Evolutionary Graph Clustering for Protein Complex Identification

Tiantian He, and Keith C.C. Chan

Abstract—This paper presents a graph clustering algorithm, called EGCPI, to discover protein complexes in protein-protein interaction (PPI) networks. In performing its task, EGCPI takes into consideration both network topologies and attributes of interacting proteins, both of which have been shown to be important for protein complex discovery. EGCPI formulates the problem as an optimization problem and tackles it with evolutionary clustering. Given a PPI network, EGCPI first annotates each protein with corresponding attributes that are provided in Gene Ontology database. It then adopts a similarity measure to evaluate how similar the connected proteins are taking into consideration the network topology. Given this measure, EGCPI then discovers a number of graph clusters within which proteins are densely connected, based on an evolutionary strategy. At last, EGCPI identifies protein complexes in each discovered cluster based on the homogeneity of attributes performed by pairwise proteins. EGCPI has been tested with several real data sets and the experimental results show EGCPI is very effective on protein complex discovery, and the evolutionary clustering is helpful to identify protein complexes in PPI networks. The software of EGCPI can be downloaded via: https://github.com/hetiantian1985/EGCPI.

Index Terms—Graph clustering, evolutionary clustering, clustering algorithms, protein-protein interaction networks, protein complex identification

1 Introduction

A protein complex is a biomolecule that contains a number of proteins connecting with each other to perform cellular functions [1]. Due to its important role in the understanding of cellular organizations and functions, such as replication, transcription and the control of gene expression, etc. [2], much effort has recently been put into discovering protein complexes in the protein-protein interaction (PPI) networks.

To identify protein complexes on a large scale, time-consuming laboratory experiments, such as affinity purification (AP) followed by mass spectrometry (MS), have to be performed [3], [4]. Though effective, AP/MS cannot be considered as an efficient method as it requires a number of different steps to be carried out with different baits every time [5].

To minimize the laborious trials-and-errors procedures, some attempts to identify protein complexes computationally have recently been made [6] [7]. These computational approaches are, by and large, developed based on different graph clustering algorithms. Given a PPI network represented as a graph that contains vertices representing proteins and edges representing protein interactions, these algorithms can discover sub-graphs or clusters based on different topological properties such as density, k-cores, core-attachment structures and peripheries, etc.

Due to some evidence of proteins in protein complexes tending to interact more with each other, graph clustering algorithms, that aim at identifying densely connected sub-graphs by considering graph modularity and density [44] [45], are usually used to identify protein complexes in PPI networks [8]. For example, one of the most popular graph clustering algorithm that is used for protein complex identification is MCODE [9]. By taking into consideration local neighborhood density, MCODE can detect sub-graphs that contain densely connected vertices in a PPI network graph.

Other than MCODE, another graph clustering algorithm called the MCL algorithm [10] has also been used for the identification of protein complexes. The MCL algorithm also discovers densely connected sub-graphs except that it does so by making use of a random-walk approach through simulating flow expansion and contraction [11] using what are called expansion and inflation operators. A number of dense clusters can be extracted from the incidence matrix of a PPI network graph when MCL achieves convergence.

Another dense sub-graph identification algorithm, called RNSC [12], can find protein complexes in a given PPI network graph by graph partitioning. RNSC attempts to find an optimal set of partitions of a PPI network graph by employing different cost functions that are defined in terms of edge density, cluster size and functional homogeneity. The graph partitions that are identified can correspond well with protein complexes. An algorithm that is similar to the RNSC is proposed in [13]. The algorithm

- *Tiantian He is with the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong SAR. E-mail: csthe@comp.polyu.edu.hk.*
- *Keith C.C. Chan is with the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong SAR. E-mail: cskc-chan@comp.polyu.edu.hk.*

***Please provide a complete mailing address for each author, as this is the address the 10 complimentary reprints of your paper will be sent*

*Please note that all acknowledgments should be placed at the end of the paper, before the bibliography (**note that corresponding authorship is not noted in affiliation box, but in acknowledgment section**).*

xxxx-xxxx/0x/$xx.00 © 200x IEEE Published by the IEEE Computer Society

also attempts to find protein complexes by partitioning a PPI network graph but it uses a minimum vertex-cut to identify cluster boundaries so that the vertices in each graph partition tend to connect more with other vertices that are in the same partition.

In [41], an algorithm called SPICi is proposed to identify protein complex by considering local density of a PPI network. Comparing with other density-based graph clustering algorithms, SPICi can be shown to be a very fast algorithm for protein complex identification.

In [47], an algorithm called DCU is proposed to detect protein complexes utilizing an uncertain graph model. DCU utilizes two measures when identifying protein complexes. One of them is called the *relative degree* measure. It is used to determine whether or not a particular protein belongs to a particular sub-graph. The other is called the *expected density* measure. It is used to determine whether or not a dense sub-graph satisfies with the minimum density to be identified as a protein complex.

Other than edge density, some graph clustering algorithms perform their tasks by considering other graph properties and some of these algorithms have also been used for protein complexes identification in PPI networks. For example, a graph clustering algorithm called DPClus [14], can discover and refine graph clusters by keeping track of cluster periphery. This DPClus algorithm is later improved to enhance computational efficiency in another algorithm called IPCA [15]. IPCA finds graph clusters based on a vertex distance and a density measure.

Another example of an algorithm that finds graph clusters based on graph properties other than edge-density is CFinder [16]. CFinder identifies graph clusters based on clique percolation and the graph clusters identified correspond well to known protein complexes. Similar to CFinder, the CMC algorithm also finds graph clusters based on the discovering of cliques [17]. By iteratively assigning weights which indicate the reliability of interactions between proteins, CMC attempts to find cliques that have the highest value of such weights. Other than CMC, another clique-based algorithm called IPC-MCE is proposed in [46]. The algorithm takes each maximal clique identified as the core of a protein complex. It then extends the core by including those "periherral" proteins that are determined to have relatively higher probability to connect to the core.

In [18], an algorithm called COACH is proposed to find protein complexes making use of a different graph property, i.e. core-attachment. In [42], another core-attachment-based algorithm called WPNCA is proposed. Different from COACH, WPNCA utilizes a Pagerank-nibble algorithm to assign a weight to each interaction in a PPI network to obtain a better performance.

Most graph clustering algorithms do not discover overlapping clusters and cannot be used to identify overlapping protein complexes. There are some exceptions, however. For example, an algorithm proposed in [19] can detect overlapping protein complexes based on a generative network model. Another algorithm called ClusterONE [20] can also do so using a measure called *graph cohesiveness*.

As there is some evidence of proteins belonging to the same protein complex performing similar or related functions [43], there are also some attempts to identify protein complexes that can take into consideration information about protein attributes rather than topology only. For example, in [21], PCIFI identifies protein complexes based on finding connected proteins that perform interdependent molecular functions. In [22], another algorithm performs the task of protein complex identification by simultaneously using PPI network data and gene expression data. In [23], an algorithm called GMFTP, is proposed to identify protein complexes based on measures of similarity between attribute values of proteins.

Based on most approaches that are proposed to identify protein complexes in PPI networks, we find that both topological and attribute information is very effective for identifying protein complexes, although there are not too many methods taking into the consideration both of the two types of information. We also find that most algorithms identify protein complexes by finding a number of clusters with some particular properties that are optimized. Hence, sometimes the process of protein complex identification can be seen as an optimization problem. For such an optimization problem, we propose to tackle it with a novel approach based on an evolutionary algorithm called Evolutionary Graph Clustering for Protein Complex Identification (EGCPI). The advantages with the use of an evolutionary algorithm is that it does not have to work under linear constraints like those found in typical numerical optimization problem. It can also discover multiple solutions and can be used to handle big data efficiently and effectively as it can be implemented in parallel.

Given a PPI network, EGCPI constructs an Attributed PPI network Graph (APPIG) by annotating attributes to each protein, based on the GO database which are constructed in Gene Ontology (GO) project [24]. To discover protein complexes, EGCPI assigns a weight to each edge in APPIG, according to the *degree of topological similarity* which quantifies the proportion of common neighboring proteins shared by two interacting proteins. This transforms an APPIG into a weighted APPIG denoted as wAPPIG.

Given a wAPPIG, EGCPI first finds a number of graph clusters in which the proteins are densely connected. It does so by optimizing against an objective function that is defined in terms of the overall *degree of topological similarity* between connecting proteins in each cluster. After the identification of these dense graph clusters, EGCPI then makes use of a *degree of attribute homogeneity* measure to determine how similar the attribute values are between each pair of connecting proteins within each graph cluster. Based on *degree of attribute homogeneity*, a breadth-first search strategy is then used to search for sub-graphs, within each graph cluster, that consist of proteins with similar attribute values. These sub-graphs are, therefore, relatively dense and their vertices share similar attribute values and they correspond well to the characteristics of many protein complexes in real life.

In order to evaluate the performance of EGCPI, we

have tested it with different real data sets. The experimental results show that EGCPI performs well in terms of the number of accurately identified protein complexes that match with known protein complexes. We believe that EGCPI has great potential as an effective protein complex identifier.

2 EGCPI IN DETAILS

Given a PPI network, EGCPI performs the task of protein complex identification in several steps. First, it constructs an Attributed PPI network Graph (APPIG) based on the PPI network and the attribute information that can be obtained from the GO database. Second, a weighted Attributed PPI Graph (wAPPIG) is then constructed by EGCPI. Given an APPIG, EGCPI determines a weight to each edge in the graph based on the *degree of topological similarity*. With all the weights for the edges determined, the APPIG is transformed into a wAPPIG. Given a wAPPIG, an evolutionary algorithm is then used to identify graph clusters within which protein are densely connected by maximizing the overall *degree of topological similarity* in each cluster. Given the graph clusters, a breadth-first search strategy is used to search for subgraphs in each graph cluster based on the the homogeneity of the attribute values associated with the connecting vertices. These subgraphs, whose vertices share similar attribute values and are relatively dense, are found to correspond well with protein complexes in real life.

