SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •
Special Focus on Biomolecular Network Analysis and Application

July 2016, Vol. 59 070103:1–070103:14
doi: 10.1007/s11432-016-5578-9

Detecting protein complexes from DPINs by density based clustering with Pigeon-Inspired Optimization Algorithm

Xiujuan LEI[1], Yulian DING[1] & Fang-Xiang WU[2,3*]

[1]*School of Computer Science, Shaanxi Normal University, Xi'an* 710062*, China;*
[2]*School of Mathematical Sciences, Nankai University, Tianjin* 300071*, China;*
[3]*Division of Biomedical Engineering, University of Saskatchewan, Saskatoon* SK S7N 5A9*, Canada*

Received March 11, 2016; accepted May 10, 2016; published online June 16, 2016

Abstract Detecting protein complexes is crucial to understand principles of cellular organization. Plenty evidences have indicated that sub-graphs with high density in protein-protein interaction (PPI) network, especially dynamic PPI network (DPIN), usually correspond to protein complexes. As a well-known density-based clustering algorithm, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) has been used in many areas due to its simplicity and the ability to detect clusters of different sizes and shapes. However, one of its limitations is that the performance of DBSCAN depends on two specified parameters ε and *MinPts*, where ε represents the maximum radius of a neighborhood from an observing point while *MinPts* means the minimum number of data points contained in such a neighborhood. In this article, we develop a new method named as P-DBSCAN to detect protein complexes in DPIN by using Pigeon-Inspired Optimization (PIO) Algorithm to optimize the parameters ε and *MinPts* in DBSCAN. The experiments on DIP and MIPS datasets show that P-DBSCAN outperforms the state-of-the-art methods for protein complex detection in terms of several criteria such as *precision*, *recall* and *f-measure*.

Keywords dynamic protein-protein interaction network (DPIN), pigeon-inspired optimization (PIO), protein complex, density based clustering, gene expression

Citation Lei X J, Ding Y L, Wu F-X. Detecting protein complexes from DPINs by density based clustering with Pigeon-Inspired Optimization Algorithm. Sci China Inf Sci, 2016, 59(7): 070103, doi: 10.1007/s11432-016-5578-9

1 Introduction

It is well known that proteins are involved in biological process in the form of protein complexes. Therefore, identifying protein complexes is important in understanding the cellular organizations and functional mechanisms. However, the experimental methods to discover protein complexes are costly and time-consuming. Fortunately, with the development of high throughput techniques such as yeast-two-hybrid [1] and proteome chips technologies [2], a large amount of protein-protein interaction (PPI) data such as DIP [3], MIPS [4] and SGD [5] have been generated and stored in public biological databases, which has greatly promoted the development of the computational methods for protein complex identification. However, due to the time-sensitivity of biological functions, there are vast amount of false-positives

* Corresponding author (email: faw341@mail.usask.ca)

exiting in the currently available PPI data [6]. Then, a lot of researchers analyze the PPI data combined with dynamic gene expression profiles [7] which creates Dynamic PPI Network (DPIN). Though protein complexes can be viewed as clusters in PPI networks, the traditional clustering methods do not perform so well in detecting protein complexes due to the topological characteristics of small-world and scale-free properties of PPI networks [8, 9]. Dense subgraphs in PPI networks generally correspond to protein complexes [10]. Thus, a series of protein complex detection methods emerged based on mining dense sub-graphs in PPI networks and namely density-based methods.

In recent years, some advanced density-based methods which detect protein complexes based on detecting cliques or dense subgraphs are sprung up continually. In 2005, Palla et al. [11] proposed the CPM algorithm, which mined adjacent k-cliques chains as protein complexes. In 2006, Adamcsek [12] developed a software package named CFinder, based on CPM. And at the same year, Altaf-ul-Admin [13] proposed the DPClus algorithm, which could detect overlapping modules by extending the neighbor nodes of function modules. IPCA [14] and SPICi [15] were proposed respectively in 2008 and 2010, and they are all seed-expanding methods, which identifies protein complexes by expanding seeds to density clusters by recursively adding the qualifying neighbours. In 2009, CMC [16] and COACH [17] were proposed. CMC algorithm first generated a weighted PPI network by an iterative scoring method and then identified protein complexes by removing or merging highly overlapped maximal cliques of this weighted PPI network based on their interconnectivity. COACH algorithm first mined dense sub-graphs as complex cores and then identified protein complex with its core and attachments separately. In 2010, Wang et al. [18] proposed CP-DR which modified CPM by adding distance restriction. Although the above methods were shown to effectively identify protein complexes, their results are sensitive to noisy data, which is still a challenge we have to face.

Density-Based Spatial Clustering of Applications with Noise(DBSCAN) proposed by Ester [19] implemented the density-based strategy with the considerations of noises in network data. It can efficiently discover the clusters with arbitrary size, shape and number in a large dataset. It is noise tolerant and independent of ordering of data objects. However, it needs two input parameters, ε—the radius of neighborhood and *MinPts*—the density threshold, which are domain specific and thus hard to be determined. Furthermore, DBSCAN uses the same global static parameters, which is not conducive to cluster data of varying densities. In this paper, we proposed a novel algorithm which employed Pigeon-Inspired Optimization Algorithm (PIO) [20], a swarm intelligence optimization algorithm, to adaptively determine two parameters of DBSCAN in every sub-network of DPIN according to the density based clustering algorithm.

Swarm intelligence optimization algorithms are commonly used to solve optimization problems by simulating the collective behavior of social insects. Their applications in detecting protein complexes are mainly divided into two directions. One is to imitate the intelligent behavior and set up corresponding clustering model, such as Bacteria Foraging Optimization (BFO) clustering model [21], PMABC-ACE clustering model [22] and predicting protein backbone based on Ant Colony Optimization Algorithm [23]. BFO clustering model is based on BFO mechanism and intuitionistic fuzzy set while PMABC-ACE is based on the propagating mechanism of artificial bee colony. The other one is to optimize the parameters of the clustering methods. Such as in 2014, Lei et al. [24] proposed F-MCL clustering model which automatically adjusts the parameters of Markov clustering method by using Firefly algorithm. In 2015, Lei et al. [25] proposed the ISHC clustering method by adopting the firefly algorithm to automatically determine the optimal threshold of the neighborhood radius of synchronization.

PIO algorithm is a novel swarm intelligence optimization algorithm, which was firstly proposed by Duan in 2014 [20]. Motivated by the homing characteristics of pigeons, two operators (map and compass operator and landmark operator) are designed. PIO algorithm was proven to be a worthy competitor to its well-known rivals [20], and widely applied to many fields such as constraining gliding trajectories [26], target detection approach for UAVs [27] and Proportion-Integral-Derivative (PID) controller design [28].

