BIOINFORMATICS REVIEW

Vol. 30 no. 10 2014, pages 1343–1352
doi:10.1093/bioinformatics/btu034

Data and text mining

Advance Access publication January 22, 2014

Algorithms and tools for protein–protein interaction networks clustering, with a special focus on population-based stochastic methods

Clara Pizzuti[1,†] and Simona E. Rombo[2,*,†]

[1]Institute for High Performance Computing and Networking (ICAR), National Research Council of Italy (CNR), Via P. Bucci 41C, 87036 Rende (CS) and [2]Department of Mathematics and Computer Science, University of Palermo, Via Archirafi 34, 90123 Palermo (PA), Italy

Associate Editor: Jonathan Wren

ABSTRACT

Motivation: Protein–protein interaction (PPI) networks are powerful models to represent the pairwise protein interactions of the organisms. Clustering PPI networks can be useful for isolating groups of interacting proteins that participate in the same biological processes or that perform together specific biological functions. Evolutionary orthologies can be inferred this way, as well as functions and properties of yet uncharacterized proteins.

Results: We present an overview of the main state-of-the-art clustering methods that have been applied to PPI networks over the past decade. We distinguish five specific categories of approaches, describe and compare their main features and then focus on one of them, i.e. population-based stochastic search. We provide an experimental evaluation, based on some validation measures widely used in the literature, of techniques in this class, that are as yet less explored than the others. In particular, we study how the capability of Genetic Algorithms (GAs) to extract clusters in PPI networks varies when different topology-based fitness functions are used, and we compare GAs with the main techniques in the other categories. The experimental campaign shows that predictions returned by GAs are often more accurate than those produced by the contestant methods. Interesting issues still remain open about possible generalizations of GAs allowing for cluster overlapping.

Availability and implementation: We point out which methods and tools described here are publicly available.

Contact: simona.rombo@math.unipa.it

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on June 16, 2013; revised on January 14, 2014; accepted on January 15, 2014

1 INTRODUCTION

Biological networks have received much attention in the past few years since they model the complex interactions occurring among different components in the cell (Atias and Sharan, 2012; De Virgilio and Rombo, 2012; Ferraro *et al.*, 2011; Panni and Rombo, 2013; Sharan *et al.*, 2007). Thanks to the development of advanced high-throughput technologies (von Mering *et al.*, 2002), large volumes of experimental data on protein–protein interactions (PPIs) have been made available. Special kinds of biological networks, PPI networks, are where the cellular components under analysis are proteins. In a PPI network, nodes correspond to proteins and edges correspond to pairwise interactions between proteins. Proteins are organized into different putative complexes, each performing specific tasks in the cell (Hartwell *et al.*, 1999; Pereira *et al.*, 2004). Proteins interacting with each other often participate in the same biological processes or can be associated with specific biological functions being strongly related (Tornw and Mewes, 2003). It is worth pointing out that interacting proteins can belong to 'protein complexes' or 'functional modules' with different biological meanings. A protein complex is a molecular machine consisting of several proteins that bind to each other at the same place and time, whereas a functional module consists of a few proteins that control or perform a particular cellular function by interacting among themselves (these proteins do not necessarily interact at the same time and place). However, pairwise protein interaction data stored in public databases usually do not distinguish explicitly between such temporal and spatial information about PPIs. In the following, we will refer either to 'complex' or 'module' to indicate a group of proteins that are connected by a large number of pairwise interactions.

The detection of protein complexes using PPI networks can help in understanding the mechanisms regulating cell life, in describing the evolutionary orthology signal [e.g. Jancura *et al.* (2011)], in predicting the biological functions of uncharacterized proteins, and, more importantly, for therapeutic purposes. The problem of detecting protein complexes using PPI networks can be computationally addressed using clustering techniques. Clustering consists of grouping data objects into groups (also called *clusters* or *communities*) such that the objects in the same cluster are more similar to each other than the objects in the other clusters (Jain, 1988). Possible uncharacterized proteins in a cluster may be assigned to the biological function recognized for that module, and groups of proteins performing the same tasks can be singled out this way. As observed in (Fortunato, 2010), a generally accepted definition of 'cluster' does not exist in the context of networks, as it depends on the specific application domain. However, it is widely accepted that a community should have more internal than external connections. For biological

*To whom correspondence should be addressed.

†The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

networks, the most common assumption is that clusters are groups of highly connected nodes, although recently the notion of community intended as a set of topologically similar links has been successfully used in Ahn *et al.* (2010) and Solava *et al.* (2012). We also observe that many different clustering techniques have been proposed for graph analysis [e.g. the minimum cut algorithm in Hartuv *et al.* (2000) and the survey on graph clustering proposed in Schaeffer (2007)].

