidate complexes
Predicted protein complexes

**Fig. 1** The flowchart of CPredictor3.0. 1) Detecting active proteins; 2) Clustering proteins by function; 3) Computing active proteins of similar function; 4) Extracting candidate complexes from PPI networks; 5) Expanding candidate complexes; 6) Merging candidate complexes

for each protein $p$ at time point $i$ ($i = 1, \ldots, n$), if its expression value is no less than $Active(p)$, then it is an active protein at time point $i$. In such a way, we can get the set of active proteins $AP_i$ for each time point $i$. Thus, we have a series of active protein sets $\{AP_i\ (i=1, \ldots, n)\}$.

### Clustering proteins by function

Here, we cluster proteins by functional annotations. First, we compute the functional similarity of any two proteins using the method proposed by Wang et al. [29], then we employ the spectral clustering algorithm to cluster the proteins with the computed similarity matrix.

The similarity between any two proteins is computed by the GO terms annotated on the two proteins. GO includes a series of biological terms to describe gene and gene products such as protein, and covers three aspects: biological process (BP), cellular component (CC), and molecular function (MF). Here, we use only BP data. GO can be represented as a directed acyclic graph (DAG), in which nodes and edges represent terms and their relationships (e.g. 'is-a' and 'part-of') between two terms. A GO term $A$ can be described as $DAG_A$=($A$, $T_A$, $E_A$) where $T_A$ consists of term $A$ and all its ancestors in DAG, $E_A$ is composed of all edges (relationships) connecting $A$ to all terms in $T_A$. As defined in Wang's method, the semantic content of a term is the sum of semantic contributions of all its ancestors in $DAG_A$ to $A$. The semantic contribution of term $t$ to $A$ is as follows:

$$S_A(t) = \begin{cases} 1 & t = A \\ max\{w_e * S_A(t') | t' \in childrenof(t)\} & t \neq A \end{cases} \tag{2}$$

where function $childrenof(t)$ returns the children of $t$ in $DAG_A$, and $w_e$ as the weight on the edge between $t$ and $t'$, which depends on the relationship type between the two terms. For example, the weight is 0.8 for 'is-a' and 0.6 for 'part-of'.

So the semantic value $SV(A)$ of term $A$ is evaluated as follows:

$$SV(A) = \sum_{t \in T_A} S_A(t). \tag{3}$$

The semantic similarity $S_{GO}(A, B)$ between term $A$ and $B$ is evaluated as follows:

$$S_{GO}(A, B) = \frac{\sum_{t \in T_A \cap T_B} SV(t)}{SV(A) + SV(B)}. \tag{4}$$

Generally, one protein may participate one or more biological functions, so one protein may be annotated by multiple terms. For two proteins $P1$ and $P2$, which are annotated by $\{go_{11},\ go_{12},\ \cdots,\ go_{1m}\}$ and $\{go_{21},\ go_{22},\ \cdots,\ go_{2n}\}$ respectively, their similarity can be evaluated as follows:

$$Sim(P1, P2) = \frac{\sum_{i=1}^{m} Sim\left(go_{1i}, P_2\right) + \sum_{j=1}^{n} Sim\left(go_{2j}, P_1\right)}{m + n} \tag{5}$$

$$Sim(go, P) = max_{1 \le i \le k}\left(S_{GO}\left(go, go_i\right)\right) \tag{6}$$

After getting the similarity matrix for all proteins, where each element represents the semantic similarity of two proteins. Then, we apply the spectral clustering algorithm [30] to the matrix to cluster all proteins into $K$ disjointed clusters $PC$=$\{PC_1, PC_2, \cdots, PC_K\}$ where $K$ is an adjustable parameter to control the number of protein clusters.

### Complex generation

#### *Computing active protein clusters of similar function*

With the sets of active proteins $\{AP_i | i = 1, \ldots, n\}$ and the set of protein clusters of similar function $\{PC_j | j = 1, \ldots, K\}$, here we go to compute the active protein clusters of similar function. For time point $i$, the set of active protein clusters of similar function is $APC_i = AP_i \cap \{PC_j | j = 1, \ldots, K\} = \{AP_i \cap PC_j | j = 1, \ldots, K\}$. Thus, we can get all active protein clusters of similar function as follows:

$$\begin{aligned} APC &= \{APC_i | i = 1, \cdots, n\} \\ &= \{AP_i \cap PC_j | i = 1, \cdots, n; j = 1, \ldots, K\} \\ &= \{APC_{ij} | i = 1, \cdots, n; j = 1, \ldots, K\}. \end{aligned} \tag{7}$$

#### *Computing candidate complexes*

We have already gotten the set of active protein clusters of similar function, considering that complexes consist of interacting proteins, we map all active protein clusters of similar function onto a PPI network $G = (V, E)$ where $V$ and $E$ represent proteins and interactions respectively, to get connected subgraphs induced by each cluster on $G$. Concretely, given the active protein cluster of similar function $APC_{ij}$, we map $APC_{ij}$ onto $G$ and get the induced graph $G_{ij} = (V_{ij}, E_{ij})$ by $APC_{ij}$. That is, $V_{ij} = APC_{ij}$ and $E_{ij}$ are the set of edges in $G$ that connect proteins in $APC_{ij}$. $G_{ij}$ may be not a connected graph, i.e., it may consist of several connected subgraphs. We treat each resulting subgraph of size $> 1$ as a candidate complex. Thus, from $G_{ij}$ we get a set of candidate complexes $CC_{ij}$. Similarly, by mapping other active protein clusters of similar function onto $G$, we obtain other candidate complexes. We denote the set of all candidate complexes as $CC$= $\{CC_{ij} | i = 1, \cdots, n; j = 1, \ldots, K\}$.