2.1 Problem statement and notation

A PPI network can be represented as a graph that contains n_V vertices that represent proteins and n_E edges that represent interactions between proteins. As we can obtain information about the attributes of each protein in a PPI network from the GO database, a set of attribute values can be considered as associating with each vertex in a PPI network graph so that this graph becomes an Attributed PPI network Graph (APPIG).

An APPIG can be represented as G = (V, E, Λ), where V represents the set of vertices, E represents the set of edges and Λ represent the set of attribute values for proteins in G. For our application, Λ contains three subsets, Λ_p, Λ_f, Λ_c corresponding to the attribute values of *biological processes*, *molecular functions* and *cellular components*, respectively. *Biological processes* is concerned with the biological objectives a protein is involved in. *Molecular functions* is concerned with the biochemical activities performed by a protein and *cellular components* is concerned with the location where a protein is most active in a cell.

TABLE 1
ATTRIBUTES VALUES OF THE PROTEIN WITH UNIPROT ID Q08683

Biological Processes (Λ_p)	{ GO:0007067, GO:0007049, GO:0016567, GO:0031497, GO:0031145, GO:0051301}
Molecular Functions (Λ_f)	{ GO:0004842}
Cellular Components (Λ_c)	{ GO:0005634, GO:0005680#}

#: Attributes like GO:0005680 is excluded from the experiments since it may bring some bias on inferring it belongs to anaphase promoting complex directly.

In order to explain how the above notations are used, let us assume that we are given a protein with Uniprot ID [25], Q08683, then the attributes in terms of the GO terms found in the GO database are shown in Table 1. It should be noted that not all attribute information available about the protein is used in our experiments with the proposed algorithm because some of them may contain information about protein complex membership. For instance, GO:0005680 is not included in the attribute value set of any proten as it might allow one to directly conclude that the Q08683 belongs to the anaphase promoting protein complex.

Other than avoiding the inclusion of these attribute values, it is noteworthy that the domains of the attributes are allowed to contain different values, i.e., it is possible that, for any vertex in APPIG, $|\Lambda^v_p| \neq |\Lambda^v_f| \neq |\Lambda^v_c|$. In addition, there is no requirement of EGCPI for any of the attributes of any vertex to have any value at all. Furthermore, there is also no requirement for the attribute value sets of two interacting proteins to have the same number of values for *degree of attribute homogeneity* to be determined.

2.2 Construction of wAPPIG

Given APPIG, EGCPI makes use of a *degree of topological similarity* (σ) measure to weight each pair of interacting proteins according to how much they are connected. It is defined as

$$\sigma_{ij} = \frac{\left|e_{i+} \cap e_{j+}\right| + e_{ij}}{\left|e_{i+} \cup e_{j+}\right| - e_{ij}} \quad (1)$$

where e_{i+} is the set of vertices that are connected to v_i, e_{ij} equals to 1 if there is an interaction between v_i and v_j. σ evaluates the extent that a protein pair is connected to each other when considering the network topology.

The magniture of σ ranges from 0 to 1. A higher value means that v_i and v_j are more highly connected. After a weight is determined for each edge in an APPIG, a weighted Attributed PPI network Graph (wAPPIG) is obtained. EGCPI then proceeds to try to find dense graph clusters using wAPPIG.

2.3 Evolutionary graph clustering

EGCPI identifies dense graph clusters using an evolutionary algorithm (EA). EAs have been shown to be very efficient at dealing with problems, such as *NP-Complete* problems that can otherwise be hard to tackle [26]. Recently, EAs have been used for graph clustering [49]. However, they have not been used for protein complex discovery in PPI networks. For EGCPI, an EA is used for the purpose of finding graph clusters that are more densely connected within a cluster than outside. This is because proteins in protein complexes are more desnsely connected in clusters as well.

Given a wAPPIG represented and encoded in a chromosome, the EA that EGCPI makes use of searches for an optimal solution based on a single-criterion objective function defined in terms of the weights in a wAPPIG. Like other EAs, EGCPI evolves an optimal graph clustering arrangement in several steps. To begin, a number of

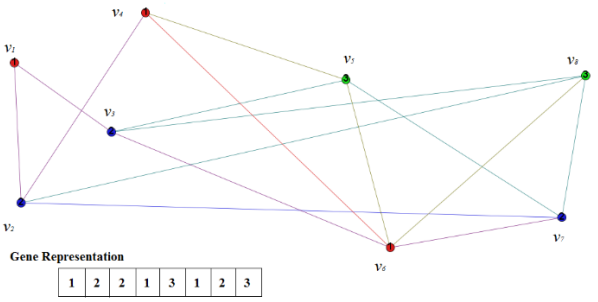

Fig. 1. Straight-forward representation.

chromosomes is first initialized and their fitness values computed based on the weights of the graphs. Based on these fitness values, a particular number of coupled individuals are then selected for reproduction, which consists of both crossover and mutation. Chromosomes with lower fitness values will then be eliminated from the population after each generation of descendants are reproduced. The steps of selection and reproduction are then repeated.

2.3.1 Gene representation

EGCPI uses a *straight-forward representation* [27] to encode a graph clustering arrangement in a chromosome which has a length equal to the total number of vertices, n_V, in the graph. Assuming that S stands for the number of clusters in the wAPPIG and that S can be different in different chromosomes, then we can represent the case of the ith vertex in the wAPPIG being in the jth cluster as the ith gene containing the allele j. An example of the encoding scheme is given in Fig. 1.

2.3.2 Initialization

Since the *straight-forward representation* is used in EGCPI, the initial number of clusters, S must be determined before the initialization. Here S is a randomized parameter decided by the algorithm before each initialization of a chromosome. In the case of EGCPI, S is not to exceed a maximum value of $n_V/2$.

According to the typical approach of initializing a chromosome, a randomized cluster ID is assigned to each vertex. However, total randomness may result in long convergence time. To avoid the problem, EGCPI initializes a population of chromosomes with a different process that consists of the following steps: (i) S nodes is first selected randomly as the initial clusters, (ii) for each vertex v, with e connections (e>0), a cluster ID among S' is assigned to it, where $|S'| \leq S$ and S' is the set of cluster IDs to which vertices connected to v are possibly assigned; (iii) all the vertices without any connection are assigned to one of S clusters randomly. These three steps are iterated p times, where p is the size of the population. With this approach of initialization, p different ways of partitioning the vertices in wAPPIG into different numbers of clusters can be generated.

2.3.3 The process of reproduction

The reproduction process consists of both crossover and mutation. Traditionally, the uniform crossover operator which swaps alleles between two selected parent chromosomes is adopted by many EAs. However, EGCPI adopts a crossover operator modified from the standard uniform crossover. In addition to the popular mutation operator that randomly changes the allele of a gene, EGCPI also makes use of a *Self-Variation* (*SV*) operator to introduce variation to a population of chromosomes.

As part of crossover, instead of allowing all chromosomes to be selected for reproduction, EGCPI allows only a proportion of the chromosomes to be randomly selected and this proportion is set arbitrarily to 30%. This means that the best 30% of the chromosomes in the population can be candidates for reproduction only.

After selection, EGCPI performs crossover by first selecting one parent as a template. The cluster IDs of the other parent then replace all the alleles in the template which share the same gene positions with its cluster member according to the crossover rate. After crossover, each member of the selected cluster can then be selected for mutation based on the mutation rate. With these steps above, EGCPI generates a new descendant.

Although the fitness of the population is incrementally improved as new populations are continually generated, it takes a rather long time for the evolution process to converge. This is especially the case with large data set size. To tackle the problem, EGCPI adds an additonal reproduction operator, called *Self-Variation* (*SV*). The main task of *SV* is to relocate a vertex to a cluster within which vertices share relatively higher weights with it, based on the current graph clustering arrangement. In other words, a vertex should belong to a cluster that shares connections of higher weights with it because connections of higher weights might lead to a higher probability of identifying protein complexes in the cluster. To achieve the expected goal, we use a two-dimension matrix to complete the stage of *SV*: Given all nodes in a network and all clusters, the matrix $VC_{[n_V][S]}$ is defined to represent the weight between a vertex and a cluster. For an element in VC, say vc_{ij}, it equals to zero if there is no connection between node i and other nodes in cluster j. Otherwise, it means some number of connections with a magnitude of aggregated weight bridge node i and other nodes cluster j. Given $VC_{[n_V][S]}$, *Self-Variation* is completed like the follows: For a vertex v_i in cluster j, EGCPI firstly computes its expected weight $E(d_{vic})$ between v_i and each cluster, $E(d_{vic})$ is defined as:

$$E\left(d_{v_i c}\right) = \sum_{1}^{S} \frac{vc_{ij} \times vc_{ij}}{w_{v_i}} \tag{2}$$

where w_{vi} is the total weights of interactions whose one endpoint is v_i. Once $E(d_{vic})$ is obtained, EGCPI determines whether or not v_i should be relocated to a new cluster by computing the largest difference of the real weight and the expected weight between v_i and each cluster, and this difference is defined as:

$$diff_{\max} = \max_{k} \left| vc_{ik} - E\left(d_{v_i c}\right) \right| \tag{3}$$

If $diff_{max} \neq j$, v_i will be moved to cluster $diff_{max}$, otherwise v_i will not be relocated. After the completion of *SV*, those clusters which possess relatively lower weight might be eliminated and the quality of clusters produced by crossover can be improved. Using the *SV* stage, EGCPI can find optimal dense clusters in wAPPIG in a less time.