In this work, we present a description of the state-of-the-art clustering methods for complex detection in PPI networks that have been proposed over the past decade. We mainly focus on methods that only use graph topology for detecting clusters and do not use similarity measures between proteins as described by vectors of features (such as protein amino acid sequences or protein domain composition). We first provide an overview of the main clustering techniques proposed for PPI networks, by placing them into five different categories. Then, we analyze in more details one such category, i.e. population-based stochastic methods, which is less explored than the others. In particular, to study how the capability of Genetic Algorithms (GAs) in clustering PPI networks varies, when different topology-based fitness functions are used, we provide an experimental evaluation based on some validation measures widely used in the literature. This aspect has not been considered in the other surveys presented in the literature, such as Aittokallio and Schwikowski (2006), Brohèe and van Helden (2006), Li *et al.* (2010), Lin *et al.* (2006), Pavlopoulos *et al.* (2011), Pizzuti *et al.* (2012), Pržulj (2005) and Wang *et al.* (2010), and . We also compare the results obtained by GAs with those returned by the main techniques in the other categories, and we discuss the open challenges to stimulate further research. Finally, to guide the reader in the choice and application of existing PPI network clustering tools, we also provide information about their availability.

2 PPI CLUSTERING METHODS

Clustering approaches to PPI networks can be broadly categorized as *topology-free* and *graph-based* ones. Topology-free approaches use traditional clustering techniques using notions of distance between proteins that do not take into account the topology of the network. Graph-based clustering approaches consider instead the topology of the network, and usually rely on specialized clustering techniques. In the following, suitable descriptions of the better known and most recent graph-based approaches are provided, and the advantages and drawbacks of each method are pointed out. Because of the huge number of proposals, a complete list of them is beyond the aim of this article. It is worth noting that graph-based clustering techniques are deeply studied in other research fields, such as physics and data mining, and are known as *community detection methods* (Girvan and Newman, 2002).

In graph-based techniques, a PPI network is modeled as an undirected graph $G = (V, E)$, where the nodes V correspond to proteins, and the edges E correspond to pairwise interactions. We classified the considered approaches in five main categories:

(1) Local neighborhood Density search (LD);
(2) Cost-based Local search (CL);
(3) Flow Simulation (FS);
(4) Link Clustering (LC);
(5) Population-based Stochastic search (PS).

For each of the categories listed above, we provide a short description of the general goals of the approach and summarize the main features of a selection of methods belonging to such a category. Figure 1 at the end of this section illustrates the main features of the discussed categories, whereas Table 1 summarizes the characteristics of each method and highlights which software is publicly available. This can hopefully be of practical help for the reader who is interested in using PPI network clustering techniques.

2.1 Local neighborhood density search

Methods in this category are based on local optimization strategies designed to find dense subgraphs (i.e. each node is connected to many other nodes in the same subgraph) within the input PPI network. They aim at maximizing the density of each found subgraph. We recall in the following the main LD techniques applied to PPI networks.

MCODE (Bader and Hogue, 2003). This approach detects dense and connected regions by weighting nodes on the basis of their local neighborhood density. To this end, the *k-core* concept is applied. A k-core is a subgraph in which each vertex has a degree of at least k. The highest k-core of a graph is the most densely connected subgraph. The core-clustering coefficient of a node v is the density of the highest k-core of the vertices directly connected to it, included v itself. The weight of a node is then defined as the product of the node core-clustering coefficient and the highest k-core of its neighborhood. MCODE selects as seed cluster the vertex with the highest weight, and neighboring nodes are recursively included in the cluster if their weight is above a fixed threshold. When no more nodes can be added to the cluster, the process stops and it is repeated for the next-highest unexamined node. The densest regions of the graph are identified this way.

DPClus (Altaf-Ul-Amin *et al.*, 2006). This method is based on the concepts of *node weight* and *cluster property*. The former is used to select a seed node further expanded by the iterative addition of neighbors. The latter is used to terminate the expansion process. DPClus needs two input parameters, i.e. a minimum density value d_{in} and a minimum value of cluster property cp_{in}, to decide the insertion of a new neighboring node in a cluster. These two parameters influence both the kind and the number of the returned clusters. The algorithm first ranks all the nodes with respect to their weights and chooses as seed the node having the highest weight. It then expands the cluster containing the seed node by the addition of neighboring nodes, provided that the density and the cluster property of the expanded cluster do not diminish below the fixed initial input values. Once a cluster is generated, its nodes are removed from the graph and the next cluster is extracted by using only the remaining nodes until all the nodes have been assigned to a cluster. The algorithm allows also to generate overlapping clusters.

SWEMODE (Lubovac *et al.*, 2006). This approach identifies dense subgraphs using suitable network measures that combine functional information with topological properties of the input network. The algorithm ranks all the nodes by assigning a weight according to their neighborhood cohesiveness. The highest ranked nodes are used as seeds for candidate modules. Then the neighborhood of each seed protein is explored to find densely connected proteins with high functional similarity, obtained according to the Gene Ontology annotations (Asburner *et al.*, 2000). Proteins satisfying a parameter, i.e. the *Node Weight Percentage* (NWP), are included in the current module. This module prediction procedure is repeated for the second highest ranked node and so on, until all the nodes have been examined. NWP influences the number of output clusters: high values correspond to few and not necessarily dense clusters, low values produce many modules with few proteins. The suggested value of NWP to obtain meaningful modules is 0.4.