#### *Candidate complex expanding*

Here, we try to expand each candidate complex on $G$. Consider a candidate complex $c \in CC$, its corresponding graph is $G_c = (V_c, E_c)$. First, we search the set of neighbor nodes of candidate complex $c$ in $G$, which is denoted as $N_G(c)$. We have

$$N_G(c) = \{p|E(p, G_c) \in E \quad and \quad p \in (V - V_c)\} \qquad (8)$$

where $p$ is any protein not in $V_c$ and $E(p, G_c)$ is the set of interactions between protein $p$ and any protein in $G_c$. For any protein $p \in N_G(c)$, if the following condition holds, we add $p$ and its interactions with proteins in $c$ to $c$:

$$|E(p, G_c)| \geq \alpha * |V_c|. \qquad (9)$$

Above, $\alpha$ is a pre-specified threshold. In this paper, it is set to 0.6 by experience. $\alpha$ is in the range 0 to 1. When $\alpha$ is set to 0, all neighbors of $V_c$ will be added into $V_c$. Otherwise, if $\alpha$ is set to 1, no nodes will be added into $V_c$. Usually, if one node interacts with more than half nodes of $V_c$, the node will be added [15, 19]. Our experimental results validate the rationality of the value setting. The expansion process continues till no any more neighbor can be added to $c$. We do expansion to all candidate complexes in $CC$, and denote the set of candidate complexes after expansion as $CC_{exp}$.

#### *Candidate complexes merging*

There may be overlapping between candidate complexes in $CC_{exp}$. For two overlapping candidate complexes, if their overlapping score is larger than a predefined threshold, we merge them to one complex. Concretely, given two candidate complexes $c_A$ and $c_B$, their overlapping score is evaluate as follows:

$$OS(c_A, c_B) = \frac{|V_{c_A} \cap V_{c_B}|}{|V_{c_A} \cup V_{c_B}|}. \qquad (10)$$

If $OS(c_A, c_B) \geq \gamma$, we merge $c_A$ and $c_B$. Here, $\gamma$ is a pre-specified parameter. By experiments, we set $\gamma = 0.8$. When there are no more candidate complexes that can be merged, the resulting and remaining candidate complexes constitute the final set of predicted complexes.

### The Algorithm

The algorithm of CPredictor3.0 is presented in Algorithm 1. Here, Lines 5-10 are for computing active protein clusters of similar function, Lines 11-24 are for candidate complexes extraction, Lines 25-35 are for candidate complexes expansion, and Lines 36-40 are for candidate complexes merging.

## Results and discussion

### Data sources and Metrics

We downloaded gene expression data GSE3413 [31] from Gene Expression Omnibus (GEO) to compute active proteins. As gene products can cover more than 96% proteins in PPI networks, it is reasonable to detect active proteins from expression data for different time points. GSE3413

**Algorithm 1** The algorithm of complex generation

```
Input: The active protein sets AP; The protein clusters of similar function PC; PPI network G=(V, E);
Output: Predicted protein complex set PPC;
1: APC ={} ; /* the set of clusters with active proteins of similar function */
2: CC = {}; /* the set of candidate complexes by mapping APC to PPIN */
3: $CC_{exp}$ = {}; /* the set of candidate complexes by expanding CC */
4: PPC ={} ; /* the set of predicted protein complexes by merging $CC_{exp}$ */
5: for i=1 to $|AP|$ do
6:     for j=1 to $|PC|$ do
7:         $APC_{ij} = AP_i \cap PC_j$;
8:         $APC = APC \cup APC_{ij}$;
9:     end for
10: end for
11: for i=1 to $|AP|$ do
12:     for j=1 to $|PC|$ do
13:         /* $G_{ij}$ is the subgragh of G, $V_{ij} = APC_{ij}$ , and $E_{ij}$ is the set of edges connecting proteins in $V_{ij}$. */
14:         $G_{ij} = (V_{ij}, E_{ij})$;
15:         for q =1 to the number of connected subgraphs in $G_{ij}$ do
16:             /* $G_{ij}^q$ is one of connected subgraph of $G_{ij}$, $V_{ij}^q$ is the set of proteins in $G_{ij}^q$, and $E_{ij}^q$ is the set of edges connecting proteins in $V_{ij}^q$. */;
17:             $G_{ij}^q = (V_{ij}^q, E_{ij}^q)$;
18:             if $|V_{ij}^q| \geq 2$ then
19:                 $CC_{ij} = CC_{ij} \cup V_{ij}^q$;
20:             end if
21:         end for
22:         $CC = CC \cup CC_{ij}$ ;
23:     end for
24: end for
25: for each candidate complex $c$ in $CC$ do
26:     $G_c = (V_c, E_c)$ /* corresponding graph of c on G */
27:     $N_G(c)$ = the neighbors of candidate complex $c$ in G;
28:     for each protein $p$ in $N_G(c)$ do
29:         $|E(p, G_c)|$ = the number of interactions between p and $G_c$;
30:         if $|E(p, G_c)| > 0.6 * |V_c|$ then
31:             $c = c \cup p$;
32:         end if
33:     end for
34:     $CC_{exp} = CC_{exp} \cup c$;
35: end for
36: for each two candidate complexes $c_A$ and $c_B$ in $CC_{exp}$ do
37:     if $OS(c_A, c_B) > 0.8$ then
38:         $PPC = PPC \cup (c_A$ merge $c_B)$;
39:     end if
40: end for
```

is the expression profiling of yeast in the form of matrix, which contains three successive metabolic cycles, and each cycle has 12 time intervals. So each protein has 12 expression values in every cycle. To reduce the impaction of noise, we took the averaged expression values of 12 time points over three cycles, and used these averaged values in our experiments.