2.3.4 The fitness function

Every time when EGCPI initializes the population or reproduces a new-birthed individual, the fitness value of each chromosome is computed and the highest fitness value is seen as the population fitness. In order to find partitions in which clusters possess more weight of intra-interactions but less that of inter-connections between clusters, we use a measure called *Independence of Cluster* (*IoC*) to evaluate a cluster that is partitioned by each individual in the population. And this measure is defined as:

$$IoC_{C_i} = \frac{\sum_{v_j, v_k \in C_i} w_{jk}}{\sum_{v_j \in C_i} w_{jk}} \quad (4)$$

where c_i is the ith cluster in a partition, w_{jk} is the weight that is assigned to interaction e_{jk} in wAPPIG, the numerator and the denominator stand for the total weight of intra-interactions and that of all interactions connecting proteins in c_i. Once the *IoC* values of all clusters partitioned by an individual are obtained, we can evaluate the fitness of the individual by using the following objective function:

$$IoC_{wAPPIG} = \sum_{i=1}^{S} \frac{n_{V_i}}{n_V} IC_{C_i} \quad (5)$$

where n_{Vi} and n_V represent the total number of vertices in cluster i and wAPPIG, respectively. IoC_{wAPPIG} evaluates to what extent the independence is if a cluster is compared to another and it ranges from 0 to 1. Apparently, the higher IoC_{wAPPIG} is, the more independent from other ones a cluster is. Therefore, IoC_{wAPPIG} helps to diminish the interdependence between any two clusters.

2.3.5 Summary remarks

Based on what has been illustrated from 2.3.1 to 2.3.4, the evolutionary algorithm is summarized as the pseudo codes shown in Fig. 2. When the phase of evolutionary clustering is finished, EGCPI obtains an individual containing optimal cluster arrangement for each protein in the wAPPIG. These clusters can be represented as $C=\{c_i, 1 \le i \le n_C\}$, where n_C stands for the number of clusters.

Algorithm 1-Evolutionary Clustering

```
Input: Attributed PPI network Graph (APPIG)
Output: A set of clusters C={c_i, 1≤i≤n_C}

generate σ for each interaction according to (1);
construct weighted Attributed PPI network Graph (wAPPIG);

population initialization;
done = false;
while (done == false) {
        for (i = 0; i < maxdescendant; i ++) {
        crossover;
        Self-Variation;
        insertion of individual;
        elimination;
        }
        if(terminal-condition) {
          done = true;
        }
        else {
             if(re-initialization) {
             population initialization
             }
        }
}
return C;
```

Fig. 2. Evolutionary Graph Clustering.

2.4 Identifying protein complexes in found clusters

After using the above EA in wAPPIG, EGCPI obtains a set of clusters from the best individual in a population. In this stage, EGCPI performs a further extraction of subgraphs as protein complexes in each cluster. Rather than detecting protein complexes based on network topology, EGCPI identifies protein complexes by taking into consideration attribute homogeneity between pairwise proteins because proteins within each found cluster are already densely connected. Before searching protein complexes, EGCPI computes the *degree of attribute homogeneity* (θ) between each pair of connected proteins:

$$\theta_{ij} = \frac{\sum_{k=p,c,f} \left| \Lambda_k^{v_i} \cap \Lambda_k^{v_j} \right|}{\sum_{k=p,c,f} \left| \Lambda_k^{v_i} \cup \Lambda_k^{v_j} \right|} \quad (6)$$

θ determines how similar the attribute values are between each pair of connecting proteins within the cluster. It ranges between 0 and 1. A higher θ means there are more functional attributes performed by both of the connected proteins, so that EGCPI tends to form protein complexes by searching such protein pairs in each graph cluster. After EGCPI uses the θ measure to weight all connected vertices, EGCPI uses a breadth-first search (BFS) method to form protein complexes in each cluster. First, it selects an interaction of a vertex with the highest θ, θ_{max}, and incorporates both of the two connected vertices v_i and v_j into a seed set for forming a protein complex; second, based on θ_{max}, EGCPI searches all the neighboring vertices and incorporates those which satisfy the minimum threshold of θ. In EGCPI, this threshold is defined as:

$$r(seed : v_k) = \begin{cases} r \cup v_m \ if\ \theta_{km} \ge \lambda \times \theta_{\max} \\ r \cup \Phi \ otherwise \end{cases} \quad (7)$$

where v_k stands for a vertex in the seed set and v_m is a vertex connecting to v_k. In other words, only vertices sharing connections with θ which is higher than $\lambda \times \theta_{max}$ can be incorporated into the seed set. The searching in the second step will be terminated till there is no new vertex added into the seed set. When the above search in a cluster is finished, EGCPI forms a protein complex using the proteins in the seed set. EGCPI will stop forming protein complexes till it traverses all vertices in the cluster. As using the above search strategy may produce some protein complexes whose sizes are small, EGCPI discards those identified protein complexes including fewer than 3 proteins.

To reduce the redundancy of proteins in the identified protein complexes, EGCPI computes overlapping score between an identified protein complex and protein complexes in the identified set. The overlapping score is de-

Algorithm 2: Protein complex identification

```
Input: A set of clusters C
Output: A set of protein complexes PC

for each cluster c_i {
    generate θ for each interaction in c_i;
    for each vertex v_i {
        find θ_max;
        create a new protein complex r;
        create a new link list P_visiting;
        P_visiting = P_visiting ∪ v_i;
        P_visiting = P_visiting ∪ v_j;
        while(|P_visiting|>0){
            v_k = head of P_visiting;
            P_visiting- v_k;
            r = r ∪ v_k;
            search v_m: neighbors of v_k;
            if(θ_km≥λ×θ_max){
              P_visiting = P_visiting ∪ v_m;
            }
         }
         if(Ov_r≤OvMax){
           PC = PC ∪ r;
         }
      }
}
return PC;
```

Fig. 3. Protein complex identification.

fined as:

$$Ov_r = \max \frac{|r \cap PC_I|}{|r \cup PC_I|} \quad (8)$$

where r and PC_I stand for an identified protein complex after once of the search and any other protein complex that is in the identified set, respectively. Then EGCPI uses a threshold *OvMax* to exclude those identified protein complexes whose overlapping scores are higher than the threshold. In order to explain this BFS method in detail, we give the pseudo codes in Fig. 3.

3 Experimental Results

For performance testing, EGCPI has been tested with five sets of real PPI network data. They include: (i) Collins [28], (ii) Gavin [2], (iii) Krogan-Core [29], (iv) DIP-Scere [30] and (v) DIP-Hsapi [30]. Data sets (i), (ii) and (iii), which can be collected from the BioGRID database [31], are concerned with *yeast Saccharomyces cerevisiae*. In our experiments, the data we used are collected from version 3.2.118 of the BioGRID database. Compared with Collins, Gavin and Krogan-Core, DIP-Scere, which is also related to *Saccharomyces cerevisiae*, has a much larger data set size. Unlike the other data sets, DIP-Hsapi, is collected from human beings. Both the two DIP data sets are collected from the 2013 version of the DIP database [30]. The properties of these five data sets are shown in Table 2.

TABLE 2
STATISTICS ON THE USED DATA SETS OF PPI NETWORKS

Data Set	n_V	n_E	n_Λ
Collins	1620	9064	2042
Gavin	1430	6531	2107
Krogan-Core	2674	7075	3064
DIP-Scere	4579	20845	4237
DIP-Hsapi	2434	3053	7031

n_V, the number of proteins; n_E, the number of interactions; n_Λ, the number of attributes.

For all the five data sets, the protein attribute information required for the construction of the APPIG were obtained from the January, 2016 version of the GO database [32]. As mentioned above, the GO terms of the *cellular components* which may provide information about the protein complexes that a protein belongs to are not included in the experimental data sets.

To evaluate the performance of different protein complex identification algorithms, we compared the protein complexes idneitified with known protein complexes in *Saccharomyces cerevisiae* as contained in the January, 2016 version of the CYC2008 [33] and MIP/CYGD [34] [35] databases. There are 408 and 255 known protein complexes for *Saccharomyces cerevisiae* in the CYC2008 and MIP/CYGD databases respectively. Following the work done in [20], we used the known protein complexes in these databases for performance evaluation. After removing protein complexes that are made up of fewer than 3 proteins, we have obtained a total of 296 distinct protein complexes in the two databases for performance evaluation.

For the evaluation of protein complexes in the data set, DIP-Hsapi, we compared the protein complexes identified by different computational approaches with the known ones as contained in the MIPS/CORUM [36] database and there are altogether 1466 known protein complexes that are made up of three or more proteins in MIPS/CORUM.

3.1 Setting up of experiments and performance evaluation

For performance evaluation, EGCPI were compared with different algorithms, including GMFTP, MCL, DPClus, IPCA, CFinder, COACH, SPICi and ClusterONE. We used these 9 algorithms to identify protein complexes in the five data sets described above. Algorithms like MCL, DPClus, IPCA, CFinder COACH, SPICi and ClusterONE identify protein complexes in PPI networks based only on network topologies. Attribute information that are made available are not considered by these algorithms. As algo-

TABLE 3
PARAMETER SETTINGS OF DIFFERENT ALGORITHMS

Approach	Parameter	Approach	Parameter
ClusterONE	s=3, density=auto (default setting)	GMFTP	K=1000 (default setting)
MCL	inflation = 1.8 (default setting)	SPICi	minimum cluster size = 3
DPClus	CP_{in}=0.5, d_{in}=0.6 (default setting)	IPCA	S=3, P=2, T_{in}=0.4/0.9
CFinder	k=3	COACH	W=0.225 (default setting)
EGCPI	λ=0.7/0.8, *OvMax*=0.7/0.8/0.9		

rithms like MCL, SPICi and ClusterONE can be used with weighted PPI network data, we also used them to identify protein complexes in the weighted PPI network data that are used by EGCPI. As GMFTP considers both network topology and functional attributes when identifying protein complexes, it is provided with exactly the same attribute information as we provided for EGCPI. For the above algorithms to perform their tasks, the settings of the parameters for each of them are given in Table 3.