DECAFF (Li *et al.*, 2007). It addresses two major limitations plaguing protein interaction data, namely, *incompleteness* and *noise*. The method consists of three main steps: (i) detect the local dense neighborhoods of each protein, (ii) merge the local subgraphs based on the similarity degree

between neighborhoods and (iii) filter away possible false complexes. To find the local dense neighborhood of a node, first its local cliques are obtained by adapting the method LCMA (Li *et al.*, 2005) proposed by the same authors. Local cliques are found by the iterative removal of nodes with the lowest degree. However, dense non-clique subgraphs for each node could be parts of complexes. A hub removal algorithm is then applied that removes hub proteins (i.e. those having the highest degree) and their edges from the graph, and this process is repeated on its connected components until a dense group of proteins has been obtained. Hub proteins are finally inserted back into the cluster. As regards merging, the authors introduce the concept of *neighborhood affinity*, which measures the similarity between two overlapping neighborhoods. Maximal dense neighborhoods are obtained by merging local dense neighborhoods having an affinity value above a given threshold. In the last step, dense subgraphs with low reliability scores are deleted.

CFinder (Adamcsek *et al.*, 2006). It is based on the clique percolation concept [see Derenyi *et al.* (2005) and Palla *et al.* (2005)]. The idea behind this method is that a cluster can be interpreted as the union of small fully connected subgraphs that share nodes, where a parameter is used for specifying the minimum number of shared nodes. CFinder extracts all the maximal complete subgraphs, i.e. the maximal cliques, in the input PPI network. Then a clique–clique overlap matrix is built such that each entry contains the number of common nodes between the two corresponding cliques, and each diagonal entry is the clique size k. The k-cliques-communities can be found by deleting every entry off the diagonal having a value $<k-1$ and every diagonal entry $<k$. The remaining separate components correspond to the k-cliques-communities. CFinder allows for overlap between communities.

RANCoC (Pizzuti and Rombo, 2012a), MF-PINCoC (Pizzuti and Rombo, 2008), PINCoC (Pizzuti and Rombo, 2007). These algorithms are based on greedy local expansion. They expand a single protein randomly selected by adding/removing proteins to improve a given quality function, based on the concept of co-clustering (Madeira and Oliveira, 2004). To escape poor local maxima, with a given probability, the protein causing the minimal decrease of the quality function is removed in MF-PINCoC and PINCoC. Instead RANCoC removes, with a fixed probability, a protein at random, even if the value of the quality function diminishes. This strategy is more efficient in terms of computation than that applied in the methods (Pizzuti and Rombo, 2007, 2008), and it is more efficacious in avoiding entrapments in local optimal solutions. All three algorithms work until either a preset of maximum number of iterations has been reached or the solution cannot further be improved. Both MF-PINCoC and RANCoC allow for overlapping clusters.

PCP (Chua *et al.*, 2007). The approach proposed in Chua *et al.* (2007) preprocesses the input PPI network by the computation of a topological weight, i.e. the *FS-weight*, that estimates the reliability of the interactions, i.e. the likelihood that two proteins share functions. PCP first finds all the maximal cliques of the input network, and then it merges them by using the concept of *inter-cluster density (ICD)*. The ICD measures the interconnectedness between two subgraphs by the computation of the FS-weight density of the inter-cluster interactions between the proteins not belonging to both subgraphs. Given two clusters, a high value of their ICD means that the two clusters are highly connected. The merge procedure considers an initial graph constituted by partial cliques, i.e. strongly connected components composed by cliques and adds an edge between two partial cliques if their ICD value is above a fixed threshold. This is repeated until no further merge is possible.

DME (Georgii *et al.*, 2009). This is a technique for extracting dense modules from a weighted interaction network. The method detects all the node subsets that satisfy a user-defined minimum density threshold and returns only locally maximal solutions, i.e. modules where all the direct supermodules (containing one additional node) do not satisfy the minimum density threshold. The obtained modules are ranked according to the *P*-value as computed from a bootstrap procedure. An interesting property of this method is that it allows to incorporate constraints with respect to additional data sources.

MCODE, DPClus and SWEMODE have a similar strategy. First, they define the weight of each node, then they choose the node with highest weight as seed cluster and finally they add neighboring nodes to the current cluster if some threshold parameters are satisfied. The main difference concerns the definition of the weight. In contrast to SWEMODE, which combines also semantic information coming from the Gene Ontology (Asburner *et al.*, 2000) database, MCODE and DPClus use only the network topology. Thus, the former approach should give increased confidence in the predicted function, although it is worth pointing out that it does not allow the participation of a protein to more than one group. In contrast to CFinder, DECAFF and PCP, which use the concepts of maximal k-clique or local cliques to grow clusters, PINCoC, MF-PINCoC and RANCoC rely on co-clustering to find dense subgraphs. All such methods need some parameters that biases the number and kind of output clusters.

2.2 Cost-based local search

These approaches divide the input graph into connected subgraphs (i.e. the output modules) by a cost function that guides the search toward a best partition.

SL (Samantha and Liang, 2003). Samantha and Liang propose a clustering method, here called SL by the names of the authors, based on the idea that if two proteins share a number of common interaction partners larger than what would be expected in a random network then they should be clustered together. The method assesses the statistical significance of forming shared partnership between two proteins using the concept of *P*-value for a pair of proteins. The *P*-values of all the protein pairs are computed and stored into a similarity matrix. The protein pair with the lowest *P*-value is chosen to form the first group, and the corresponding rows and columns of the matrix are merged in a new row/column. The new *P*-value of the merged row/column is the geometric mean of the *P*-values of the corresponding elements. This step is repeated by adding new proteins to the current cluster until a threshold value has been reached. The whole process is repeated until all the proteins have been clustered.