In this paper, we proposed a new method named P-DBSCAN which combines DBSCAN and PIO algorithm, for addressing the shortcoming of DBSCAN such as the difficulties of determining the parameters and the unreasonable strategy of using a set of global static parameters for a large varying dense database.

In a large DPIN network, we use PIO to optimize ε—the radius of neighborhood and *MinPts*—the density threshold in every sub-network. At last we checked the quality of every cluster and chose the optimal cluster results. For testing the performance of our algorithm, we compared P-DBSCAN with density based clustering methods such as DBSCAN [19], CFinder [12], DPClus [13], CMC [16], COACH [17], HC-PIN [29] as well as with traditional clustering methods such as MCL [30], F-MCL [24], RNSC [31].

The rest of the paper is organized as follows. In Section 2, after reviewing the PIO algorithm, basic DBSCAN and improved DBSCAN, our proposed P-DBSCAN are presented. In Section 3, experiment results and analysis are described and discussed. Section 4 is with the concluding remarks.

2 Method

2.1 Pigeon-inspired optimization algorithm

The homing pigeon has an inborn homing ability to find its way home over extremely long distances by using three homing tools: magnetic field, sun and landmarks. Inspired by the above homing behaviors of pigeons, a novel bio-inspired swarm intelligence optimizer which is named PIO has been proposed in 2014 [20]. In order to idealize some of the homing characteristics of pigeons, two operators are designed according to some rules. The map and compass operator model is presented based on magnetic field and sun while the landmark operator model is designed based on landmarks.

(1) Map and compass operator. In the map and compass operator, computer-generated pigeons are used. The position and the velocity of pigeon i can be denoted by X_i and V_i, which will update in each iteration in a d-dimensional search space. The new position and velocity of pigeon i at the tth iteration can be obtained [20] as

$$V_i(t) = V_i(t-1) \cdot \mathrm{e}^{-RT} + rand \cdot (X_g - X_i(t-1)), \tag{1}$$

$$X_i(t) = X_i(t-1) + V_i(t), \tag{2}$$

where R is the map and compass factor, *rand* is a random number generated from the uniform distribution on [0,1]. X_g is the current global best position which can be calculated by comparing all the positions among the whole swarm.

(2) Landmark operator. In the landmark operator, we assume the pigeons are still distant from the destination, and obviously they are unfamiliar to the landmarks. All pigeons are ranked according their fitness values. Then half of pigeons ($N_p/2$) is decreased according to

$$N_p(t) = \frac{N_p(t-1)}{2}. \tag{3}$$

We then find the central pigeon from the kept pigeons at the tth iteration, whose position (X_c) is the desirable destination. This can be described by (4). The new position of other pigeons can be calculated by (5).

$$X_c(t) = \frac{\sum X_i(t) \cdot fitness(X_i(t))}{N_p \sum fitness(X_i(t))}, \tag{4}$$

$$X_i(t) = X_i(t-1) + rand \cdot (X_c(t) - X_i(t-1)), \tag{5}$$

where $fitness(X_i(t))$ is the fitness value of the each pigeon in the swarm. For the minimum optimization problems, we can choose $fitness(X_i(t)) = \frac{1}{f_{\min}(X_i(t))+\delta}$ where δ is a small positive number. For maximum optimization problems, we can choose $fitness(X_i(t)) = f_{\max}(X_i(t))$.

2.2 DBSCAN and the application in PPI network of its improved version

2.2.1 *Basic DBSCAN algorithm*

DBSCAN, introduced by Ester et al. [19], is a clustering algorithm based on dense area for spatial datasets. It assumes that a cluster is a region in the data space with a high density, separated by regions

of low density in the data space. For each object in the dataset, the algorithm evaluates the number of neighbours that an object has by counting the number of objects that are within a proximity radius (minimum ε), specified as an input parameter of the algorithm. Based on this calculation, each object is labelled **core**, **border** or **noise** according to its number of neighbours. If a given object has more neighbours than a threshold value (*MinPts*), it is classified as **core**. All objects reachable from it, either directly (direct neighbours) or indirectly (neighbours of neighbours), are classified as **border**. All other objects, not reachable from any core, are classified as **noise**. DBSCAN starts with an arbitrary object P in data set D and retrieves all objects in D density-reachable from P with respect to ε and *MinPts*. If P is a core object, this procedure yields a cluster with respect to ε and *MinPts*. If P is a border object, no objects are density-reachable from P and P is assigned to noise temporarily. The algorithm ends with unvisited objects in the data set D.

DBSCAN is popular because of its capability of discovering clusters with arbitrary shapes without any preliminary information about the groups present in a dataset. What's more, DBSCAN can deal with the noise nodes and is insensitive to noise and does not need the number of clusters as input parameter and independent of ordering of data objects. In DBSCAN, the cluster number is derived from the parameters ε and *MinPts* which also define the density of the clusters to be found. However, there are also some limitations of DBSCAN. Firstly, the performance of clustering depends on two specified parameters. One is the maximum radius of a neighborhood from the observing point and the other is the minimum number of data points contained in such a neighborhood. It is difficult to estimate appropriate values of these two parameters for various datasets without enough prior knowledge. If the selection of parameters is inappropriate, it will affect the clustering results. Secondly, adjacent clusters of different densities cannot be properly identified due to the use of the global density parameters. The clustering of different data space needs different local density parameters, and the global parameters cannot portray the inner clustering structure very well. Thirdly, there is no overlap between two different clusters, so DBSCAN cannot be used to detect overlapping clusters.

2.2.2 *Application of DBSCAN to PPI networks*

In PPI networks, we cannot directly calculate the distance between two proteins by Euclidean distance. Therefore we introduce two properties about modularity analysis in PPI networks, node clustering coefficient and edge clustering coefficient.

The node clustering coefficient (NCC) [32] of node v is defined as

$$NCC(v) = \frac{2n_v}{k_v(k_v - 1)}. \tag{6}$$

In this coefficient, k_v is the degree of node v, n_v is the number of links connecting the k_v neighbors of node v to each other which indicates the number of triangles that pass through node v, and $\frac{k_v(k_v-1)}{2}$ is the total number of triangles that could pass through node v. The clustering coefficient reflects the degree of aggregation of nodes in the network.

The edge clustering coefficient (ECC) [33] of edge (v_i, v_j) is defined as

$$ECC(v_i, v_j) = \frac{z(v_i, v_j)}{\min(d_{i-1}, d_{j-1})}. \tag{7}$$

where $z(v_i, v_j)$ stands for the number of triangles which include the edge (v_i, v_j), and $\min(d_i - 1, d_j - 1)$ represents the maximum number of triangles that may contain edge (v_i, v_j). The edge clustering coefficient, *ECC*, which describes the similarity of nodes v_i and v_j. The higher the value is, the larger the probability that nodes v_i and v_j belong to the same protein complex is.