When using EGCPI to find graph clusters, the population size is set to 100, and the crossover rate is set to 0.6 for evolutionary clustering. We ran the EA for 30 generations, before it was required to return the best partition of graph clusters. The reason why the maximum number of generations is set to 30 is because we found that the EA used in EGCPI could usually achieve the best results within around 30 generations. For λ, it was set to 0.7 or 0.8. As for *OvMax*, it was set to 0.7, 0.8 or 0.9 to obtain a better performance.

For other algorithms, their parameters were set following the recommendations of the authors, or modified as many times as possible to obtain a better performance. For example, different settings of T_{in} in IPCA have been proposed. In [37], it was set to 0.9 and in [38], it was set to 0.4 to obtain best results. For our experiments, we therefore used both these two settings to obtain a better performance. Another example is the parameter setting of CFinder. As there are no recommended settings for the

TABLE 4
RESULTS OF PRECISION, RECALL, F-MEASURE AND MMR

Data Set	Approach	#	Coverage	*f-measure*			*MMR*
				Precision	*Recall*	*f-measure*	
Collins	EGCPI	236	1160	0.67	0.54	0.6[1st]	0.29[2nd]
	GMFTP	203	1160	0.59	0.47	0.52[3rd]	0.28[3rd]
	ClusterONE	203	1293	0.54	0.45	0.49	0.25
	DPClus	203	1185	0.53	0.44	0.48	0.26
	MCL	282	1620	0.4	0.49	0.44	0.26
	SPICi	120	973	0.68	0.34	0.45	0.2
	IPCA	499	1160	0.48	0.66	0.55[2nd]	0.33[1st]
	COACH	245	1114	0.51	0.48	0.49	0.27
	CFinder	114	1160	0.7	0.32	0.44	0.2
Gavin	EGCPI	298	1150	0.66	0.56	0.6[1st]	0.27[1st]
	GMFTP	172	917	0.64	0.42	0.5[2nd]	0.22[3rd]
	ClusterONE	243	1268	0.4	0.37	0.38	0.19
	DPClus	217	1107	0.41	0.36	0.38	0.19
	MCL	177	1430	0.39	0.27	0.33	0.14
	SPICi	126	907	0.54	0.27	0.36	0.15
	IPCA	695	1124	0.36	0.61	0.46[3rd]	0.25[2nd]
	COACH	324	1052	0.43	0.46	0.44	0.22[3rd]
	CFinder	98	1124	0.56	0.2	0.29	0.11
Krogan-Core	EGCPI	526	1442	0.53	0.65	0.59[1st]	0.32[1st]
	GMFTP	299	1411	0.41	0.49	0.44	0.28[3rd]
	ClusterONE	242	1071	0.48	0.42	0.45[3rd]	0.23
	DPClus	497	1758	0.25	0.5	0.33	0.26
	MCL	514	2674	0.2	0.42	0.27	0.22
	SPICi	233	1239	0.38	0.36	0.37	0.19
	IPCA	701	1140	0.41	0.67	0.51[2nd]	0.3[2nd]
	COACH	349	1056	0.49	0.53	0.51[2nd]	0.27
	CFinder	115	1140	0.49	0.21	0.3	0.14
DIP-Hsapi	EGCPI	489	1241	0.37	0.13	0.2[1st]	0.05[1st]
	GMFTP	187	806	0.31	0.04	0.07	0.02
	ClusterONE	201	710	0.29	0.04	0.08	0.02
	DPClus	563	1644	0.19	0.09	0.12	0.05[1st]
	MCL	549	2434	0.17	0.07	0.1	0.04[2nd]
	SPICi	194	863	0.35	0.05	0.09	0.02
	IPCA	289	515	0.56	0.12	0.19[2nd]	0.05[1st]
	COACH	151	492	0.63	0.07	0.13[3rd]	0.03[3rd]
	CFinder	111	515	0.5	0.04	0.08	0.02
DIP-Scere	EGCPI	441	2787	0.48	0.64	0.55[1st]	0.32[2nd]
	GMFTP	517	2509	0.28	0.60	0.38[3rd]	0.3[3rd]
	ClusterONE	335	1368	0.35	0.42	0.38[3rd]	0.19
	DPClus	856	2973	0.15	0.54	0.24	0.26
	MCL	691	4579	0.12	0.32	0.18	0.18
	SPICi	394	2055	0.24	0.39	0.3	0.18
	IPCA	1602	2142	0.23	0.8	0.36	0.37[1st]
	COACH	853	1952	0.27	0.7	0.39[2nd]	0.32[2nd]
	CFinder	192	2143	0.3	0.2	0.24	0.13

#: The number of protein complexes identified. Coverage: The number of distinct proteins in the identified protein complexes.

size of clique, we tried different values of k, from 3 to 50, and found that CFinder performed better when k was set to 3.

For the purpose of performance evaluation, we used two measures. One is the *f-measure* which can be taken as a measure that determines the overall accuracy of the identified protein complex. The *f-measure* can be defined as follows:

$$f-measure = 2\times\frac{precision\times recall}{precision+recall}$$
$$precision=\frac{TP}{TP+FP},\quad recall=\frac{TP}{TP+FN} \tag{9}$$

The *f-measure* is determined by the value of *precision* and *recall*. *TP* is the number of identified protein complexes whose matching rates are equal to or larger than a particular threshold O'. In our experiments, we set the threshold to 0.2, which is recommended in [48]. *FP* is the number of identified protein complexes whose matching rates are less than the threshold O'. *FN* is the number of known protein complexes that are not matched by any identified protein complex.

The other measure we used to determine the quality of identified protein complexes is the *Maximum Matching Rate* (*MMR*) [20]. Unlike the *f-measure*, it needs a predefined threshold to evaluate the identified protein complexes, *MMR* offers a natural way to measure how accurately the identified protein complexes can represent a benchmarking set. Based on the features of *f-measure* and *MMR*, these two evaluation criteria are complementary to each other.

3.2 Performance analysis

The experimental results of *f-measure* and *MMR* obtained by different algorithms have been summarized in Table 4. As the table shows, EGCPI obtains best *f-measure* in all the five data sets. Although EGCPI doesn't always obtain best performance on *Precision* or *Recall*, but it makes a better compromise between the two measures so that the results of *f-measure* obtained by EGCPI are better than those done by other approaches. Given the results of *f-measure*, it is said the overall accuracy of protein complexes identified by EGCPI is better than prevalent algorithms. When evaluated by the *MMR* measure, EGCPI also performed robustly in all the data sets. As the table shows, in the data sets of Gavin, Krogan-Core and DIP-Hsapi, EGCPI ranks the first and it holds the second best place in Collins and DIP-Scere. Given such results, it is concluded that the protein complexes identified by EGCPI may more accurately represent the known protein complexes in the benchmarking sets. As for the number of identified protein complexes and the coverage, EGCPI covered relatively more proteins when detecting protein complexes in each set of data, but it did not identify a large number of protein complexes. Together with the results of *f-measure* and *MMR*, it is seen that EGCPI is not an algorithm that obtains better experimental results by increasing the number of identified protein complexes.

To investigate whether other algorithms can obtain a competitive performance if they are used with the same weighted PPI network data, we compared the experimental performance obtained by those algorithms which can deal with weighted network data, including ClusterONE, MCL and SPICi with that obtained by EGCPI. The results are summarized in Table 5. As the table shows, the performance of ClusterONE, MCL and SPICi is improved to some extent, but EGCPI still outperforms these three algorithms even when they use the weighted network data generated by *degree of topological similarity*. Obtaining such results shows taking into consideration both topology and attribute makes EGCPI outperform those algorithms considering network topology only.

TABLE 5
EXPERIMENTAL RESULTS USING WEIGHTED NETWORK DATA

Data Set	Approach	#	Coverage	*f-measure*			*MMR*
				Precision	*Recall*	*f-measure*	
Collins	EGCPI	236	1160	0.67	0.54	0.6	0.29
	ClusterONE	181	1207	0.59	0.44	0.5	0.26
	MCL	302	1620	0.4	0.52	0.46	0.28
	SPICi	94	819	0.79	0.31	0.44	0.18
Gavin	EGCPI	298	1150	0.66	0.56	0.6	0.27
	ClusterONE	156	950	0.6	0.37	0.46	0.2
	MCL	202	1430	0.44	0.36	0.39	0.17
	SPICi	83	532	0.75	0.25	0.37	0.13
Krogan-Core	EGCPI	526	1442	0.53	0.65	0.59	0.32
	ClusterONE	222	1007	0.5	0.42	0.46	0.25
	MCL	581	2674	0.2	0.48	0.29	0.24
	SPICi	65	435	0.86	0.22	0.35	0.13
DIP-Hsapi	EGCPI	489	1241	0.37	0.13	0.2	0.05
	ClusterONE	229	775	0.31	0.05	0.09	0.02
	MCL	644	2434	0.17	0.08	0.11	0.05
	SPICi	37	148	0.73	0.02	0.04	0.01
DIP-Scere	EGCPI	441	2787	0.48	0.64	0.55	0.32
	ClusterONE	243	919	0.52	0.44	0.48	0.21
	MCL	903	4579	0.13	0.46	0.2	0.23
	SPICi	44	235	0.7	0.11	0.19	0.07

In the experiement shown in this table, ClusterONE, MCL and SPICi identified protein complexes using weighted PPI network graphs that are generated based on the *degree of topological similairty* measure.