In addition to gene expression data, we used three PPI datasets of yeast, which are referred to as Krogan [32], Collins [33] and WI-PHI [34]. The numbers of proteins and interactions in these three datasets are presented in the 2nd and 3rd columns of Table 1. MIPS [35] and CYC2008 [36] were used as reference complex sets, the numbers of complexes and proteins contained in these two sets are presented in the 2nd and 3rd columns of Table 2. In this paper, the GOSemsin package [37] was employed to compute the protein functional similarity matrix.

To measure the quality of predicted protein complexes, predicted complexes are checked against with reference complexes. Let $P=(V_p, E_p)$ and $R=(V_r, E_r)$ are a predicted complex and a known complex, respectively. The affinity score (AS) of the two complexes is defined as follows:

$$AS(P,R) = \frac{|V_p \cap V_r|^2}{|V_p| * |V_r|}. \tag{11}$$

Usually, $P$ and $R$ are considered matched when $AS(P,R) \geq 0.2$. This criterion was widely used in the literature [19, 21, 38–43]. However, as stated in PPSampler2 [43], for complexes of size 2, that is, the size of $V_p$ and $V_r$ is 2, then we have $\frac{1}{2*2} = 0.25 > 0.2$. This means that size-2 candidates can be easily considered as real complexes, which may bring randomness to the final result and affect the correctness of performance evaluation. Actually, most existing methods cannot effectively detect size-2 complexes, because they treat complexes as dense subgraphs while size-2 complexes are just single edges. So a common strategy is simply neglecting the size-2 complexes. In our method, we follow this strategy to discard those predicted complexes with only two proteins.

In our method, *recall*, *precision* and *F1-measure* are used to measure the prediction performance. Let $PS = \{ps_1, \cdots, ps_m\}$ and $RS=\{rs_1, \cdots, rs_n\}$ are the predicted complex set and the benchmark complex set respectively, the three performance metrics are evaluated as follows:

$$recall = \frac{N_r}{|RS|}, \tag{12}$$

$$precision = \frac{N_p}{|PS|}, \tag{13}$$

$$F1 - measure = \frac{2 * recall * precision}{recall + precision}. \tag{14}$$

Above, $N_r$ is the number of reference complexes that match at least one predicted complex, $N_p$ is the number of predicted complexes that match at least one reference complex. $|RS|$ and $|PS|$ are the size of benchmark complex set and the size predicted complex set respectively.

**Table 1** The statistics of PPI datasets

| PPI network | # proteins | # interactions |
|---|---|---|
| Krogan | 2674 | 7075 |
| Collins | 1622 | 9074 |
| WI-PHI | 6400 | 50000 |

**Table 2** The statistics of benchmark datasets

| benchmark database | # complexes | # proteins |
|---|---|---|
| MIPS | 313 | 1237 |
| CYC2008 | 349 | 1627 |

## Experimental results

We present the experimental results from three aspects. Firstly, we count the size distribution of predicted protein complexes of different algorithms. Secondly, we check the impact of two parameters $K$ and $\beta$ on prediction performance of our method. Finally, we compare our method with major existing methods in terms of recall, precision and F1-measure.

### *The size distribution of predicted protein complexes*

As our method employs function and expression constraints to filter complexes, which may tend to produce small complex candidates. However, our method also use cluster expansion and merging strategies to generate the final predictions. To check the effectiveness of the expansion and merging strategies, here we present the size distribution of predicted protein complexes for different methods on different PPI datasets against different complex benchmark sets in Fig. 2. It is clear that complexes with 5 or more proteins count the largest part of our method's prediction results. This means that the expansion and merging strategies employed in our method are effective.

### *The effect of parameters on the performance of CPredictor3.0*

In our method, there are two adjustable parameters $K$ and $\beta$ which can impact prediction performance. Here, we present the results of how F1-measure changes with the values of the two parameters, which are shown in Fig. 3.

By checking the complexes in the reference sets, we can see that the size of most protein complexes is less than 30. In experiments, we set the value of $K$ to from 1 to 100.