In the basic DBSCAN algorithm, the definition of core points simply depends on the degree. However, it's not enough for complex PPI networks because the clustering coefficient of a vertex in a graph measures the extent of the interconnectivity between the direct neighbors of the vertex. Generally, the clustering coefficient of v_i not only measures the local connectivity around this vertex, but also characterizes the

Table 1 The corresponding relationships between DBSCAN and PIO algorithms

The PIO algorithm	The clustering process of DBSCAN
The abscissa of pigeons' position	The value of parameter ε
The ordinate of pigeons' position	The value of parameter *MinPts*
The quality of the pigeon individual	The clustering result of P-DBSCAN
Pigeon adjust its flying direction by following the specific pigeon	Searching for the optimal value of two parameters
The destination of pigeon	The best clustering result of DBSCAN

Table 2 The symbols and their meanings in the algorithm of P-DBSCAN

Symbol	***maxiter***	x_i	N	***bestx***
Meaning	Maximal iterations of external loop	Location of pigeon i	The number of pigeons	Optimal pigeon
Symbol	***maxiter1***	v_i	x_c	min_ε
Meaning	Maximal number of compass operator	Speed of pigeon i	Location of center pigeon	Minimum of ε
Symbol	ECC	min_{MinPts}	t	max_ε
Meaning	Clustering coefficient of edge	Minimum of $MinPts$	Iterations of external loop	Maximum of ε
Symbol	NCC	max_{MinPts}	R	***bestf***
Meaning	Clustering coefficient of node	Maximum of $MinPts$	Map and compass factor	Optimal value
Symbol	ε	***bestcluster***	$MinPts$	
Meaning	The radius of neighborhood	Optimal cluster result	The density threshold	

effect of the node v_i [34]. So, we redefine the core point as a point p whose ε-neighborhood contains at least *MinPts* points and node clustering coefficient is greater than zero.

The DBSCAN algorithm needs to calculate the distance between any two points. However, in some PPI networks, such as DIP, DPIN are still unweighted. Then we use the aggregation coefficient of edge to be the weight of interactions. Thus the ε-neighborhood of a point can be redefined as the adjoining points whose value of *ECC* is larger than the parameter ε.

2.3 P-DBSCAN algorithm and code design

To overcome the defects of DBSCAN, we proposed a new clustering model by combining with PIO algorithm, named P-DBSCAN, to detect the protein complexes in PPI networks. In our proposed algorithm, to address the main limitation of DBSCAN that the appropriate values of parameters are difficult to be obtained, we introduced PIO into DBSCAN in order to find a pair of suitable parameters for getting better clustering results. Firstly, we set parameter ε as the abscissa of the location of pigeons and *MinPts* as the ordinate of the location of pigeons. We also respectively set the range for these two parameters and randomly locate pigeons in this area. Each paired value (ε, *MinPts*) is a pigeon's position. Then we calculate every pigeon's fitness by using DBSCAN with parameters ε and *MinPts*. After finishing the DBSCAN clustering process, an *f-measure* is received which is the fitness of pigeon, and will be introduced in Subsection 3.1. After all pigeons' fitness has been calculated, we get the relative attractiveness. Then we constantly update all pigeons' positions and calculate the fitness according to the map and compass operator. The iteration number meets the specified value for the map and compass operator, we continue to update all pigeons' positions and calculate the fitness according to the landmark operator. This process terminates and outputs the clustering result when the iteration number reaches the specified value for the process of landmark operator. The global best pigeon's location is the parameters that we use to get clustering at last and the parameters also correspond to the best clustering result.

To address the second limitation of DBSCAN that it is not reasonable to use global parameters in a large dataset, we set up dynamic model (defined in Subsection 3.2) which divides the whole PPI network into several smaller sub-networks, each for a dynamic time point. Then we find the best parameters obtained by PIO algorithm in different sub-networks. Although it cannot completely avoid the irrationality of

global parameters, it effectively attenuates this defect.

To address the third limitation that DBSCAN cannot detect the overlapping clusters, we make the following changes on the DBSCAN. If protein a is a border point, and node a not only connects with core node b but also with core node c which does not belong to the same cluster with b, we allow that node a not only belongs to the cluster b but also to the cluster c. Then the improved DBSCAN algorithm can detect the protein complexes with overlaps.

The corresponding relationships between DBSCAN and PIO are showed in Table 1.

Algorithm 1 P-DBSCAN

The meanings of the symbols in Table 2.

Input: DIP dataset;

Initialization: calculate the ***NCC*** of every protein node, ***ECC*** of every edge, randomly set the location of N pigeons abscissa between min_{ε} and max_{ε}, ordinate between min_{MinPts} and max_{MinPts};

while $t \leqslant$ ***maxiter*** **do**
  if $t \leqslant$ ***maxiter1*** **then**
    each pigeon represents a pair of value of ε and $MinPts$;
    for $i = 1$ to N **do**
      new_DBSCAN(ε_i,$MinPts_i$);
      calculate the $f-measure$ and return it as the $fitness$ of pigeon;
    end for
    record the optimal value ***bestf***, the best pigeon's location ***bestx*** and the ***bestcluster***;
    for $i = 1$ to N **do**
      $v_i(t)$=$v_i(t-1) \cdot \mathrm{e}^{-RT} + rand \cdot (bestx - x_i(t-1))$;
      $x_i(t)$=$x_i(t-1) + v_i(t)$;
    end for
    $\boldsymbol{t} = \boldsymbol{t} + 1$;
  else
    if $N > 1$ **then**
      calculate the *fitness* of every pigeon and rank them according their *fitness* value;
      record the optimal value ***bestf***, the best pigeon's location ***bestx*** and the ***bestcluster***;
      N=$\frac{N}{2}$;
      the half of pigeons which have a lower *fitness* value are removed;
      for $i = 1$ to N **do**
        $x_i(t)$=$\frac{\sum x_i(t) \cdot fitness(x_i(t))}{N \cdot \sum fitness(x_i(t))}$;
        caculate the position of the central pigeon;
        $x_i(t)$=$x_i(t-1) + rand \cdot (x_i(t) - x_i(t-1))$;
      end for
    end if
    $\boldsymbol{t} = \boldsymbol{t} + 1$;
  end if
end while

Output: the optimal clustering result ***bestcluster***

3 Experimental simulation and analysis

The operating environment of simulation experiment in this paper is: Windows 7 operating system, the double cores of Intel TM, the physical memory is 4 GB, the speed of processor is 3.1 GHz. The algorithm is run on the software Matlab R2011b.