In total, EGCPI's performance on the task of protein complex identification is very promising. It obtains better results in both *MMR* and *f-measure* in most data sets. Therefore, EGCPI can perform better when it treats the task of protein complex identification as an optimization problem which takes into consideration both attribute information and topology of a PPI network.

3.3 The effects of parameter settings

As described before, there are two parameters in EGCPI, λ and *OvMax* determining the results of identified protein complexes. In order to investigate how these parameters, impact the results of protein complex identification, we executed EGCPI to identify protein complexes in the five data sets with λ and *OvMax* changing from 0.1 to 1.0, using a 0.1 increment. After collecting the identified proteincomplexes using different combinations of λ and *OvMax*, we evaluate them with *Precision*, *Recall*, *f-measure* and *MMR*. Here we take the variations of the above measures obtained in the data set Krogan-Core as an example (Fig. 4). As the surfs shown in Fig. 4 (a) and Fig. 4 (c), *Precision* and *f-measure* share an analogous trend when λ and *OvMax* change. Simply setting λ and *OvMax* to the values near to 0 or 1 may not obtain satisfying results. For example, EGCPI may obtain a relatively low *Precision* when λ is set to 0.2, no matter how to configure *OvMax*. When using a small λ, EGCPI may incorporate more proteins with lower *degree of attribute homogeneity* so that the protein complexes may not well match the known protein complexes. Although *Precision* and *f-measure* are relatively higher when λ and *OvMax* are set very near to 1, EGCPI cannot identify those protein complexes including more proteins so that some biological significance of the identified protein complexes is missing. Given such concerns, appropriate settings of λ and *OvMax* are essential to the experimental performance of EGCPI. As Fig. 4 (b) and Fig. 4 (d) show, *Recall* and *MMR* share the similar variations under different combinations of λ and *OvMax*. Using higher λ and *OvMax*, EGCPI may identify more protein complexes in the PPI network (Fig. 5) so that it is possible for EGCPI to identify more protein complexes in the benchmarking set and higher *Recall* can be obtained. Since each identified protein complex including fewer proteins as λ and *OvMax* are set as higher values, its *MMR* is consequently larger than that of the identified protein complex including more proteins. Since we desire an approach that can accurately identify protein complexes including relatively more proteins, in general we recommend to let EGCPI perform the task of protein complex discovery when λ and *OvMax* are set between 0.6 and 0.9. EGCPI may obtain a robust performance when λ and *OvMax* are appropriately configured in that range. This is also the reason why we used the parameter settings for EGCPI which are shown in Table 3 in our experiments.

3.4 Complexity of EGCPI

To determine the efficiency of EGCPI, we analyze the complexity of EGCPI and recorded the execution time when it performed the task of protein complex identification. Here we mainly focus on the complexity of the evo-

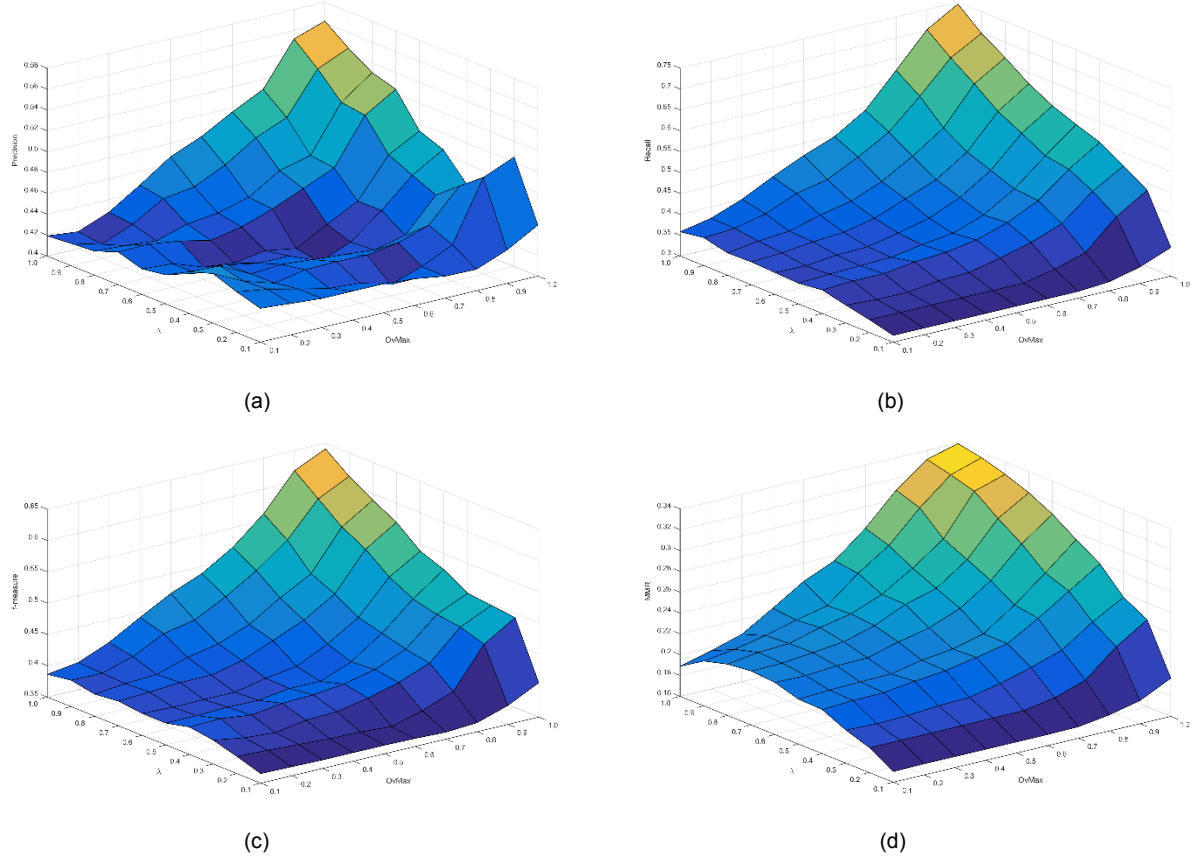

Fig. 4. The evaluation of protein complexes identified by EGCPI using different settings of λ and *OvMax*

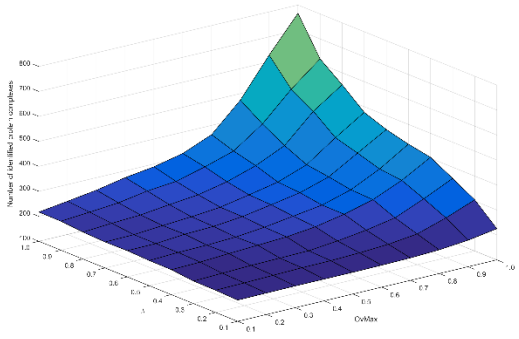

Fig. 5. The number of protein complexes identified by EGCPI using different settings of λ and *OvMax*

lutionary clustering as it is the dominant part of EGCPI. Unlike other algorithms, the complexity of EGCPI can be considered separately for initialization and reproduction. When EGCPI initializes population of p individuals for a PPI network containing n_V vertices and n_E edges, for each chromosome which can be randomly initialized to contain S clusters, it performs its tasks requiring $O(2p(n_E+S))$, to construct the chromosomes and compute their fitness values. For reproduction, if d descendants are produced for each generation, and if the rate of crossover is r, EGCPI works under the complexity of $O(n_V r)$ and it takes $O(2n_E+n_V(4+r+S)+2S)$, for mutation and the computation of fitness. For the whole reproduction process, therefore, the complexity is $O(d(2n_E+n_V(4+r+S)+2S))$. If EGCPI takes g generations to achieve convergence, the complexity is of $O(2p(n_E+S)+dg(2n_E+n_V(4+r+S)+2S))$. Since $2p$ and dg are much smaller than n_E and n_V, we can assume that the two are equal to a constant, c. So, the complexity of EGCPI is approximately of $O(c(3n_E+Sn_V))$. But as reproduction progresses, the complexity should be much lower than this estimation since the operations for SV (i.e., $O(n_E+n_V(S+2))$) decrease tremendously as the evolutionary clustering goes on. In our experiment, we ran EGCPI on a workstation with 4 CPU (3.5GHz) and 16GB RAM. The total time consumption for evolutionary graph clustering in the largest dataset DIP-Scere was less than 3.5 seconds and that EGCPI reproduced individuals of each generation costed less than 0.15 second, when we used the settings of EGCPI mentioned in Section 3.1. Given the complexity analysis and time-consumption recorded in the experiment, EGCPI can be seen as an efficient algorithm for protein complex identification.

3.5 Biological significance of the identified protein complexes

Besides evaluating EGCPI by *f-measure* and *MMR*, we also investigated whether there was something biologically significant in the identified protein complexes.