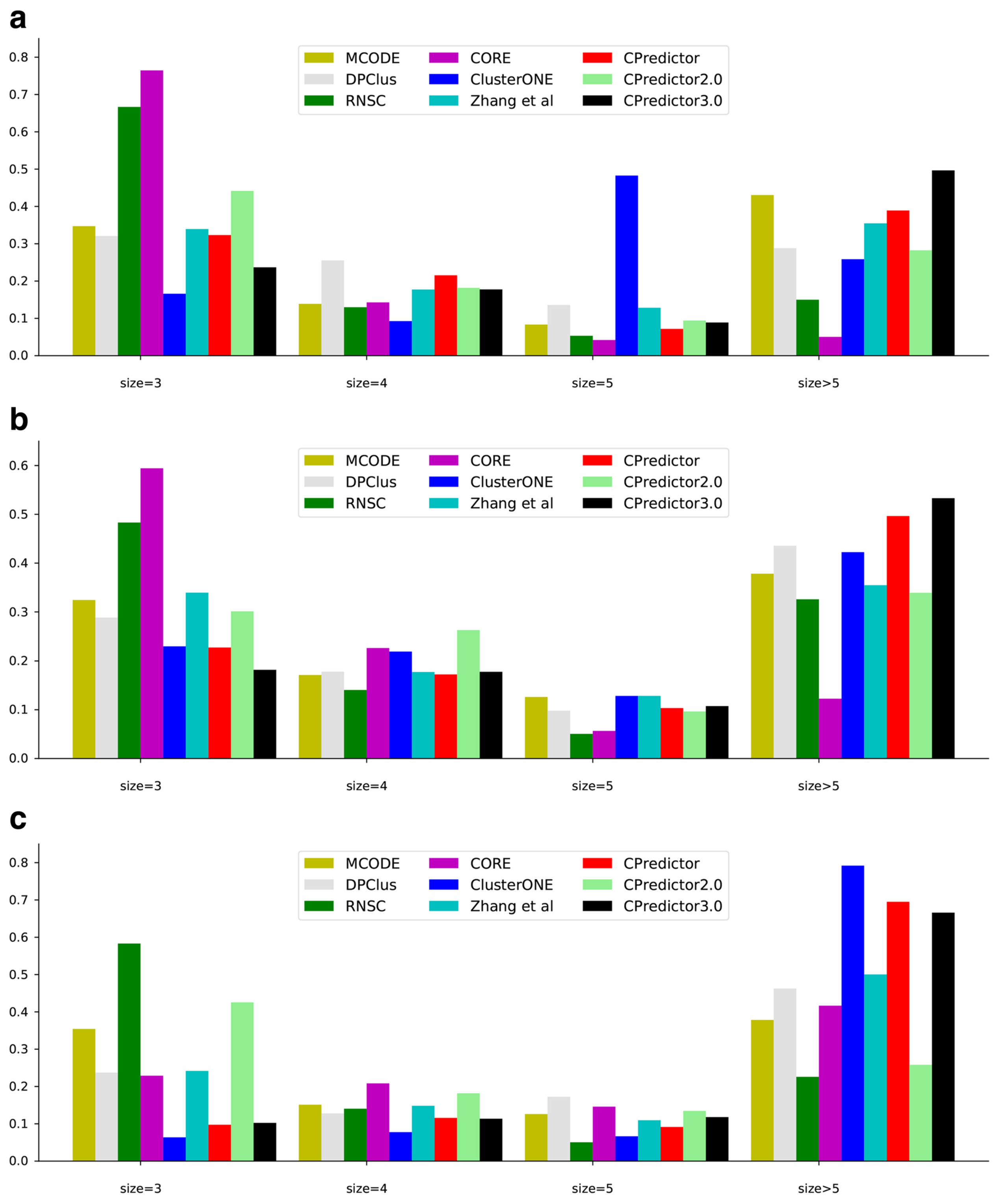

**Fig. 2** The distribution of protein complex size. **a** Krogan PPI data set. **b** Collins PPI data set. **c** WI-PHI PPI data set

The parameter $\beta$ is used to set the threshold for filtering active proteins. According to three sigma(SD) model, we set the largest value of $\beta$ to 3, and change it from 0 to 3. As shown in Fig. 3, the performance tends to be stable when $K$ is greater than 20. For different $K$ values, the best F1-measure is achieved when $\beta$ is set to 0. So in the