RNSC (King *et al.*, 2004). This algorithm explores the solution space of all possible clusterings to minimize a cost function that reflects the number of inter-cluster and intra-cluster edges. The algorithm begins with a random clustering and attempts to find a clustering with the best cost repeatedly moving one node from a cluster to another. A *tabu* list of moves is used to forbid cycling back to previously examined solutions. To output clusters that are likely to correspond to true protein complexes, thresholds for minimum cluster size, minimum density and functional homogeneity must be set. Only clusters satisfying these criteria are given as the final result. This obviously implies that many proteins are not assigned to any cluster.

Farutin (Farutin *et al.*, 2006). Farutin *et al.* measure the *community strength* of a module quantifying the preferential attachment of each element to the other ones in the same module with respect to how unlikely it is observed in a random graph. Because it is necessary to count the number of edges in the graph, the authors assume a random graph as the null model, where an edge is the random variable. To identify clusters, a greedy approach that searches for a set of nodes in the network with small values of community strength is adopted. A list of two adjacent nodes is considered and then nodes that lead to the largest decrease of the community score are added. This is repeated for each connected node pair, thus the obtained clusters can partially overlap.

Qcut (Ruan and Zhang, 2008). Several community discovery algorithms have been proposed based on the optimization of a modularity-based function [see e.g. Fortunato (2010)]. Modularity measures the fraction of edges falling within communities, minus what would be expected if the edges were randomly placed. Qcut is an efficient heuristic algorithm

applied to detect protein complexes. It optimizes modularity combining spectral graph partitioning and local search. By optimizing modularity, communities that are smaller than a certain scale or that have relatively high inter-community density may be merged into a single cluster. To overcome this drawback, the authors introduce an algorithm that runs Qcut recursively to divide a community into subcommunities. To avoid overpartitioning, a statistical test is used for deciding whether a community contains intrinsic subcommunity.

ModuLand (Kovacs *et al*., 2010). ModuLand is a family of integrative methods for detecting overlapping network modules as hills of an influence function-based centrality-type community landscape and including several widely used modularization methods as special cases. Several algorithms obtained from ModuLand provide an efficient analysis of weighted and directed networks, return overlapping modules with high resolution, uncover a detailed hierarchical network structure allowing an efficient zoom-in analysis of large networks and provide the extraction of key network nodes. It is implemented as a Cytoscape (Shannon *et al*., 2003) plug-in.

OCG (Becker *et al*., 2012). This recent approach decomposes the input network into overlapping clusters and assigns multifunctional proteins to the found partitions. It is based on the principle of covering the graph with initial overlapping classes, stored as leaves of a tree, that are progressively and hierarchically fused maximizing a modularity function. The starting point is the set of all the nodes taken as singletons. This initial partition has a null modularity, as there are no internal edges. Then, although modularity increases, the two clusters whose union gives the most positive maximal gap are merged. The gap is equal to the difference between the modularity values when the two clusters are separated or joined together. A hierarchy of nested clusters is built iteratively, and the algorithm stops when no further fusions can produce a gain in modularity.

SL and Farutin are greedy approaches, where the former optimizes the concept of *P*-value to build clusters recursively merging protein pairs having the smallest *P*-value, and the latter optimizes the community strength of a module. In contrast to RNSC, that moves the nodes among the clusters to improve its cost function, both Qcut and OCG merge the clusters for optimizing the modularity, and they use the strategy recently proved in Fortunato and Barthélemy (2007) for overcoming the resolution limit problem. This strategy relies on the fact that methods maximizing modularity could not discover structures at small scales, hidden within large groups. ModuLand is based on a different approach, as it uses different influence functions of nodes to find regions where nodes influence each other. These regions are then explored to obtain local maxima corresponding to communities.

2.3 Flow simulation

Methods based on the FS approach mimic the spread of information on a network, using random walk (Lovasz, 1996), or biological knowledge for passing information between proteins in the network to cluster proteins.

MCL (Enright *et al*., 2002; Van Dongen, 2008). In a random walk, the direction to be followed at each node is given by chance. MCL simulates many random walks (or flows) within a graph by strengthening flow where it is strong, and weakening it where it is weak. By repeating this process, a number of regions come out with strong internal flow (the clusters), separated by boundary with no flow. The flow is simulated by algebraic operations on a stochastic Markov matrix associated with the input graph, such as flow expansion and an inflation operator that raises each entry of the matrix to a given power, and then rescales the matrix so that the column sum equals 1. By repeating a number of times squaring, inflating and scaling the matrix tends to an equilibrium state that shows the cluster structure. The inflation parameter influences the number of clusters.