To perform the investigation, we used GO::TermFinder [39] to make a functional enrichment analysis. Provided by SGD [40], GO::TermFinder is a web-based service that can be used for searching significant shared GO terms in the proteins of an identified protein complex. In our analysis, we set different thresholds of *p-value* from 1E-2 to 1E-15. In other words, those GO terms whose *p-values* are equal to or lower than the threshold may be identified as significant GO ones. Not all these protein complexes whose proteins share significant GO terms are known ones that can be found in databases such as MIPS/CYGD and CYC2008, but they can be considered as candidates of real protein complexes due to their statistical significance revealed by the functional enrichment analysis. Having obtained the *p-value* of each protein complex, we recorded

TABLE 6
P-VALUE TEST ON PROTEIN COMPLEXES IDENTIFIED BY DIFFERENT ALGORITHMS

Data Set	Approach	<1E-15	<1E-10	<1E-5	<1E-2
Collins	EGCPI	36.86%	60.59%	87.29%	94.91%
	GMFTP	31.53%	59.6%	85.71%	94.09%
	MCL	15.6%	28.72%	70.21%	83.69%
	IPCA	30.7%	50.3%	78.4%	90.2%
	ClusterONE	25.6%	48.8%	78.8%	93.1%
Gavin	EGCPI	42.62%	63.09%	86.91%	95.64%
	GMFTP	31.4%	51.74%	82.56%	92.44%
	MCL	22.59%	33.33%	64.97%	80.79%
	IPCA	14.9%	37.7%	72.9%	92.8%
	ClusterONE	20.9%	31.7%	60.5%	90.1%
Krogan-Core	EGCPI	32.13%	46.19%	74.52%	89.54%
	GMFTP	18.39%	35.45%	63.88%	69.23%
	MCL	8.37%	14.79%	43.17%	70.82%
	IPCA	14.6%	28.4%	67.6%	88.7%
	ClusterONE	21.9%	38%	68.2%	88.4%
DIP-Scere	EGCPI	33.56%	52.15%	82.54%	94.56%
	GMFTP	14.2%	26.26%	54.28%	78.99%
	MCL	8.17%	11.43%	32.56%	62.95%
	IPCA	3.9%	15.2%	58.1%	91.5%
	ClusterONE	15.5%	26.9%	60.6%	85.1%
DIP-Hsapi	EGCPI	22.49%	50.1%	93.66%	99.79%
	GMFTP	16.04%	39.04%	84.49%	97.33%
	MCL	9.29%	25.87%	71.22%	87.07%
	IPCA	1.4%	18%	92%	100%
	ClusterONE	9.5%	33.8%	84.6%	99%

TABLE 7
TEN MATCHED PROTEIN COMPLEXES IDENTIFIED BY EGCPI

Protein complex	*mr*	Subunits (Uniprot ID)	Data Set
Cytoplasmic exosome complex	0.9	**P46948**,**P25359**,**P53859**,**P53256**,**Q12277**,**P38792**,**Q05636**,Q08162,**Q08285**,**P48240**	Collins
RSC complex	0.94	**P38781**,**Q12406**,**Q02206**,**Q06168**,**P25632**,**P53330**,P38210,**Q9URQ5**,**P53236**,**Q06639**,**P32832**,**Q05123**,**Q03124**,**P32597**,**Q07979**,**Q06488**,**P43609**	Collins
Mitochondrial ribosomal complex (small subunit)	0.81	**P53733**,P10663,**P47150**,**P21771**,**P38175**,**P10662**,**Q01163**,P02381,**P53305**,**Q02608**,**P36056**,**P12686**,**P32902**,**Q02950**,**P17558**,**Q03201**,**P27929**,**P47141**,P38796,**P28778**,P19955,**P40496**,**P33759**,**Q45TY3**,O75012,**Q03799**,**Q03246**,**Q03976**,**P38120**,**P53292**,**P42847**	Gavin
TFIID complex	0.86	**Q03750**,P11747,P35189,**P46677**,**P38129**,**Q12030**,**P50105**,**Q12297**,**P23255**,**Q03761**,**P53040**,**Q05027**,**Q05021**,**Q04226**	Gavin
mRNA cleavage and polyadenylation specificity factor complex	0.93	**P36104**,**P35728**,**P39927**,**P29468**,**Q06224**,**Q06632**,**Q12102**,P32598,**Q01329**,**P45976**,**Q08553**,**P42073**,**P42841**,**Q06102**,**P53538**	Krogan-Core
DASH complex	1.0	**P69852**,**P69851**,**P69850**,**P36131**,**P53168**,**Q12248**,**P36162**,**Q03954**,**P35734**,**P53267**	Krogan-Core
Anaphase-promoting complex	0.8	**P14724**,**Q12440**,**Q04601**,**P09798**,**Q08683**,**P53068**,**Q12157**,**P40577**,Q12379,**P38042**,**P53886**,**Q12107**,**P16522**,P26309,P53197	DIP-Scere
Nuclear exosome complex	1.0	**P46948**,**P25359**,**P53859**,**P38801**,**P53256**,**Q12277**,**P38792**,**Q05636**,**Q08162**,**Q08285**,**Q12149**,**P48240**	DIP-Scere
PBAF Complex	0.634	O96019,**Q68CP9**,**Q86U86**,**P51532**,**Q12824**,Q92922,**Q8TAQ2**,**Q969G3**,**Q96GM5**,Q92925,Q6STE5	DIP-Hsapi
Arp2/3 Complex	0.875	**P61160**,**P61158**,**O15143**,**O15144**,**O15145**,**P59998**,**O15511**	DIP-Hsapi

mr, matching rate between the identified protein complex and the known protein complex; Uniprot IDs of matched proteins are in bold font.

the ratio that the identified protein complexes containing at least one GO term with *p-value* lower than different thresholds in each GO category.

Besides analyzing the protein complexes identified by EGCPI, we also performed the same *p-value* test on the protein complexes identified by GMFTP, MCL, IPCA and ClusterONE. GMFTP has been proved to be a very effective method which takes into consideration both network topology and functional attributes. MCL, IPCA and ClusterONE are also proved to be effective methods considering network topology to identify protein complexes in the PPI network. Selecting the above approaches to compare with EGCPI is because all of them obtained robust performances in the five sets of data. Those approaches which did not perform robustly were not considered in the *p-value* test. The results of *p-value* test of EGCPI, GMFTP, MCL, IPCA and ClusterONE are presented in Table 6. As the table shows, the proportion of protein complexes with significant GO terms identified by EGCPI is higher than that of other algorithms, especially when the threshold of *p-value* is tightened (e.g., *p-value*<1E-15). This means EGCPI can identify more protein complexes with shared significant GO terms, compared with other approaches. Despite some of those identified protein complexes are not the known protein complexes currently, they have a higher possibility to be identified as real protein complexes through laboratory experiments in future. Based on the results of *p-value* test, it is seen that EGCPI is a promising approach to protein complex discovery.

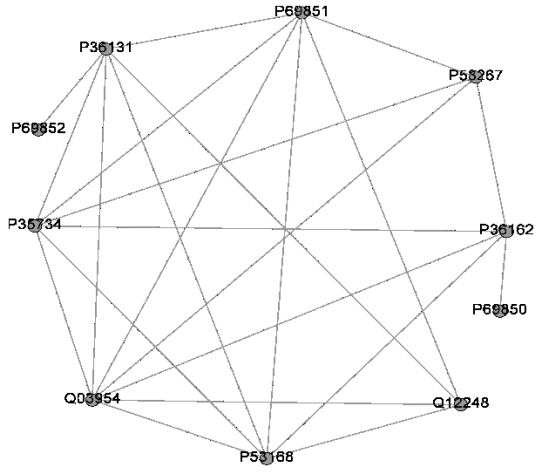

Fig. 6. The structure of DASH complex in CYC2008 database. The structure identified by EGCPI completely matches that of the DASH complex.

Moreover, we also enumerate a number of matched protein complexes identified by EGCPI and select several protein complexes to make an analysis on both topology and GO information.

In Table 7, we enumerate 10 matched protein complexes identified by EGCPI. As the table shows, known protein complexes including more proteins like RSC complex, Mitochondrial ribosomal complex (small unit), Anaphase-promoting complex and PBAF complex can be successfully identified by EGCPI. Meanwhile, protein complexes including fewer sub-units such as Cytoplasmic exosome complex and Arp2/3 Complex can be detected by the proposed approach, too. Given such results, it is seen that EGCPI is effective for identifying protein complexes with different sizes.

In data set Krogan-Core, DASH complex was identified successfully by EGCPI. The structure stored in CYC2008 database is shown in Fig. 6. It is noticed that proteins except of P69850 and P69852 connect to each other densely. Due to this topological feature, P69850 and P69852 might be excluded from the protein complex by some algorithms based on network topology. For examples, CFinder, which is based on clique percolation, also identified DASH complex successfully, but P69850 and

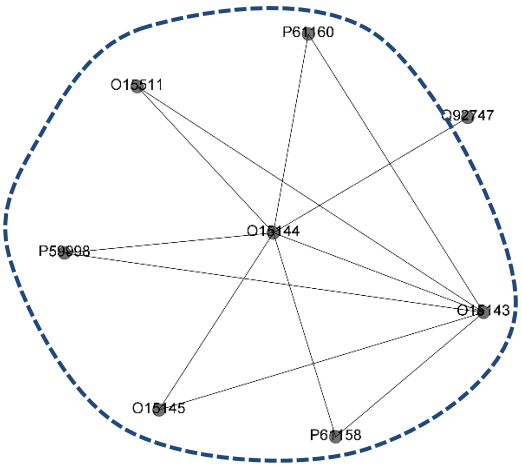

Fig. 7. The structure of Arp2/3 Protein complex in MIPS/CORUM database. The matched proteins identified by EGCPI are in the dashed circle.