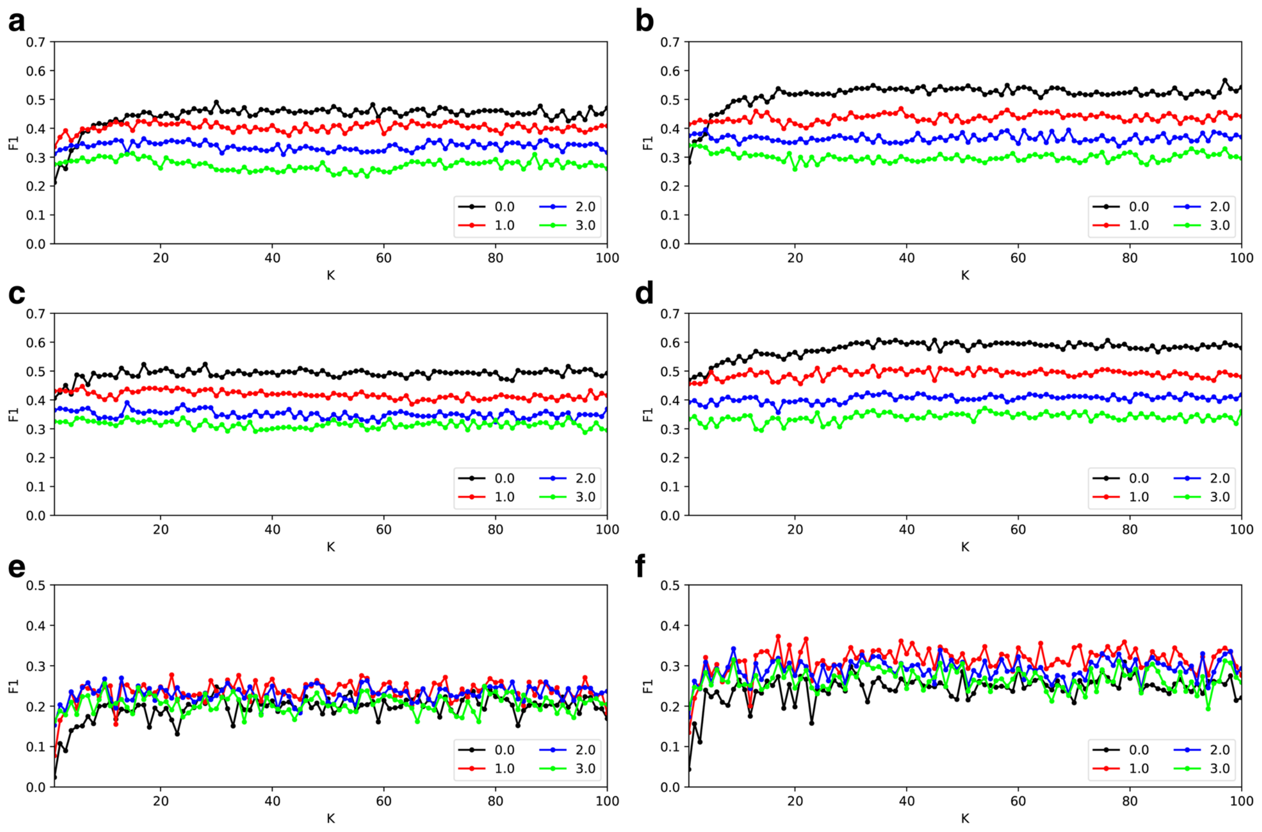

**Fig. 3** The effect of $K$ and $\beta$ on prediction performance. **a** Krogan PPI data and MIPS reference complexes set. **b** Krogan PPI data and CYC2008 reference complexes set. **c** Collins PPI data and MIPS reference complexes set. **d** Collins PPI data and CYC2008 reference complexes set. **e** WI-PHI PPI data and MIPS reference complexes set. **f** WI-PHI PPI data and CYC2008 reference complexes set

comparison experiments, we set $K$=30, $\beta$=0 for Collins PPI data, $K$=35, $\beta$=0 for Krogan PPI data, and $K$=17, $\beta$=1 for WI-PHI PPI data.

By checking the predicted complexes further, we can see that there are some large complexes of size > 100 when $K$ is set small. This is reasonable. As parameter $K$ indicates the number of clusters that the proteins are to be divided. So, small $K$ will lead to large clusters, i,e, large complexes, and vice versa.

As for parameter $\beta$, which is the threshold for filtering active proteins from gene expression data. A larger $\beta$ will results in more proteins being filtered as inactive proteins. In Fig. 3, we can see that when $\beta$ is set to 0, i.e., we set $\beta$ to the mean of expression values over all time points, we get the best performance on Collins and Krogan PPI networks, while the best performance is achieved on WI-PHI network with $\beta = 1.0$.

### *Comparison with major existing methods*

Here, we compare our method CPreditor3.0 with eight existing protein complex prediction methods, including MCODE [4], DPClus [20], RNSC [44], CORE [14], ClusterONE [6], Zhang et al. [24], CPredictor [19], and CPredictor2.0 [16, 17]. Some of them are the state of the art techniques, such as ClusterONE [6] and CPredictor2.0 [16, 17]. All parameters in these compared methods were set as suggested by their authors.

The experimental results are shown in Fig. 4. We can see in five of the six experimental settings, CPredictor3.0 achieves the highest F1-measure. And in the remaining setting, CPredictor3.0 still has comparable F1-measure to the best one. In three of the six settings, CPredictor3.0 has the highest precision, and has the 2nd highest precision in the other three settings. As for recall, CPredictor3.0 stays at the second or third position in five settings and at the fifth position in one setting. Thus, in overall our method performs best among the nine methods.

From Fig. 4, we can see that all methods have different performance on different PPI datasets and complexes reference sets. To give a detailed picture, we compute the average F1 values of all compared methods in the six settings. The results are presented in Table 3. Checking these results, we can see that: on the one hand, giving the PPI dataset (Krogan, Collins or WI-PHI), the performance with CYC2008 as reference set is better than that with MIPS as reference set. On the other hand, giving