RRW (Macropol *et al*., 2009). This algorithm starts with the choice of a protein as initial cluster, and then expands it including the protein with the highest proximity to that cluster. This iterative process is repeated either k times or until a stopping condition is met, to obtain clusters of size $\leq k$. All significant overlapping clusters are recorded and post-processed to remove redundant clusters based on a given overlap threshold. Random walks with restarts are used to find the closest proteins to a given cluster. To increase the algorithm's speed, the random walk results from a given cluster are computed using the linear combinations of precomputed random walk results obtained starting from single proteins.

IFB (Cho *et al*., 2006). This algorithm integrates topological and biological knowledge to select a number of informative proteins and simulates the information flow through the network from each informative protein. The weighted degree of a node is defined as the sum of the weights of the edges containing that node, and the weight of an edge is computed using the correlation between the expression profiles of the two genes encoding the proteins linked by that edge. This weighted degree provides the semantic information of a node. A variant of the approach is presented by the same authors in Cho *et al*. (2007) to compute the weight of an edge and, consequently, to select the informative proteins.

STM (Hwang *et al*., 2006). This method finds clusters of arbitrary shape modeling the dynamic relationships between proteins of a PPI network as a signal transduction system. The overall signal transduction behavior between two proteins of the network is defined to evaluate the perturbation of one protein on the other one, both biologically and topologically. The signal transduction behavior is modeled using the Erlag distribution. The algorithm starts with the computation of the transduction signal for all the protein pairs. Then, the cluster representatives are selected for each cluster. The cluster (or module) representatives are the most influential nodes, where influential means having the highest scores of the transduction signal on each node of the module. From these nodes, preliminary clusters are created aggregating each node w to the module having as representative the node v for which the signal trasduction is the highest. Finally, these clusters are merged if there exists a substantial number of interconnections.

In contrast to MCL, which simulates the behavior of many walkers starting from the same point and moving within a graph in a random way, IFB aims at imitating the scattering of information inside the network from some informative nodes, to identify the proteins influenced from the starting nodes. The main difference between the two approaches is that the former uses only the network topology, whereas the latter relies on semantic information. Differently than STM, which models the dynamic relations between proteins using a signal transduction model, RRW uses random walks to compute the nearest proteins of a cluster.

2.4 Link clustering

LC methods group the set of edges rather than the set of nodes of the input network, often exploiting suitable techniques to compute edge similarity (Kuchaiev *et al*., 2011; Milenkovic and Pržulj, 2008; Pržulj, 2007; Solava *et al*., 2012). In Evans and Lambiotte (2009, 2010) and Pizzuti (2009), LC is used to discover overlapping communities in complex networks different than PPI networks. In the following, we summarize two LC techniques applied to PPI networks.

Pereira (Pereira *et al*., 2004). Given an input PPI network N, the approach by Pereira *et al*. builds the corresponding line graph G. In particular, a vertex of G represents an edge of N, and two vertices are adjacent in G if and only if their corresponding edges in N share a common endpoint. Thus, each node of G represents an interaction between two proteins, and each edge represents pairs of interactions connected by a common protein. Pereira *et al*. apply MCL (Enright *et al*., 2002) on G and detect this way overlapping protein modules in N.

Ahn (Ahn *et al*., 2010). Ahn *et al*. propose an agglomerative LC approach to group links into topologically related clusters. The algorithm applies a hierarchical method based on the notion of *link similarity*, which is used to find the pair of links with the largest similarity to merge their respective communities. The similarity between two links takes into

account the size of both the intersection and the union of their neighborhoods. The agglomerative process is repeated until all the links belong to a single cluster. To find a meaningful community structure, it is necessary to decide where the built dendrogram must be cut. To this end, the authors introduce the concept of *partition density* to measure the quality of a link partitioning, and they choose the partitioning having the best partition density value.

LC approaches have the main advantage that nodes are automatically allowed to be present in multiple communities, without the necessity of performing multiple clustering on the set of edges. As a negative point, if the input network is dense then LC may become computationally expensive. We also observe that the performances of these techniques may depend on the link similarity measure they adopt. This issue is addressed by Solava *et al.* (2012), where a new similarity measure, extending that proposed in by Pržulj (2007), has been defined. In particular, this measure is based on the topological similarity of edges, computed by taking into account non-adjacent, though close, edges and counting the number of graphlets (i.e. small induced subgraphs containing from 2 to 5 nodes) each edge touches.

2.5 Population-based stochastic search

The GAs (Goldberg, 1989) are a class of adaptive general-purpose search techniques inspired by natural evolution. They have been proposed by Holland (1975) in the early 1970s as computer programs that simulate the evolution process in nature. A standard GA evolves a constant size population of elements (i.e. *chromosomes*), using the genetic operators of *reproduction*, *recombination* and *mutation*. Each chromosome represents a candidate solution to a given problem, and it is associated with a *fitness value* that reflects how good it is, with respect to the other solutions in the population. The reproduction operator copies elements of the current population into the next generation with a probability proportionate to their fitness (this strategy is also called *roulette wheel selection scheme*). The recombination operator generates two new chromosomes crossing two elements of the selected population proportionate to their fitness. The mutation operator randomly alters the chromosomes.

PS has been used to develop algorithms for network community detection, although only the works summarized later in the text have been applied to PPI networks.