P69852 were excluded from the complex because of their lower connectivity, compared with other proteins in the complex. MCL identified all the proteins of DASH complex, but the identified structure involved a superfluous protein, Q12374 into the complex. Compared with other algorithms, EGCPI can identify complexes successfully may be due to the following reasons. First, it utilizes an evolutionary clustering approach to locate proteins sharing with higher *degree of topological similarity* in a cluster. Then, the BFS method is used to form the protein complex not only based on topology, but also the *degree of attribute homogeneity* between two connected proteins. For example, there are 21 GO terms and 16 GO terms annotated to P36131 and P69852 as attributes values, respectively. 16 GO terms are shared by the two proteins so that *degree of attribute homogeneity* between them is 0.76, which is relatively high. Given this reason, P69852 is treated as a member of the identified protein complex by EGCPI. Taking into consideration attribute homogeneity when identifying protein complex makes EGCPI find more proteins with homogeneous attributes and improve its accuracy of protein complex identification.

In DIP-Hsapi, EGCPI successfully identified Arp2/3 Complex. The structure of Arp2/3 complex is shown in Fig. 7. As the figure shows, Arp2/3 complex is a typical star structure that O15144 connects all the other proteins in the protein complex. The identified protein complex successfully matched all the proteins in the known complex, but Q92747 was incorrectly incorporated into Arp2/3 complex. By investigating the GO terms annotated to Q92747 and O15144, we found that more than 60 percent of the attributes of Q92747 were also associated to O15144. As a result, EGCPI treated Q92747 as a member of the identified protein complex because Q92747 performs similar functions to those of O15144. Given the "incomplete" status of Arp2/3 complex and the similar functional attributes performed by Q92747 and O15144, there is a high possibility that Q92747 is confirmed as a member of Arp2/3 complex in future through laboratory-based experiments.

4 Conclusion

In this paper, a novel approach to protein complex identification, EGCPI is proposed. EGCPI constructs a weighted PPI network Graph by assigning interactions with weights according to the *degree of topological similarity* measure. Based on the evolutionary strategy that can form the optimal clusters with vertices that are densely connected in the PPI network graph, a breadth-first search method is then used to further partition each cluster and discover protein complexes with proteins sharing high *degree of attribute homogeneity*. The experimental performance proves that EGCPI using evolutionary graph clustering can obtain better results when identifying known protein complexes. In future, we will attempt to improve the efficiency of EGCPI, develop some evolutionary approaches which can discover overlapping protein complexes in the PPI network.

References

[1] V. Spirin and L. A. Mirny, "Protein Complexes and Functional Modules in Molecular Network," *Proc. Nat'l Academy of Sciences*, USA, vol. 100, no. 21, pp. 12123-12128, 2003.

[2] A. Gavin, P. Aloy, P. Grandi, R. Krause, M. Boesche, M. Marzioch, C. Rau, L.J. Jensen, S. Bastuck, B. Dumpelfeld, A. Edelmann, M. Heurtier, V. Hoffman, C. Hoefert, K. Klein, M. Hudak, A. Michon, M. Schelder, M. Schirle, M. Remor, T. Rudi, S. Hooper, A. Bauer, T. Bouwmeester, G. Casari, G. Drewes, G. Neubauer, J.M. Rick, B. Kuster, P. Bork, R.B. Russell, and G. Superti-Furga, "Proteome survey reveals modularity of the yeast of the yeast cell machinery," *Nature*, vol. 440, pp. 631-636, 2006.

[3] A. Gavin, M. Bosche, R. Krause, P. Grandi, M. Marzioch, A. Bauer, J. Schultz, J.M. Rick, A. Michon, C. Cruciat, M. Remor, C. Hofert, M. Schelder, M. Brajenovic, H. Ruffner, A. Merino, K. Klein, M. Hudak, D. Dickson, T. Rudi, V. Gnau, A. Bauch, S. Bastuck, B. Huhse, C. Leutwein, M. Heurtier, R.R. Copley, A. Edelmann, E. Querfurth, V. Rybin, G. Drewes, M. Raida, T. Bouwmeester, P. Bork, B. Seraphin, B. Kuster, G. Newbauer, and G. Superti-Furga, "Functional Organization of the Yeast Proteome by Systematic Analysis of Protein Complexes," *Nature*, vol. 415, pp. 141-147, 2002.

[4] Y. Ho, A. Gruhler, A. Heilbut, G.D. Bader, L. Moore, S. Adams, A. Millar, P. Taylor, K. Bennett, K. Boutilier, L. Yang, C. Wolting, I. Donaldson, S. Schandorff, J. Shewnarane, M. Vo, J. Taggart, M. Goudreault, B. Muskat, C. Alfarano, D. Dewar, Z. Lin, K. Michalickova, A.R. Willems, H. Sassi, P.A. Nielsen, K.J. Rasmussen, J.R. Andersen, L.E. Johansen, L.H. Hansen, H. Jespersen, A. Podtelejnikov, E. Nielsen, J. Crawford, V. Poulsen, B.D. Sorensen, J. Matthiesen, R.C. Hendrickson, F. Gleeson, T. Pawson, M.F. Moran, D. Durocher, M. Mann, C.W.V. Hogue, D. Figeys, and M. Tyers, "Systematic Identification of Protein Complexes in Saccharomyces cerevisiae by Mass Spectrometry," *Nature*, vol. 415, pp. 180-183, 2002.

[5] T. Deisboeck and J. Y. Kresh, *Complex systems Science in BioMedicine*, Springer, 2006.

[6] X. Li, M. Wu, C.K. Kwoh, and S.K. Ng, "Computational approaches for detecting protein complexes from protein interaction networks: a survey," *BMC Genomics*, vol. 11(1) article 1, 2010.

[7] J. Ji, A. Zhang, C. Liu, and X. Quan, "Survey: Functional module detection from protein-protein interaction networks," *IEEE Trans. Knowledge and Data Engineering*, vol. 26, no. 2, pp. 261-277, 2014.

[8] A.H.Y. Tong, B. Drees, G. Nardelli, G.D. Bader, B. Brannetti, L. Castagnoli, M. Evangelista, S. Ferracuti, B. Nelson, S. Paoluzi, M. Quondam, A.

Zucconi, C.W.V. Hogue, S. Fields, C. Boone, and G. Cesareni, "A combined Experimental and Computational Strategy to Define Protein Interaction Networks for Peptide Recognition Modules," *Science*, vol. 295, no. 5553, pp. 321-324, 2002.

[9] G. Bader and C. Hogue, "An Automated Method for Finding Molecular Complexes in Large Protein Interaction Networks," *BMC Bioinformatics*, vol. 4, article 2, 2003.

[10] S.v. Dongen, "Graph Clustering by Flow Simulation," PhD thesis, Univ. of Utrecht, The Netherlands, 2000.

[11] S.v. Dongen, "A Cluster Algorithm for Graphs," Technical Report, R 0010, CWI, 2000.

[12] A.D. King, N. Przulj, and I. Jurisica, "Protein Complex Prediction via Cost-Based Clustering," *Bioinformatics*, vol. 20, no. 17, pp. 3013-3020, 2004.

[13] X. Ding, W. Wang, X. Peng, and J. Wang, "Mining Protein Complexes from PPI Networks Using the Minimum Vertex Cut," *Tsinghua Science and Technology*, vol. 17, no. 6, pp. 674-681, 2012.

[14] M. Altaf-Ul-Amin, Y. Shinbo, K. Mihara, K. Kurokawa, and S. Kanaya, "Development and Implementation of an Algorithm for Detection of Protein Complexes in Large Interaction Networks," *BMC Bioinformatics*, vol. 7, no. 1, article 207, 2006.

[15] M. Li, J. Chen, J. Wang, B. Hu, and G. Chen, "Modifying the DPClus Algorithm for Identifying Protein Complexes Based on New Topological Structures," *BMC Bioinformatics*, vol. 9, no. 1, article 398, 2008.

[16] B. Adamcsek, G. Palla, I.J. Farkas, I. Derenyi, and T. Vicsek, "CFinder: Locating Cliques and Overlapping Modules in Biological Networks," *Bioinformatics*, vol. 22, no. 8, pp. 1021-1023, 2006.

[17] G. Liu, L. Wong, and H.N. Chua, "Complex Discovery from Weighted PPI Networks," *Bioinformatics*, vol. 25, no. 15, pp. 1891-1897, 2009.

[18] M. Wu, X. Li, C. Kwoh, and S. Ng, "A Core-Attachment Based Method to Detect Protein Complexes in PPI Networks," *BMC Bioinformatics*, vol. 10, no. 1, article 169, 2009.

[19] X. Zhang, D. Dai, and X. Li, "Protein Complexes Discovery Based on Protein-Protein Interaction Data via a Regularized Sparse Generative Network Model," *IEEE/ACM Trans. Computational Biology and Bioinformatics*, vol. 9, no. 3, pp. 857-870, May/June 2012.

[20] T. Nepusz, H. Yu, and A. Paccanaro, "Detecting overlapping protein complexes in protein-protein interaction networks,", *Nat. Methods*, vol. 9, pp. 471-472, 2012.

[21] W.W.M. Lam, and K.C.C. Chan, "Discovering Functional Interdependence Relationship in PPI Networks for Protein Complex Identification," *IEEE Trans. Biomedical Eng.*, vol. 59, no. 4, pp. 899-908, Apr. 2012.

[22] M. Li, X. Wu, J. Wang, and Y. Pan, "Towards the Identification of Protein Complexes and Functional Modules by Integrating PPI Network and Gene Expression Data," *BMC Bioinformatics*, vol. 13, no. 1, article 109, 2012.

[23] X. Zhang, D. Dai, O. Le, and H. Yan, "Detecting overlapping protein complexes based on a generative model with functional and topological properties," *BMC Bioinformatics*, vol. 15, article 186, 2014.