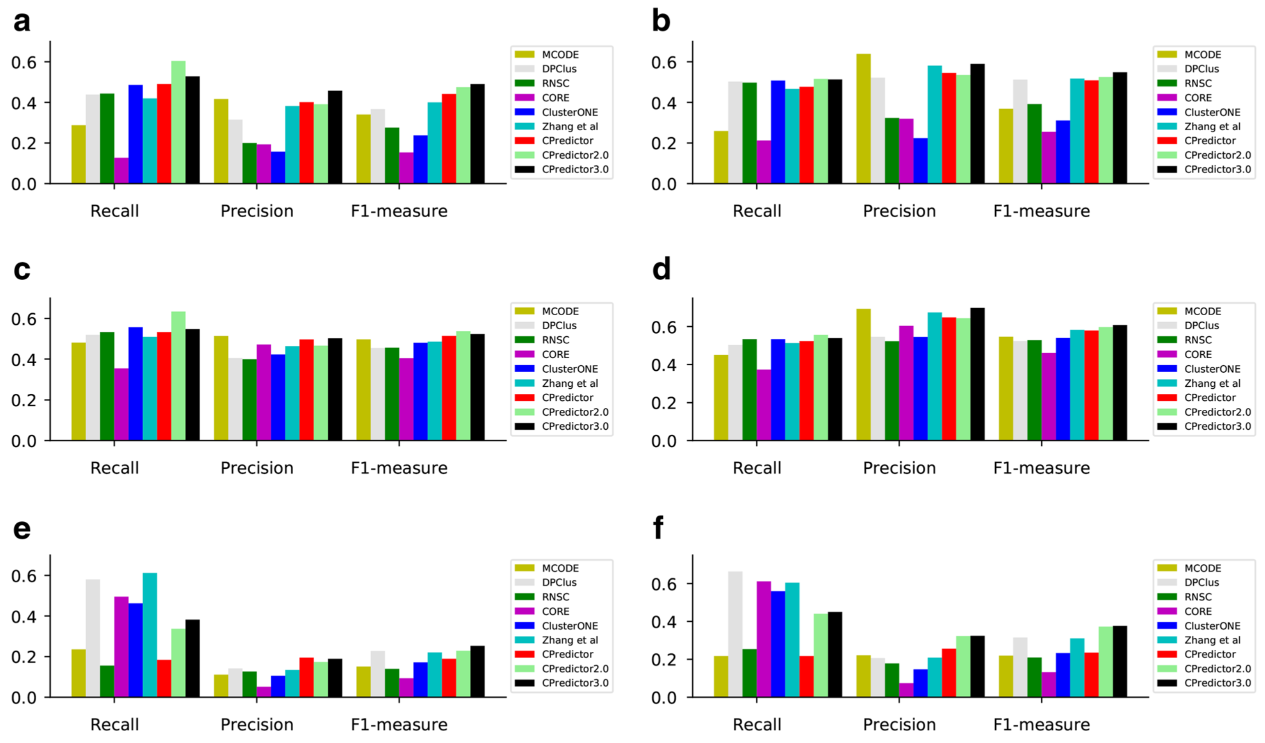

**Fig. 4** Performance comparison with eight existing protein complex prediction algorithms in terms of recall, precision, and F1-measure. Our method CPreditor3.0 achieves the highest F1-measure in five of the six experimental settings. (**a**) Results with Krogan as PPI dataset and MIPS as complex reference set, (**b**) Results with Krogan as PPI dataset and CYC2008 as complex reference set, (**c**) Results with Collins as PPI dataset and MIPS as complex reference set, (**d**) Results with Collins as PPI dataset and CYC2008 as complex reference set, (**e**) Results with WI-PHI as PPI dataset and MIPS as complex reference set, (**f**) Results with WI-PHI as PPI dataset and CYC2008 as complex reference set

the complex reference set (MIPS or CYC2008), using the Collins PPI dataset gets the best performance and using the WI-PHI PPI dataset has the worst performance. This observation can be explained by the number of overlapping proteins between the PPI dataset and the reference set used in the prediction. Comparing with Krogan and Collins, WI-PHI has the largest number of proteins. Most predicted complexes from WI-PHI cannot find matching complexes in the two reference sets, which results in low performance.

To give a detailed explanation, we compute the ratio of the number of overlapping proteins between each PPI dataset and each complexs reference set over the number of proteins contained in the PPI dataset. We call it “overlapping ratio” in short. The results are presented in Table 4. From this table, we can see that 30.6% and 49.2% proteins in the Krogan PPI dataset and the Collins PPI dataset are overlapping with that of the MIPS complex set, while there are only 19.1% proteins in the WI-PHI PPI dataset are overlapping with that of the MIPS complex set. The overlapping ratio of Krogan, Collins and WI-PHI with CYC2008 are 43.1, 68.8 and 25.3% respectively. In summary, for any PPI dataset, the overlapping ratio with CYC2008 is higher than that with MIPS; For any reference set, the highest overlapping ratio is with Collins, then with Krogan, and the lowest overlapping ratio is with WI-PHI. This trend is completely consistent with the results in Table 3. This explains the performance difference of the six settings.

## Conclusions

This paper introduced a new method CPredictor3.0 to boost complex prediction performance from PPI networks by using both expression data and functional

**Table 3** The average F1-measure values of the nine algorithms on various PPI datasets and complexes reference sets

| | Collins | | Krogan | | WI-PHI | |
|---|---|---|---|---|---|---|
| | CYC2008 | MIPS | CYC2008 | MIPS | CYC2008 | MIPS |
| F1 | 0.5518 | 0.4837 | 0.4376 | 0.3534 | 0.2672 | 0.1861 |

**Table 4** Overlapping protein ratios of between PPI datasets and complexes reference sets

| Benchmark database | Krogan | Collins | WI-PHI |
|---|---|---|---|
| MIPS | 30.6% | 49.2% | 19.1% |
| CYC2008 | 43.1% | 68.8% | 25.3% |

annotations. Experiments on three commonly used PPI datasets and two benchmark complexes sets show that CPreditor3.0 performs best in overall. It is well recognized that complexes consist of proteins that have similar function and are active at the same time and place in cellular systems. Our method considers all these aspects, including function and dynamic interaction by using PPI data, functional annotations and expression data. This may explain the best performance of our method.

As for future work, on the one hand, we are considering more advanced models to extract complexes from PPI networks, such as graph sparsity models [45] and temporal graph mining models [46]. On the other hand, small complex detection is a more challenging task [17], which is another focus of our future study. Thirdly, for better complex prediction performance, we will also consider building reliable and robust PPI networks by fusing multiple networks [47].

**Abbreviations**
GO: Gene ontology; DAG: Directed acyclic graph; PPI: Protein-protein interaction

**Acknowledgements**
Not applicable.

**Funding**
National Natural Science Foundation of China (NSFC) (grant No. 61772367) for manuscript writing and publication costs of this article; The National Key Research and Development Program of China (grant No. 2016YFC0901704) for data collection and analysis; the Program of Shanghai Subject Chief Scientist (15XD1503600) for data collection and manuscript writing. No funding body played any role in design/conclusion.

**Availability of data and materials**
All datasets, codes and results are available at http://dmb.tongji.edu.cn/supplementary-information/cpredictor3.

**About this supplement**
This article has been published as part of *BMC Systems Biology* Volume 11 Supplement 7, 2017: 16th International Conference on Bioinformatics (InCoB 2017): Systems Biology. The full contents of the supplement are available online at https://bmcsystbiol.biomedcentral.com/articles/supplements/volume-11-supplement-6.

**Authors' contributions**
JH and SG designed the research and revised the manuscript. YX developed the algorithm, carried out experiments, analyzed the experimental results, and drafted the manuscript. JG was involved in data analysis and revising the paper. All authors read and approved the final manuscript.