3.1 Evaluating criteria of cluster results

Usually, clustering results are evaluated in terms of *precision*, *recall*, and *f-measure*. *Precision* [35] is the ratio of the maximal number of common nodes in both experimental results and the standard dataset to the number of nodes in experimental results. *Recall* [35], also termed the true positive rate or sensitivity, is the ratio of the number of proteins in both experimental results and standard dataset to the number of proteins in the standard dataset.

Given a clustering result $c = c_1, c_2, \ldots, c_k$ which is generated by the algorithm, and the standard clusters $S = s_1, s_2, \ldots, s_l$, for any predicted cluster c_i and known cluster s_j, the overlap score OS [36] is

defined as

$$OS(c_i, s_j) = \frac{|c_i \cap s_j|^2}{|c_i| \times |s_j|}, \tag{8}$$

where $|*|$ is the cardinality of the set $*$. The bigger the value of $OS(c_i, s_j)$ is, the more possible two clusters c_i and s_j are matched, and the more accurate predicted cluster c_i for s_j is. If it is bigger than a user-specified threshold, we can say that they are matched. Usually we set the threshold as 0.2 [36]. If $OS(c_i, s_j) = 1$, they are perfectly matched.

For the entire clustering result C which contains all clusters whose OS is bigger than the threshold, we calculate the *precision*, *recall* and *f-measure* by the following formula:

$$precision = \frac{|C \cap S|}{|C|}, \tag{9}$$

$$recall = \frac{|C \cap S|}{|S|}. \tag{10}$$

In general, a large module has the higher *recall* value, while a smaller module has higher *precision*. Therefore, in order to balance the *precision* and *recall* values, we can define the *f-measure* [35] value as

$$f\text{-}measure = \frac{2(precision \cdot recall)}{precision + recall}. \tag{11}$$

3.2 Dynamic model construction

In recent years, researchers have tried to inject dynamic information into static PPI networks. The three-sigma method [36] is a successful example. The three-sigma method identifies active time points of each protein in a cellular cycle, where three-sigma principle is used to compute an active threshold for each gene according to its dynamic (time-series) expression profile.

Let $EV_i(p)$ be the expression value of gene p at time point i, $\mu(p)$ be the algorithmic mean of its expression values over time points 1 to n and $\sigma^2(p)$ be the standard deviation of its expression values.

$$\mu(p) = \frac{\Sigma_{i=1}^{n} EV_i(p)}{n}, \tag{12}$$

$$\sigma^2(p) = \frac{\Sigma_{i=1}^{n} (EV_i(p) - \mu(p))^2}{n-1}, \tag{13}$$

$$F(p) = \frac{1}{1 + \sigma^2(p)}. \tag{14}$$

$F(p)$ can reflect the fluctuation of the expression curve of gene p. It is clear that the higher $\sigma^2(p)$ is, the smaller F is. The value range of F is from 0 to 1.

A protein is considered to be active at the time points with expression values that are above or equal to its active threshold, denoted as *Active_Th(p)*. For gene p, its active threshold can be calculated by using the three-sigma method as follows:

$$Active\_Th(p) = S_1(p) \times F(p) + S_2(p) \times (1 - F(p)) = \mu(p) + 3\sigma(p)(1 - F(p)), \tag{15}$$

where $S_1(p) = \mu(p)$ and $S_2(p) = \mu(p) + 3\sigma(p)$.

Then we can use the *Active_Th* to identify the active time points of proteins and construct a DPIN. In the DPIN, there is a sub-network at each time point, such as the DPIN model in [36], which divides the static PPI network into n sub-networks according to n time points.

3.3 Experimental result analysis

3.3.1 *Experimental datasets*

In this paper, we use the unweighted PPI data of S.cerevisiae from DIP database (version DIP 20140427) [3] and weighted PPI data from MIPS database [4] to investigate the performance of our algorithm. The

Table 3 The number of proteins and interactions in each sub-network of the DPIN contains

Timestamps	1	2	3	4	5	6
Proteins	797	941	796	623	610	530
Interations	981	1444	1188	745	750	646
Timestamps	7	8	9	10	11	12
Proteins	493	944	1090	591	661	461
Interactions	573	1705	2185	856	974	526

Table 4 General properties of the gold standard protein complexes

Datasets	♯ Complexes	Protein coverage	Max size	Overlapping complex pairs	Avg size	Min size
CYC2008	408	1628	81	151	4.71	2
MIPS	162	1171	95	401	14.93	4

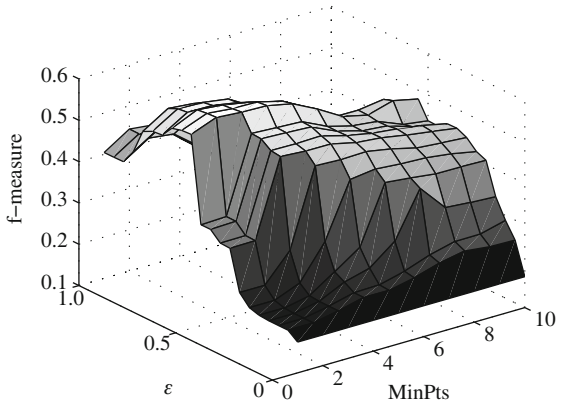

Figure 1 Performance of DBSCAN on protein complex detection with different values of ε and *MinPts*.

static unweighted DIP contains 4995 proteins and 21554 interactions after removing self-interactions and repeated ones. The average degree is 8.6268. The static MIPS dataset consists of 1376 proteins and 6880 interactions.

Then we download dynamic gene expression data set GSE3431 about the yeast metabolic cycle from GEO dataset [37]. This dataset includes 6777 genes that cover 95% proteins in the static DIP network. By using methods for construction, we get the DPIN of DIP which contains 12 static PPI sub-networks [24] at 12 time points. Different sub-networks have different scales, shown in Table 3.

Two benchmark complex sets which are respectively derived from CYC2008 [38] and MIPS [39] are chosen as our gold standard. The general properties of them are shown in Table 4.