CGA (Liu and Liu, 2006). Liu and Liu propose this algorithm for enumerating maximal cliques, based on chaos optimization and GAs. As the authors state, there are two main differences between standard GAs and CGA. The first one is that CGA uses chaotic variables to determine the range of initial populations; the second difference is that individuals having highest fitness values are directly put in the next generation, to avoid being changed during the evolutionary process. Each chromosome is a binary string whose length is the number of edges, where 0 indicates that the edge remains in the next generation, whereas 1 that it is discarded. The used fitness function combines the notions of *clustering coefficient* and *number of nodes*.

IGA (Ravaee *et al.*, 2010). This approach uses GAs in combination with the concept of *artificial immune system*. IGA finds dense subgraphs generating a population of *antibodies*. Each antibody is a string of integers representing a permutation of vertices and some splitting bits. The length of an antibody is $2|V| - 1$, where V is the set of nodes. The $|V|$ integers in the odd positions represent the vertices of the graph, whereas separator bits are present in the even positions. A 0 value means separation of two nodes belonging to the same cluster, whereas a 1 value denotes a boundary between two clusters. The authors introduce specialized operators such as local and global mutations, plus an immune selection operator and a vaccination operator that injects previous knowledge into the current solution. Experiments on the DIP yeast network proved good results when compared with MCODE and CFinder.

GA-PPI (Pizzuti and Rombo, 2012b, 2013). More recently, Pizzuti and Rombo apply GAs to PPI networks, referred as GA-PPI, performing an extensive experimental evaluation aiming at exploring the capability of GAs to find clusters in PPI networks, when different topology-based fitness functions are used. The adopted representation of individuals is the graph-based adjacency representation, originally proposed in Park and Song (1989), where an individual of the population consists of n genes, each corresponding to a node of the graph modeling the PPI network. A value j assigned to the ith gene is interpreted as a link between the proteins i and j and implies that i and j belong to the same cluster. In particular, in Pizzuti and Rombo (2012b) the fitness functions of *conductance, expansion, cut ratio* and *normalized cut*, introduced by Leskovec *et al.* (2010), are employed, whereas in Pizzuti and Rombo (2013) the cost functions of the RNSC algorithm (King *et al.*, 2004) have been used. In the next section, *GAs* are more deeply explored and experiments comparing different fitness functions are shown.

The methods described earlier in the text use different representations of candidate solutions and different fitness functions. Individuals are represented by bit strings associated with the presence of edges in CGA, nodes in IGA and connections between pairs of nodes in GA-PPI. As for LC methods, CGA performances may become worse when the input networks have a large number of edges.

3 EXPERIMENTAL EVALUATION

In this section, we investigate the population-based methods, running IGA and GA-PPI on three yeast PPI networks, and we compare the results returned by these methods with those produced by MCODE, RNSC, MCL, Ahn, OCG and RanCoc on the same datasets. In particular, we chose MCODE (in the non-overlapping mode), RNSC and MCL, as they are the most popular and accurate techniques performing non-overlapping clustering. We chose Ahn, OCG and RanCoc, as they are among the most recent approaches allowing for cluster overlapping, and they have been shown to outperform their competitors [see also the experimental evaluations described in Ahn *et al.* (2010); Becker *et al.* (2012); Pizzuti and Rombo (2012a)]. As regards GA-PPI, the tested fitness functions are described in Section 3.2, and the parameters have been fixed as follows: population size 100, number of generations 100, elite reproduction 10% of the population size, roulette selection function, crossover 0.8 and mutation 0.2. These values have been chosen by taking into account the experimental evaluation reported in Pizzuti and Rombo (2012b). The implementation has been written in MATLAB 7.14 R2012a, using GAs and Direct Search Toolbox 2. As regards MCODE, RNSC and MCL, we used the parameter values reported in Brohée and van Helden (2006), optimized for precision. Ahn, IGA and OCG did not require any parameter setting, whereas for RanCoC, we set $mflip = 1000$, $mr = 1$, $P = 0.1$ and $r = 3$, as suggested in Pizzuti and Rombo (2012b).

We ran the methods on three different yeast PPI datasets. The first two are the same used by Zaki *et al.* (2012). In particular, they have filtered two networks, one used by Gavin *et al.* (2006) and another containing yeast protein interactions generated by six individual experiments (including interactions characterized by mass spectrometry technique and interactions produced using two-hybrid techniques) to delete unreliable interactions. They obtained 990 proteins with 4687 interactions for the first network, here referred to as Yeast-D1, and 1443 proteins with 6993 interactions for the second network, here denoted by Yeast-D2. The third dataset Y2H we considered is a PPI network

built on the interactions obtained by high-throughput yeast two-hybrid screening described in Yu *et al.* (2008), where self-edges have been eliminated according to Ahn *et al.* (2010) and Solava *et al.* (2012). Y2H has 1966 nodes and 2705 edges.

We considered three reference sets of gold standard complexes, Cmplx1 for Yeast-D1, Cmplx2 for Yeast-D2 and Cmplx3 for Y2H, respectively. Cmplx1 includes 81 complexes of sizes at least 5 created from MIPS (Mewes *et al.*, 2000). Cmplx2 is made of 162 hand-curated complexes (size no less than four proteins) from MIPS (Mewes *et al.*, 2006). Finally, Cmplx3 includes 975 known and curated complexes from ftp://ftpmips.gsf.de/yeast/catalogues/complexcat.