**Ethics approval and consent to participate**
Not applicable.

**Consent for publication**
Not applicable.

**Competing interests**
The authors declare that they have no competing interests.

**Author details**
[1]Department of Computer Science and Technology, Tongji University, Shanghai 201804, China. [2]Shanghai Key Lab of Intelligent Information Processing, and School of Computer Science, Fudan University, Shanghai 200433, China. [3]The institute of subtropical Agriculture, China Academy of Sciences, 444 Yuandaer Road, Mapoling, Changsha 410125, China. [4]The Bioinformatics Lab at Changzhou NO. 7 People's Hospital, Changzhou, Jiangsu 213011, China.

Published: 21 December 2017

**References**
1. Gavin AC, Aloy P, Grandi P, Krause R, Boesche M, Marzioch M, Rau C, Jensen LJ, Bastuck S, Dumpelfeld B, Edelmann A, Heurtier MA, Hoffman V, Hoefert C, Klein K, Hudak M, Michon AM, Schelder M, Schirle M, Remor M, Rudi T, Hooper S, Bauer A, Bouwmeester T, Casari G, Drewes G, Neubauer G, Rick JM, Kuster B, Bork P, Russell RB, Superti-Furga G. Proteome survey reveals modularity of the yeast cell machinery. Nature. 2006;440(7084):631–6.
2. Rigaut G, Shevchenko A, Rutz B, Wilm M, Mann M, Seraphin B. A generic protein purification method for protein complex characterization and proteome exploration. Nat Biotechnol. 1999;17(10):1030–2.
3. Barabasi AL, Oltvai ZN. Network biology: Understanding the cell's functional organization. Nat Rev Genet. 2004;5(2):101–15.
4. Bader GD, Hogue CW. An automated method for finding molecular complexes in large protein interaction networks. BMC Bioinformatics. 2003;4:2.
5. Pereira-Leal JB, Enright AJ, Ouzounis CA. Detection of functional modules from protein interaction networks. Proteins Struct Funct Bioinforma. 2004;54(1):49–57.
6. Nepusz T, Yu H, Paccanaro A. Detecting overlapping protein complexes in protein-protein interaction networks. Nat Methods. 2012;9(5):471–2.
7. Spirin V, Mirny LA. Protein complexes and functional modules in molecular networks. Proc Natl Acad Sci. 2003;100(21):12123–8.
8. Adamcsek B, Palla G, Farkas IJ, Derenyi I, Vicsek T. Cfinder: locating cliques and overlapping modules in biological networks. Bioinformatics. 2006;22(8):1021–3.
9. Zhang W, Zou X. A new method for detecting protein complexes based on the three node cliques. IEEE/ACM Trans Comput Biol Bioinforma. 2015;12(4):879–86.
10. Ulitsky I, Shamir R. Identification of functional modules using network topology and high-throughput data. BMC Syst Biol. 2007;1:8.
11. Ou-Yang L, Dai DQ, Zhang XF. Detecting protein complexes from signed protein-protein interaction networks. IEEE/ACM Trans Comput Biol Bioinforma. 2015;12(6):1333–44.
12. Lubovac Z, Gamalielsson J, Olsson B. Combining functional and topological properties to identify core modules in protein interaction networks. Proteins Struct Funct Bioinforma. 2006;64(4):948–59.
13. Cho YR, Hwang W, Ramanathan M, Zhang A. Semantic integration to identify overlapping functional modules in protein interaction networks. BMC Bioinformatics. 2007;8:265.
14. Leung HCM, Xiang Q, Yiu SM, Chin FYL. Predicting protein complexes from ppi data: A core-attachment approach. J Comput Biol. 2009;16(2):133–44.
15. Wu M, Li X, Kwoh CK, Ng SK. A core-attachment based method to detect protein complexes in ppi networks. BMC Bioinformatics. 2009;10:169.
16. Xu B, Guan J, Wang Y, Zhou S. Cpredictor2.0: Effectively detecting both small and large complexes from protein-protein interaction networks. In: Bourgeois A, Skums P, Wan X, Zelikovsky A, editors. Lecture Notes in Bioinformatics. vol. 9683. Berlin: Springer. 2016. p. 301–3.
17. Xu B, Wang Y, Wang Z, Zhou J, Zhou S, Guan J. An effective approach to detecting both small and large complexes from protein-protein interaction networks. BMC Bioinformatics. 2017;18(S12):19–28.
18. Tang X, Wang J, Liu B, Li M, Chen G. A comparison of the functional modules identified from time course and static ppi network data. BMC Bioinformatics. 2011;12(13):339–53.
19. Xu B, Guan J. From function to interaction: A new paradigm for accurately predicting protein complexes based on protein-to-protein interaction networks. IEEE/ACM Trans Comput Biol Bioinforma. 2014;11(4):616–27.
20. Altaf-Ul-Amin M, Shinbo Y, Mihara K, Kurokawa K, Kanaya S. Development and implementation of an algorithm for detection of protein complexes in large interaction networks. BMC Bioinformatics. 2006;7:207.
21. Cao B, Luo J, Liang C, Wang S, Song D. Moepga: A novel method to detect protein complexes in yeast protein–protein interaction networks