3.3.2 *Parameter selection*

In P-DBSCAN algorithm, the parameters are the maximum values of ε and minimum value of *MinPts*, which are also the domain of pegions' position. The codomain of ε is from 0 to 1. If the values of ε and *MinPts* are too large, there is no meaningful cluster that satisfies the P-DBSCAN algorithm. For example, when we set ε to 0.95, *MinPts* to 8, there is no meaningful cluster that can be identified from the DIP network. On the other hand, if the values of two parameters are too small, the cluster will include too many proteins and the precision of clusters is very low. So the setting of these two parameters directly influence the performance of P-DBSCAN. In this article, the maximum value of ε and minimum value of *MinPts* are set according to Figure 1 for DIP database. The x-axis denotes the values of ε which range from 0 to 1, the y-axis denotes the values of *MinPts* which range from 0 to 10, and the z-axis denotes the values of *f-measure* which are computed from DBSCAN by the corresponding parameters of ε and *MinPts*. The colour of this figure changes from dark grey to light grey as the *f-measure* increases. When the value of *f-measure* is very low, the corresponding area is dark grey. The dark grey changes to light

Table 5 The parameters' setting of each algorithm on DIP and MIPS datasets

Algorithms	Parameters	DIP	MIPS
CMC	*Overlap threshold, Merge threshold, min size*	0.8, 0.2, 4	0.5, 0.15, 20
CFinder	*K-clique template*	4	4
RNSC	*Shuffling diversification length,*	9	9
	Diversification frequency, Experiments number,	20, 3	20, 3
	Naive stopping tolerance, Scaled stopping tolerance,	20, 5	20, 5
	Tabu length, Tabu tolerance	10, 1	100, 5
MCL	*Inflation*	2.0	2.3
F-MCL	*Population size, iteration count, step length,*	5, 5, 0.05	5, 5, 0.05
	Light absorption coefficient, maximal attractiveness	1.0, 1.0	1.0, 1.0
COACH	*omega*	0.225	
HC-PIN	*lamada, min size*	0.5, 2	0.5, 2
DPClus	*cp value, density value*	0.8, 0.5	
DBSCAN	ε, *MinPts*	0.5, 4	0.3, 4
P-DBSCAN	*Min_*ε, *Max_*ε, *Min_MinPts, Max_MinPts,*	0.2, 0.9, 2, 8	0.2, 0.7, 2, 8
	R, Pig_N, Iter1, Iter2	0.3, 5, 30, 20	0.3, 10, 30, 20

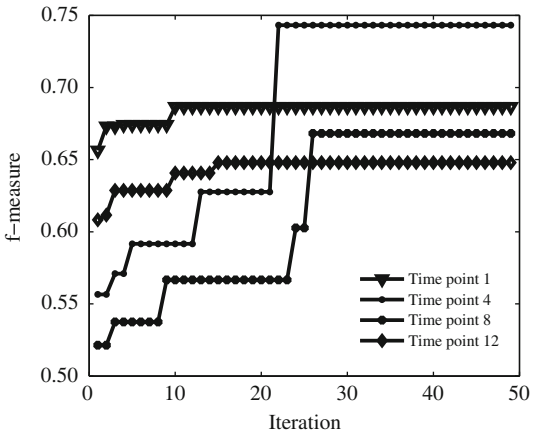

Figure 2 The convergence curves of P-DBSCAN algorithm operated for four sub-networks of dynamic DIP database.

grey, when the *f-measure* becomes larger. From Figure 1, we observe that the *f-measure* increases initially as the value of ε (or *MinPts*) increases and decreases after reaching the maximum for a fixed value of *MinPts* (or ε). Then we chose the area of light grey as the pegions' location in P-DBSCAN. Overall, we find that the optimal values of $\varepsilon \in [0.2, 0.9]$ and $MinPts \in [2, 8]$. So the $Min\_\varepsilon = 0.2$, $Max\_\varepsilon = 0.9$, $Min\_MinPts = 2$ and $Max\_MinPts = 8$.

The rest of the parameters of DBSCAN and P-DBSCAN are set as in Table 5. In Table 5, the radius of neighborhood ε and the radius of neighborhood *MinPts* in DBSCAN algorithm are set according to [19]. In P-DBSCAN, the map and compass factor R, the number of pigeon *pig-N*, the interactions of map and compass operator *Iter1* and the interactions of mark operator *Iter2* are set according to [20]. The optimal parameters of comparative algorithms on two different test data sets are listed in Table 5.

3.3.3 *Parameter optimization process*

In this section, we show the parameter optimization process and the optimal parameters based on DIP database. As pigeons' location in P-DBSCAN algorithm changes, the performance of clustering is also constantly changing. As shown in Figure 2, with the increasing of the number of iterations, a pigeon constantly updates its position and the performance of clustering get better and better. At last, when the pigeon finds its destination, the performance of clustering also achieves the best. The most appropriate

Table 6 The value of parameters, *precision*, *recall*, and *f_measure* corresponding to the best clustering in every sub-network of DIP

Timestamps	1	2	3	4	5	6
ε	0.5595	0.6654	0.5669	0.6322	0.6843	0.6364
MinPts	4	2	4	2	2	3
precision	0.7391	0.7671	0.7467	0.8276	0.7317	0.7872
recall	0.6415	0.5545	0.5385	0.6761	0.5882	0.4512
f_measure	0.6869	0.6437	0.6257	0.7442	0.6522	0.5736
Timestamps	7	8	9	10	11	12
ε	0.4793	0.5809	0.5123	0.5486	0.2436	0.6521
MinPts	3	4	5	2	3	3
precision	0.5490	0.6311	0.6259	0.6364	0.6842	0.6744
recall	0.5714	0.7196	0.5823	0.7206	0.6964	0.6098
f_measure	0.5600	0.6725	0.6033	0.6759	0.6842	0.6472

Table 7 The analysis of seven protein complexes in clustering result of dynamic DIP database

Serial number	**Real complexes contained proteins**	**DBSCAN** correct proteins	**DBSCAN** wrong proteins	**P-DBSCAN** correct proteins	**P-DBSCAN** wrong proteins
1	YBL037W YJR005W	YBL037W YJR005W		YBL037W YJR005W	
	YJR058C YNR056C	YJR058C YNR056C	YMR119W	YJR058C YNR056C	
2	YGL153W YLR191W	YGL153W YLR191W		YGL153W YLR191W	
	YNL214W	YNL214W		YNL214W	YAR042W
3	YGL075C YLR457C	YGL075C YLR457C	YHL006C	YGL075C YLR457C	
	YPL255W	YPL255W	YLR392C	YPL255W	YLR392C
4	YBR103W YCR033W	YBR103W YCR033W		YBR103W YCR033W	
	YGL194C YIL112W	YGL194C YIL112W	YMR173C	YGL194C YIL112W	
	YKR029C YDR155C	YKR029C			
	YOL068C				
5	YBR102C YER008C	YBR102C YER008C		YBR102C YER008C	
	YIL068C YJL085W	YIL068C YJL085W		YIL068C YJL085W	
	YPR005W YLR166C	YPR005W		YPR005W YLR166C	
	YGL233W YPR166C				
6	YDR394W YLR457C	YDR394W YLR457C	YHL006C	YDR394W YLR392C	
	YPL255W	YPL255W	YLR392C	YPL255W	YLR392C
7	YHR158C YAL024C	YHR158C YAL024C		YHR158C YAL024C	
	YGR238C	YGR238C	YHR133C	YGR238C	

parameters and the performance of clustering about 12 sub-networks are shown in Table 6. It is obvious that different sub-networks have different optimal parameters and the performances of the clustering are also different. It can be seen that the disadvantage of DBSCAN which uses global density parameters makes it not suitable in big varying density datasets.