In the following, first we describe the measures adopted to evaluate the methods, then we briefly summarize the used fitness functions, and finally we present the results.

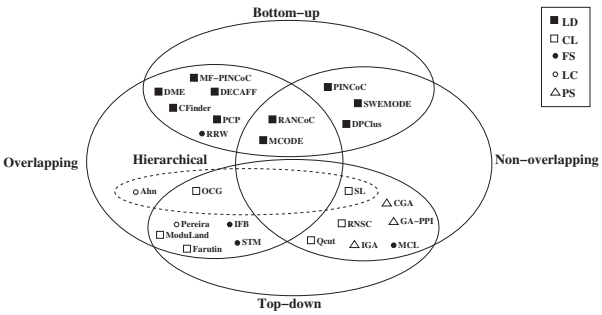

Fig. 1. Features of the considered clustering categories

3.1 Validation measures

To assess the quality of the results, we adopted as validation measures *precision* P, *recall* R and *F-measure* F_m, that have been widely applied in the literature (Altaf-Ul-Amin *et al.*, 2006; Bader and Hogue, 2003; Li *et al.*, 2008).

For the generic predicted cluster P_i and the generic known complex K_j, let $|P_i|$ and $|K_j|$ be their sizes, respectively. Furthermore, let $|P_i \cap K_j|$ be the size of the intersection set of the predicted cluster and the known complex. To evaluate how a predicted cluster P_i matches a known complex K_j, the *overlapping score* between P_i and K_j is defined as $OS(P_i, K_j) = \frac{|P_i \cap K_j|^2}{|P_i| \cdot |K_j|}$.

A known complex and a predicted cluster are considered a *match* (Li *et al.*, 2008) if $OS(P_i, K_j) \geq \sigma_{OS}$, i.e. their overlapping score is equal to or larger than a specific threshold σ_{OS}. To estimate the performance of algorithms for detecting protein complexes with respect to the overlapping score, the notions of *recall* and *precision*, as well as a cumulative measure called *F-measure* can be defined as follows.

Recall: $R = \frac{TP}{TP+FN}$ is the fraction of the true-positive predictions out of all the true predictions, where TP (true positive) is the number of the predicted clusters matched by the known complexes with $OS(P_i, K_j) \geq \sigma_{OS}$, and FN (false negative) is the number of the known complexes that are not matched by the predicted clusters.

Precision: $P = \frac{TP}{TP+FP}$ is the fraction of the true-positive predictions out of all the positive predictions, where FP (false positive) equals the total number of the predicted clusters minus TP.

Table 1. Summary of some characteristics of the methods

Method	Structure	Class	Simultaneous clustering	Overlapping	Unassigned proteins	URL
MCODE (Bader and Hogue, 2003)	Dense	LD	Yes	Yes	Yes	http://baderlab.org/Software/MCODE
DPCLus (Altaf-Ul-Amin *et al.*, 2006)	Dense	LD	Yes	No	Yes	http://kanaya.naist.jp/DPClus/
SWEMODE (Lubovac *et al.*, 2006)	Dense	LD	No	No	Yes	–
DECAFF (Li *et al.*, 2007)	Dense	LD	No	Yes	Yes	–
CFinder (Adamcsek *et al.*, 2006)	Dense	LD	Yes	Yes	Yes	http://hal.elte.hu/cfinder/wiki/?n=Main.Manual
PINCoc (Pizzuti and Rombo, 2007)	Arbitrary	LD	No	No	No	http://wwwinfo.deis.unical.it/rombo/co-clustering/
MF-PINCoC (Pizzuti and Rombo, 2008)	Arbitrary	LD	No	Yes	No	http://wwwinfo.deis.unical.it/rombo/co-clustering/
RANCoC (Pizzuti and Rombo, 2012a)	Arbitrary	LD	No	Yes	No	http://wwwinfo.deis.unical.it/rombo/co-clustering/
PCP (Chua *et al.*, 2007)	Dense	LD	No	Yes	Yes	http://www.comp.nus.edu.sg/wongls/projects/complexprediction/PCP-3aug07/
DME (Georgii *et al.*, 2009)	Dense	LD	No	Yes	Yes	people.kyb.tuebingen.mpg.de/georgii/dme.htmls
SL (Samantha and Liang, 2003)	Arbitrary	CL	No	No	No	–
RNSC (King *et al.*, 2004)	Dense	CL	Yes	No	Yes	http://www.cs.toronto.edu/juris/data/rnsc/
Farutin (Farutin *et al.*, 2006)	Arbitrary	CL	No	Yes	No	–
Qcut (Ruan and Zhang, 2008)	Dense	CL	Yes	No	No	http://cs.utsa.edu/jruan/Software.html
ModuLand (Kovacs *et al.*, 2010)	Dense	CL	Yes	Yes	No	http://www.linkgroup.hu/modules.php
OCG (Becker *et al.*, 2012)	Dense	CL	No	Yes	No	http://tagc.univ-mrs.fr/welcome/spip.php?rubrique197
MCL (Enright *et al.*, 2002)	Arbitrary	FS	Yes	No	No	http://micans.org/mcl/
RRW (Macropol *et al.*, 2009)	Arbitrary	FS	No	Yes	No	http://www.cs.ucsb.edu/kpm/software.html
IFB (Cho *et al.*, 2006)	Arbitrary	FS	No	Yes	Yes	–
STM (Hwang *et al.*, 2006)	Arbitrary	FS	Yes	Yes	Yes	–
Pereira (Pereira *et al.*, 2004)	Arbitrary	LC	Yes	Yes	No	–
Ahn (Ahn *et al.*, 2010)	Arbitrary	LC	No	Yes	No	http://barabasilab.neu.edu/projects/link communities/
CGA (Liu and Liu, 2006)	Dense	PS	Yes	No	No	–
IGA (Ravaee *et al.*, 2010)	Dense	PS	Yes	No	No	–
GA-PPI (Pizzuti and Rombo, 2012b, 2013)	Dense	PS	Yes	No	No	http://staff.icar.cnr.it/pizzuti/codes.html