[24] M. Ashburner, C.A. Ball, J.A. Blake, D. Botstein, H. Butler, J.M. Cherry, A.P. Davis, K. Dolinski, S.S. Dwight, J.T. Eppig, M.A. Harris, D.P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J.C. Matese, J.E. Richardson, M. Ringwald, G.M. Rubin, and G. Sherlock, "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics*, vol. 25, no. 1, pp. 25-29, 2000.

[25] C.H. Wu, R. Apweiler, A. Bairoch, D.A. Natale, W.C. Barker, B. Boeckmann, S. Ferr, E. Gasteiger, H. Huang, R. Lopez, M. Magrane, M.J. Martin, R. Mazumder, C. O'Donovan, N. Redaschi, and B. Suzek, "The universal protein resource (UniProt): an expanding universe of protein information," *Nucleic acids research*, vol. 34, D187-D191, 2006.

[26] T. Bäck, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, 1st ed. Oxford, UK: Oxford Univ. Press, 1996.

[27] Y. Hong, S. Kwong, H. Xiong, and Q. Ren, "Genetic-guided semi-supervised clustering algorithm with instance-level constraints," in *Proc. 10th Annu. Conf. Genetic and evolutionary computation*, 2008, pp. 1381–1388.

[28] S.R. Collins, P. Kemmeren, X. Zhan, J.F. Greenblatt, F. Spencer, F.C.P. Holstege, J.S. Weissman, and N.J. Krogan, "Toward a comprehensive atlas of the physical interactome of saccharomyces cerevisiae," *Molecular&Cellular Proteomics*, vol. 6, pp. 439-450, Mar. 2007.

[29] N.J. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A. P. Tikuisis, T. Punna, J. Peregrin-Alvarez, M. Shales, X. Zhang, M. Davey, M. D. Robinson, A. Paccanaro, J.E. Bray, A. Sheung, B. Beattie, D.P. Richards, V. Canadien, A. Lalev, F. Mena, P. Wong, A. Starostine, M.M. Canete, J. Vlasblom, S. Wu, C. Orsi, S.R. Collins, S. Chandran, R. Haw, J.J. Rilstone, K. Gandi, N.J. Thompson, G. Musso, P. St Onge, S. Ghanny, M.H.Y. Lam, G. Butland, A. Altaf-Ul, S. Kanaya, A. Shilatifard, E. O'Shea, J.S. Weissman, C.J. Ingles, T.R. Hughes, J. Parkinson, M. Gerstein, S.J. Wodak, A. Emili, and J.F. Greenblatt, "Global Landscape of Protein Complexes in the Yeast Saccharomyces cerevisiae," *Nature*, vol. 440, no. 7084, pp. 637-643, 2006.

[30] I. Xenarios, L. Salwı´nski, X.J. Duan, P. Higney, S. Kim, and D. Eisenberg, "DIP, the Database of Interacting Proteins: A Research Tool for Studying Cellular Networks of Protein Interactions," *Nucleic Acids Research*, vol. 30, no. 1, pp. 303-305, 2002.

[31] C. Stark, B. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, "BioGRID: A General Repository for Interaction Datasets," *Nucleic Acids Research*, vol. 34, no. Suppl. 1, pp. D535-D539, 2006.

[32] E. Camon, M.E. Camon, M. Magrane, D. Barrell, V. Lee, E. Dimmer, J. Maslen, D. Binns, N. Harte, R. Lopez, and R. Apweiler, "The Gene Ontology Annotation (GOA) Database: Sharin Knowledge in Uniprot with Gene Ontology," *Nucleic Acids Research*, vol. 32, no. Suppl. 1, pp. D262-D266, 2003.

[33] S. Pu, J. Wong, B. Turner, E. Cho, and S.J. Wodak, "Up-to-Date Catalogues of Yeast Protein Complexes," *Nucleic Acids Research*, vol. 37, no. 3, pp. 825-831, 2009.

[34] H.W. Mewes, D. Frishman, U. Guldener, G. Mannhaupt, K. Mayer, M. Mokrejs, B. Morgenstern, M. Munsterkotter, S. Rudd, and B. Weil, "MIPS: A Database for Genomes and Protein Sequences," *Nucleic Acids Research*, vol. 30, no. 1, pp. 31-34, 2002.

[35] U. Güldener, M. Münsterkötter, G. Kastenmüller, N. Strack, J. van Helden, C. Lemer, J. Richelles, S.J. Wodak, J. García-Martínez, J.E. Pérez-Ortín, H. Michael, A. Kaps, E. Talla, B. Dujon, B. André, J.L. Souciet, J. De Montigny, E. Bon, C. Gaillardin, and H.W. Mewes, "CYGD: The Comprehensive Yeast Genome Database," *Nucleic Acids Research*, vol. 33, no. suppl. 1, pp. D364-D368, 2005.

[36] A. Ruepp, B. Brauner, I. Dunger-Kaltenbach, G. Frishman, C. Montrone, M. Stransky, B. Waegele, T. Schmidt, O.N. Doudieu, V. Stümpflen, and H.W. Mewes, "CORUM: The Comprehensive Resource of Mammalian Protein Complexes," *Nucleic Acids Research*, vol. 36, no. suppl. 1, pp. D646-D650, 2008.

[37] J. Wang, G. Chen, B. Liu, M. Li, and Y. Pan, "Identifying Protein Complexes from Interactome Based on Essential Proteins and Local Fitness Method," *IEEE Trans. NanoBioscience*, vol. 11, no. 4, pp. 324-335, Dec. 2012.

[38] G. Liu, C.H. Yong, H.N. Chua, and L. Wong, "Decomposing PPI Networks for Complex Discovery," *Proteome Science*, vol. 9, no. Suppl. 1, article S15, 2011.

[39] E.I. Boyle, S. Weng, J. Gollub, H. Jin, D. Botstein, J.M. Cherry, and G. Sherlock, "GO::TermFinder—Open Source Software for Accessing Gene Ontology Information and Finding Significantly Enriched Gene Ontology Terms Associated with a List of Genes," *Bioinformatics*, vol. 20, no. 18, pp. 3710-3715, 2004.

[40] J.M. Cherry, C. Adler, C. Ball, S.A. Chervitz, S.S. Dwight, E.T. Hester, Y. Jia, G. Juvik, T. Roe, M. Schroeder, S. Weng, and D. Botstein, "SGD: Saccharomyces Genome Database," *Nucleic Acids Research*, vol. 26, no. 1, pp. 73-79, 1998.

[41] P. Jiang, and M. Singh, "SPICi: a fast clustering algorithm for large biological networks," *Bioinformatics*, vol. 26, no. 8, pp. 1105-1111, Apr., 2010.

[42] W. Peng, J. Wang, B. Zhao, and L. Wang, "Identification of protein complexes using weighted pagerank-nibble algorithm and core-attachment structure," *IEEE/ACM Trans. Computational Biology and Bioinformatics*, vol. 12, no. 1, pp. 179-192, Jan./Feb. 2015.

[43] Y. Wang, R. Wang, X. Zhang, and L. Chen, "Establishing protein functional linkage in a systematic way," *Lect. Notes Oper. Res.*, vol. 7, pp. 75-88, 2007.

[44] X. Zhang, R. Wang, Y. Wang, and J. Wang, "Modularity optimization in community detection of complex networks," *Europhysics Letters*, vol. 87, no. 3, pp. 38002, 2009.

[45] J. Ren, J. Wang, M. Li, and L. Wang, "Identifying protein complexes based on density and modularity in protein-protein interaction network," *BMC Systems Biology*, vol. 7, no. 4, article 1, 2013.

[46] M. Li, J. Wang, J. Chen, and Z. Cai, "Identifying the overlapping complexes in protein interaction networks," *International Journal of Data Mining and Bioinformatics*, vol. 4, no.1, pp. 91-108, 2010.

[47] B. Zhao, J. Wang, M. Li, F. Wu, and Y. Pan, "Detecting protein complexes based on uncertain graph model," *IEEE/ACM Trans. Computational Biology and Bioinformatics*, vol. 11, no. 3, pp. 486-497, May/Jun. 2014.

[48] M. Li, T. Yu, X. Wu, J. Wang, F. Wu, and Y. Pan, "C-DEVA: detection, evaluation, visualization and annotation of clusters from biological networks," *BioSystems*, vol. 150, pp. 78-86, 2016.

[49] T. He, and K.C.C. Chan, "Evolutionary community detection in social networks," in *Proc. 2014 IEEE Congress on Evolutionary Computation*, 2014, pp. 1496-1503.

Tiantian He received BEng degree in computer science and technology from North China University of Technology, Beijing, China in 2008 and MSc degree in information systems from The Hong Kong Polytechnic University, Hong Kong in 2012. Currently he is working towards his doctor degree in Department of Computing, The Hong Kong Polytechnic University. His research interests include graph clustering, evolutionary computation and bioinformatics.

Keith C.C. Chan received the BMath (Hons.) degree in computer science and statistics in 1984 and the MASc and PhD degrees in systems design engineering in 1985 and 1989, respectively, from the University of Waterloo, Ontario, Canada. Soon after graduation, he worked as a software analyst for the development of multimedia and software engineering tools at the IBM Canada Laboratory in Toronto, Canada. He joined the Hong Kong Polytechnic University in 1994, where he is currently a professor in the Department of Computing. His research interests include bioinformatics, data mining, and software engineering. He has over 200 publications in these areas, and his research is supported both by government research funding agencies and the industry. Chan serves on the editorial board of five journals and has also been serving on the program committees of numerous conferences.