3.3.4 *Clustering analysis and overlapping protein complexes*

In this section, we analyze the wrong and correct proteins in the cluster modules and show the overlapping of protein complexes. All those results come from dynamic DIP database. In Table 7, we random choose seven protein complexes in the DBSCAN and P-DBSCAN to analyze the performance of clustering results. It is obvious that the clustering result from P-DBSCAN has less wrong proteins and more correct proteins than DBSCAN. The result from P-DBSCAN is closer to the standard dataset.

In order to more clearly show the clustering effect, we visualize the detected protein complexes. In

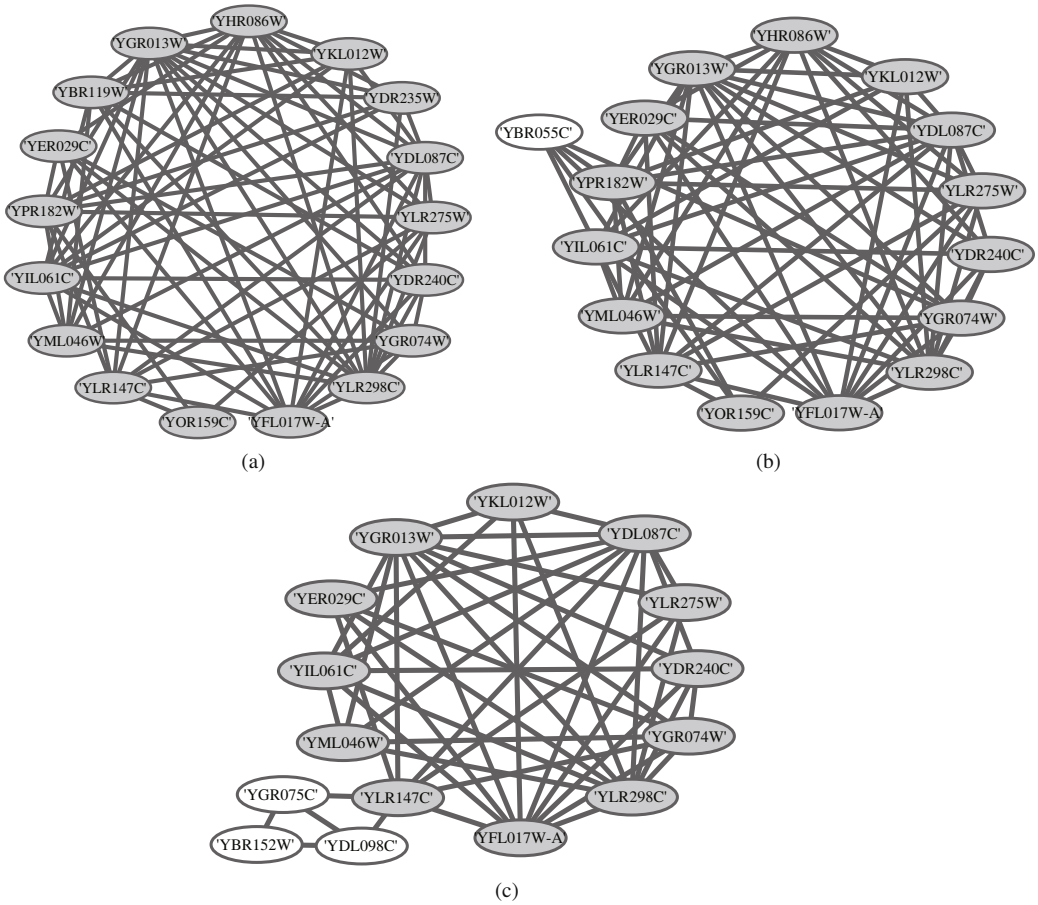

Figure 3 Visualization of a protein complex in the standard dataset and corresponding to the protein complex from P-DBSCAN and DBSCAN. (a) Standard; (b) P-DBSCAN; (c) DBSCAN.

this paper, we use the software Cytoscape [40] to visualize a protein complex and generated P-DBSCAN (Figure 3(b)), DBSCAN (Figure 3(c)) and the corresponding cluster in the standard dataset (Figure 3(a)). In Figure 3(b), the fifteen proteins with background are the correct proteins and the white protein YBR055C is the wrong protein. It misses proteins YBR119W and YDR235W. In Figure 3(c), there are twelve proteins with background are the correct proteins while the white proteins YDL098C, YBR152W, YGR075C are the wrong proteins. It misses five proteins. So, we can see that the P-DBSCAN algorithm can identify more correct proteins and less wrong proteins than DBSCAN.

Figure 4 shows two overlapping protein complexes by P-DBSCAN. The shaded areas represent the protein complex detected by P-DBSCAN, the white nodes are the wrong nodes and the dark grey nodes belong to both protein complexes in the standard.

3.3.5 *Performance comparison*

In this section, we compare P-DBSCAN clustering algorithm with other algorithms on both unweighted DIP and weighted MIPS.

We first compare P-DBSCAN with the basic DBSCAN algorithm [19], the typical density based clustering algorithm CMC [16], CFinder [12], DPClus [13], COACH [17] and the typical clustering algorithms RNSC [31], MCL [30], F-MCL [24], HC-PIN [29]. All those algorithms are compared with each other on dynamic DIP database using the CYC2008 gold standard. The algorithms which need weighted networks use the *ECC* to weight the edges. The performances of all clustering algorithms are reported in Table 8 which contains the category of each algorithm, the number of predicted protein complexes, the average size of protein complexes, the number of coverage numbers, the *precision*, *recall*, *f-measure*, whether they need weighted network and whether it can detect overlapping protein complexes. The average size of