based on multiobjective evolutionary programming genetic algorithm. Comput Biol Chem. 2015;58:173–81.
22. Peng W, Wang J, Zhao B, Wang L. Identification of protein complexes using weighted pagerank-nibble algorithm and core-attachment structure. IEEE/ACM Trans Comput Biol Bioinforma (TCBB). 2015;12(1): 179–92.
23. Wang J, Peng X, Li M, Pan Y. Construction and application of dynamic protein interaction network based on time course gene expression data. Proteomics. 2013;13(2):301–12.
24. Zhang Y, Lin H, Yang Z, Wang J, Liu Y, Sang S. A method for predicting protein complex in dynamic ppi networks. BMC Bioinformatics. 2016;17(7):229.
25. Lei X, Wang F, Wu FX, Zhang A, Pedrycz W. Protein complex identification through markov clustering with firefly algorithm on dynamic protein–protein interaction networks. Inf Sci. 2016;329: 303–16.
26. Qi Y, Balem F, Faloutsos C, Klein-Seetharaman J, Bar-Joseph Z. Protein complex identification by supervised graph local clustering. Bioinformatics. 2008;24(13):250–8.
27. Yong CH, Maruyama O, Wong L. Discovery of small protein complexes from ppi networks with size-specific supervised weighting. BMC Syst Biol. 2014;8(S-5):S3.
28. Ruepp A, Brauner B, Dunger-Kaltenbach I, Frishman G, Montrone C, Stransky M, Waegele B, Schmidt T, Doudieu ON, Stumpflen V, Mewes HW. Corum: the comprehensive resource of mammalian protein complexes. Nucleic Acids Res. 2007;36(Database):646–50.
29. Wang JZ, Du Z, Payattakool R, Yu PS, Chen CF. A new method to measure the semantic similarity of go terms. Bioinformatics. 2007;23(10):1274–81.
30. von Luxburg U. A tutorial on spectral clustering. Stat Comput. 2007;17(4): 395–416.
31. Tu BP, Kudlicki A, Rowicka M, McKnight SL. Logic of the yeast metabolic cycle: Temporal compartmentalization of cellular processes. Science. 2005;310(5751):1152–8.
32. Krogan NJ, Cagney G, Yu HY, Zhong GQ, Guo XH, Ignatchenko A, Li J, Pu SY, Datta N, Tikuisis AP, Punna T, Peregrin-Alvarez JM, Shales M, Zhang X, Davey M, Robinson MD, Paccanaro A, Bray JE, Sheung A, Beattie B, Richards DP, Canadien V, Lalev A, Mena F, Wong P, Starostine A, Canete MM, Vlasblom J, Wu S, Orsi C, Collins SR, Chandran S, Haw R, Rilstone JJ, Gandi K, Thompson NJ, Musso G, St Onge P, Ghanny S, Lam M, Butland G, Altaf-Ui AM, Kanaya S, Shilatifard A, O'Shea E, Weissman JS, Ingles CJ, Hughes TR, Parkinson J, Gerstein M, Wodak SJ, Emili A, Greenblatt JF. Global landscape of protein complexes in the yeast saccharomyces cerevisiae. Nature. 2006;440(7084):637–43.
33. Collins SR, Kemmeren P, Zhao XC, Greenblatt JF, Spencer F, Holstege FCP, Weissman JS, Krogan NJ. Toward a comprehensive atlas of the physical interactome of saccharomyces cerevisiae. Mol Cell Proteomics. 2007;6(3): 439–50.
34. Kiemer L, Costa S, Ueffing M, Cesareni G. Wi-phi: a weighted yeast interactome enriched for direct physical interactions. Proteomics. 2007;7(6):932–43.
35. Mewes HW, Amid C, Arnold R, Frishman D, Guldener U, Mannhaupt G, Munsterkotter M, Pagel P, Strack N, Stumpflen V, Warfsmann J, Ruepp A. Mips: analysis and annotation of proteins from whole genomes. Nucleic Acids Res. 2004;32(SI):41–4.
36. Pu S, Wong J, Turner B, Cho E, Wodak SJ. Up-to-date catalogues of yeast protein complexes. Nucleic Acids Res. 2009;37(3):825–31.
37. Yu G, Li F, Qin Y, Bo X, Wu Y, Wang S. Gosemsim: an r package for measuring semantic similarity among go terms and gene products. Bioinformatics. 2010;26(7):976–8.
38. Pellegrini M, Baglioni M, Geraci F. Protein complex prediction for large protein protein interaction networks with the core&peel method. BMC Bioinformatics. 2016;17(12):372.
39. Li X, Wu M, Kwoh CK, Ng SK. Computational approaches for detecting protein complexes from protein interaction networks: a survey. BMC Genomics. 2010;11(1):3.
40. Luo J, Lin D. A cell-core-attachment approach for identifying protein complexes in ppi network. In: Natural Computation (ICNC), 2015 11th International Conference On. Piscataway: IEEE. 2015. p. 405–12.
41. Hu AL, Chan KC. Utilizing both topological and attribute information for protein complex identification in ppi networks. IEEE/ACM Trans Comput Biol Bioinforma. 2013;10(3):780–92.
42. Li XL, Foo CS, Tan SH, Ng SK. Interaction graph mining for protein complexes using local clique merging. Genome Inform. 2005;16(2):260–9.
43. Widita CK, Maruyama O. Ppsampler2: Predicting protein complexes more accurately and efficiently by sampling. BMC Syst Biol. 2013;7(Suppl 6):14.
44. King AD, Przulj N, Jurisica I. Protein complex prediction via cost-based clustering. Bioinformatics. 2004;20(17):3013–20.
45. Gao L, Zhou S. Group and graph joint sparsity for linked data classification. In: Schuurmans D, Wellman MP, editors. Proceedings of AAAI. Palo Alto: AAAI press. 2016.
46. Yang Y, Yan D, Wu H, Cheng J, Zhou S, Lui JCS. Diversified temporal subgraph pattern mining. In: Krishnapuram B, Shah M, Smola AJ, Aggarwal CC, Shen D, Rastogi R, editors. Proceedings of KDD. New York: ACM. 2016.
47. Zheng X, Wang Y, Tian K, Zhou J, Guan J, Luo L, Zhou S. Fusing multiple protein-protein similarity networks to effectively predict lncrna-protein interactions. BMC Bioinformatics. 2017;18(S12):11–18.