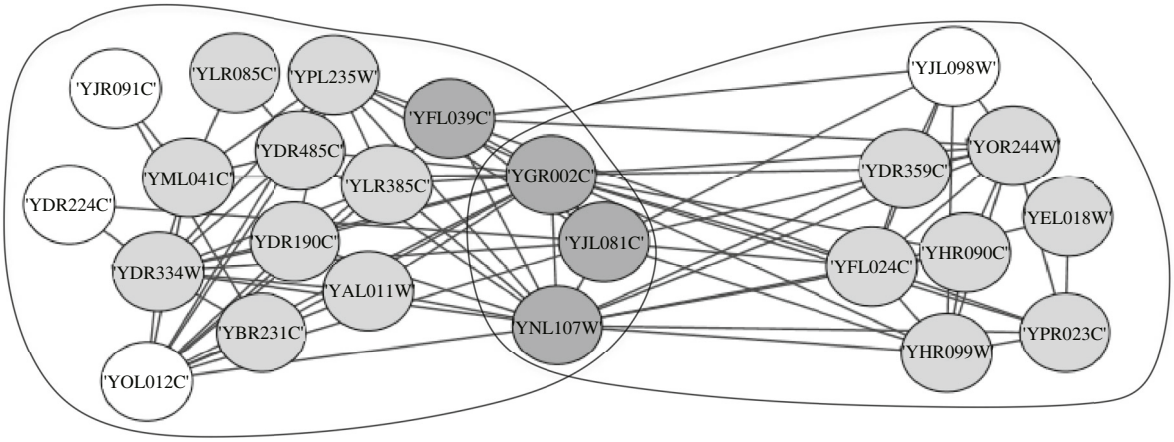

Figure 4 Two overlapping protein complexes by P-DBSCAN.

Table 8 The performance comparisons of various protein complex detection algorithms on DIP dataset using the CYC2008 gold standard

Algorithms	Category	Clusters	*Avg. size*	*Coverage*	*Precision*	*Recall*	*F-measure*	Weighted	Overlapping
CMC	Density	1263	4.39	2048	0.31	0.59	0.4064	yes	yes
CFinder	Density	609	6.18	2135	0.5607	0.3528	0.4331	no	yes
RNSC	Partition	549	3.89	2133	0.4067	0.4696	0.4359	yes	no
MCL	Flow	623	6.57	4096	0.3569	0.3879	0.3717	yes	no
F-MCL	Swarm-Flow	1588	4.62	2802	0.6804	0.554	0.6122	yes	yes
COACH	Core	903	3.89	2133	0.5038	0.5	0.5019	no	yes
HC-PIN	Hierarchy	273	5.73	1564	0.4316	0.3149	0.3641	no	no
DPClus	Density	827	5.28	3258	0.43	0.507	0.4653	no	yes
DBSCAN	Density	492	6.26	1817	0.5463	0.5744	0.56	yes	no
P-DBSCAN	**Swarm-Density**	**642**	**4.98**	**1652**	**0.6888**	**0.6081**	**0.6459**	**yes**	**yes**

P-DBSCAN is much more close to the average size of gold standard than other density-based clustering algorithms. The complexes number of protein complexes generated from P-DBSCAN is larger than that of from DBSCAN while the average size of protein complexes generated from P-DBSCAN is smaller than that from DBSCAN due to that P-DBSCAN not only considers the degree of vertices but also measures the extent of the interconnectivity between the direct neighbors of the vertexes. In Table 8, P-DBSCAN has the highest value in *precision*, *recall* and *f-measure*. In our algorithm, we use PIO algorithm to optimize the parameters of DBSCAN. After optimization, the P-DBSCAN algorithm produces the clustering results based on the optimal parameters. So it has a much better performance than the basic DBSCAN algorithm. What's more, P-DBSCAN also outperforms other typical clustering algorithms.

To further show P-DBSCAN's performance, P-DBSCAN is also performed on weighted PPI network from MIPS. To adopt P-DBSCAN for weighted networks, we replace ECC values with the weights of interactions in MIPS network. Since COACH and DPClus can only be implemented on unweighted PPI networks, we compare our method with CMC [16], CFinder [12], RNSC [31], MCL [30], F-MCL [24] and HC-PIN [29]. The results show that the good performance of P-DBSCAN is not limited to the dynamic networks, all those algorithms are compared with each other on weighted static MIPS network using the MIPS gold standard. The relevant results of the algorithms such as CMC, CFinder, HC-PIN and MCL is obtained from [16], the result of RNSC can be seen in [25] and the other results is coming from our experiments. In Table 9, the cluster number of P-DBSCAN is still larger than DBSCAN and the average size of clusters is smaller than DBSCAN. What's more, P-DBSCAN outperforms other typical clustering algorithms.

Table 9 The performance comparisons of various protein complex detection algorithms on weighted MIPS datasets using the MIPS gold standard

Algorithms	Category	Clusters	*Avg size*	*Precision*	*Recall*	*F-measure*	Weighted	Overlapping
CMC	Density	121	9.42	0.339	0.315	0.327	yes	yes
CFinder	Density	95	11.77	0.389	0.302	0.34	no	yes
RNSC	Partition	88	8.67	0.407	0.469	0.435	yes	no
MCL	Flow	116	10.31	0.353	0.315	0.333	yes	no
F-MCL	Swarm-Flow	141	6.5	0.462	0.416	0.437	yes	yes
HC-PIN	Hierarchy	119	7.76	0.225	0.706	0.341	no	no
DBSCAN	Density	78	11.95	0.443	0.347	0.389	yes	no
P-DBSCAN	**Swarm-Density**	**93**	**9.13**	**0.595**	**0.373**	**0.459**	**yes**	**yes**

4 Conclusion

With the development of high throughput techniques, static and dynamic PPIs are increasing fast and available conveniently. At the same time, the research on identifying protein complexes are sprung up. In this article, we have proposed a new approach called P-DBSCAN for identifying protein complexes in DPIN, which introduced the PIO optimization algorithm to overcome the shortcomings of DBSCAN that (1) the performance of the algorithm depends on two specified parameters; (2) it not reasonable to use global parameters in a large dataset; and (3) it cannot detect overlapping clusters. To address the shortcoming (1), we have used PIO algorithm to adaptively select parameters; To address shortcoming (2), we have chosen different parameters for different sub-networks; To address shortcoming (3), we have changed the way to mark proteins and as a result, predicted protein complexes can have some overlap. After effectively addressing the shortcomings of DBSCAN, the P-DBSCAN clustering algorithm outperforms other clustering algorithms in detecting protein complexes in terms of *precision*, *recall*, and *f-measure*. The limitation of P-DBSCAN is that it is complicated and time consuming. As future work, it would be necessary to enhance the efficiency of P-DBSCAN by designing some parallel algorithms or by running it on GPU.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61502290, 61401263), Industrial Research Project of Science and Technology in Shaanxi Province (Grant No. 2015GY016), Fundamental Research Funds for the Central Universities, Shaanxi Normal University (Grant No. GK201501008), Graduated Student Innovation Foundation of Shaanxi Normal University (Grant No. 2015CXS030) and China Postdoctoral Science Foundation (Grant No. 2015M582606).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Uetz P, Giot L, Cagney G, et al. A comprehensive analysis of protein-protein interactions in Saccharomyces cerevisiae. Nature, 2000, 403: 623–627
2 Zhu H, Bilgin M, Bangham R, et al. Global analysis of protein activities using proteome chips. Science, 2001, 293: 2101–2105
3 Xenarios I, Salwnski L, Duan X J, et al. DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions. Nucl Acids Res, 2002, 30: 303–305
4 Güldener U, Münsterkötter M, Kastenmller G, et al. CYGD: the comprehensive yeast genome database. Nucl Acids Res, 2005, 33: 364–368
5 Cherry J M. SGD: Saccharomyces Genome Database. Nucl Acids Res, 1998, 26: 73–79
6 Montanez G, Cho Y R. Predicting false positives of protein-protein interaction data by semantic similarity measures. Curr Bioinform, 2013, 8: 339–346
7 Li M, Zheng R, Zhang H, et al. Effective identification of essential proteins based on priori knowledge, network topology and gene expressions. Methods, 2014, 67: 325–333
8 Watts D J, Strogatz S H. Collective dynamics of 'small-world' networks. Nature, 1998, 393: 440–442
9 Antonio S, Paul O M. Small-world network approach to identify key residues in protein-protein interaction. Proteins, 2005, 58: 672–682

10 Rives A W, Galitski T. Modular organization of cellular networks. Proc Nat Acad Sci USA, 2003, 100: 1128–1133
11 Palla G, Dernyi I, Farkas I J, et al. Uncoverring the overlapping community structure of complex networks in nature and society. Nature, 2005, 435: 814–818
12 Adamcsek B, Palla G, Farkas I, et al. CFinder: locating cliques and overlapping modules in biological networks. Bioinformatics, 2006, 22: 1021–1023
13 Altaf-Ul-Amin M, Shinbo Y, Mihara K, et al. Development and implementation of an algorithm for detection of protein complexes in large interaction networks. BMC Bioinform, 2006, 7: 207–228
14 Li M, Chen J, Wang J, et al. Modifying the DPClus algorithm for identifying protein complexes based on new topological structures. BMC Bioinform, 2008, 9: 398–413
15 Peng J, Mona S. SPICi: a fast clustering algorithm for large biological networks. Bioinformatics, 2010, 26: 1105–1111
16 Liu G, Wong L, Chua H N. Complex discovery from weighted PPI networks. Bioinformatics, 2009, 25: 1891–1897
17 Leung H C M, Xiang Q, Yiu S M, et al. Predicting protein complexes from PPI data: a core-attachment approach. J Comput Biol, 2009, 16: 133–144
18 Wang J X, Liu B B, Li M, et al. Identifying protein complexes from interaction networks based on clique percolation and distance restriction. BMC Genom, 2010, 11: S10–S24
19 Ester M, Kriegel H P, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, 1996. 226–231
20 Duan H B, Qiao P X. Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning. Int J Intell Comput Cybern, 2014, 7: 24–37
21 Lei X J, Wu S, Ge L, et al. Clustering and overlapping modules detection in PPI network based on IBFO. Proteomics, 2013, 13: 278–290
22 Lei X J, Tian J F, Ge L, et al. The clustering model and algorithm of PPI network based on propagating mechanism of artificial bee colony. Inform Sci, 2013, 247: 21–39
23 Lv Q, Wu H J, Wu J Z, et al. A parallel ant colonies approach to de novo prediction of protein backbone in CASP8/9. Sci China Inf Sci, 2013, 56: 108103
24 Lei X J, Wang F, Wu F X, et al. Protein complex identification through Markov clustering with firefly algorithm on dynamic proteinCprotein interaction networks. Inf Sci, 2016, 329: 303–316
25 Lei X J, Ying C, Wu F X, et al. Clustering PPI data by combining FA and SHC method. BMC Genom, 2015, 16: S3–S12
26 Zhao J, Zhou R. Pigeon-inspired optimization applied to constrained gliding trajectories. Nonlinear Dyn, 2015, 82: 1781–1795
27 Li C, Duan H B. Target detection approach for UAVs via improved Pigeon-inspired Optimization and Edge Potential Function. Aerosp Sci Technol, 2014, 39: 352–360
28 Sun H, Duan H B. PID controller design based on Prey-Predator Pigeon-Inspired Optimization algorithm. In: Proceedings of the International Conference on Mechatronics and Automation, Tianjin, 2014. 1416–1421
29 Wang J X, Li M, Chen J, et al. A fast hierarchical clustering algorithm for functional modules discovery in protein interaction networks. IEEE/ACM Trans Comput Biol Bioinform. 2011, 8: 607–620
30 van Dongen S. Graph clustering by flow simulation. Dissertation for Doctoral Degree. Center for Math and Computer Science (CWI), University of Utrecht. 2000
31 King A D, Przulj N, Jurisica I. Protein complex prediction via cost-based clustering. Bioinformatics, 2004, 20: 3013–3020
32 Zhang A D. Protein interaction networks. New York: Cambridge University Press, 2009
33 Radicchi F, Castellano C, Cecconi F, et al. Defining and identifying communities in networks. Proc Nat Acad Sci USA, 2004, 101: 2658–2663
34 Washburn M P, Wolters D, Yates J R. Large-scale analysis of the yeast proteome by multidimensional protein identification technology. Nat Biotechnol, 2001, 19: 242–247
35 Cho Y R, Hwang H, Ramanathan M, et al. Semantic integration to identify overlapping functional modules in protein interaction networks. BMC Bioinform, 2007, 8: 265–277
36 Wang J X, Peng X Q, Li M, et al. Construction and application of dynamic protein interaction network based on time course gene expression data. Proteomics, 2013, 13: 301–312
37 Tu B P, Kudlicki A, Rowicka M, et al. Logic of the yeast metabolic cycle: temporal compartmentalization of cellular processes. Science, 2005, 310: 1152–1158
38 Pu S, Wong J, Turner B, et al. Up-to-date catalogues of yeast protein complexes. Nucl Acids Res 2009, 37: 825–831
39 Mewes H W, Amid C, Arnold R, et al. MIPS: analysis and annotation of proteins from whole genomes. Nucl Acids Res, 2004, 32: 41–44
40 Tang Y, Li M, Wang J X. CytoNCA: a cytoscape plugin for centrality analysis and evaluation of protein interaction networks. Biosystems, 2015, 127: 67–72