Note: Column 1: method acronym and reference. Column 2: topological structure a method searches for (arbitrary or dense sub-graphs). Column 3: the class of the method. Column 4: if the method finds all clusters simultaneously. Column 5: if the method generates overlapping clusters. Column 6: if the method returns some unassigned proteins. Column 7: the link to the software implementing that method, if publicly available.

F-measure: $F_m = \frac{2 \cdot R \cdot P}{R+P}$ is a measure that takes into account both recall and precision. High values of F-measure means that both recall and precision are sufficiently high.

3.2 Fitness functions

Given a graph $G = (V, E)$ that models a PPI network, consider a cluster S of G having n_s nodes and m_s edges, such that $c_s = |\{(u, v)|u \in S, v \notin S\}|$ is the number of edges on the boundary of S, $d_s = |\{u \in S|(v, u) \in E\}|$ is the sum of degrees of the nodes of S and $c_s(v) = |\{(v, u)|u \notin S\}|$ is the number of cross-edges incident with v. Furthermore, $l_s(v) = |\{u \in S|(v, u) \notin E\}|$ is the number of nodes in S that are not connected to v, S_v is the cluster v belongs to and $N(v)$ is the set of neighbor nodes of v. Finally, let $\{S_1, \ldots, S_k\}$ be a partition of G in k clusters.

The metrics we used to catch the concept of quality of a clustering are summarized later in the text. Many of them have been experimented by (Leskovec *et al.*, 2010) for community detection in complex networks. Besides the measures used in Pizzuti and Rombo (2012b, 2013), we provide further experiments considering other two metrics, *internal density* (Leskovec *et al.*, 2010) and *community score* (Pizzuti, 2008).

Modularity (Newman and Girvan, 2004): $Q = \sum_{s=1}^{k} [\frac{2m_s}{m} - (\frac{d_s}{2m})^2]$ measures the expected number of edges between the nodes of a cluster S in a random graph with the same degree sequence.
Conductance (Shi and Malik, 2000): $CO = \sum_{s=1}^{k} \frac{c_s}{2m_s + c_s}$ measures the fraction of edges pointing outside the clustering.
Expansion (Radicchi *et al.*, 2004): $EX = \sum_{s=1}^{k} \frac{c_s}{n_s}$ measures the number of edges per nodes that point outside the clustering.
Cut ratio (Fortunato, 2010): $CR = \sum_{s=1}^{k} \frac{c_s}{n_s(n-n_s)}$ measures the fraction of all possible edges leaving the clustering.
Normalized cut (Shi and Malik, 2000): $NC = \sum_{s=1}^{k} \frac{c_s}{2m_s + c_s} + \frac{c_s}{2(m-m_s)+c_s}$ measures the fraction of total edge connections to all the nodes in the graph.
Internal density (Radicchi *et al.*, 2004): $ID = \sum_{s=1}^{k} 1 - \frac{m_s}{n_s(n_s-1)/2}$ measures the internal edge density of a clustering.
Community score (Pizzuti, 2008): $CS = \sum_{s=1}^{k} \left(\frac{2m_S}{n_S}\right)^2$ measures the edge density of each cluster with respect to its size.
Scaled cost function (King *et al.*, 2004): $SCF = \frac{n-1}{3} \sum_{v \in V} \frac{(c_s(v) + l_s(v))}{|N(v) \cup S_v|}$ measures the number of *bad connections* incident with v, i.e. one that exists between v and a node not belonging to the same cluster of v or one that does not exist between v and another node in the same cluster as v, scaled with respect to the size of the area v effects in the clustering.

3.3 Experimental results

The results discussed here refer to $\sigma_{OS} = 0.2$. Those obtained for larger values of σ_{OS} are illustrated in the Supplementary Material (as well as further precision-recall curves).

Figures 2–4 show the recall, precision and F-measure values for both the genetic approaches and the other methods (notice that the results of RanCoC on Yeast-D1 and Yeast-D1 are poor and the corresponding bars are not visualized, as the returned values are equal to zero).

The first observation is that the contestant methods obtain higher recall values with respect to the GA-PPI approaches for all the fitness functions, on all the considered datasets. Only GA-CS overcomes MCODE and Ahn, but it is defeated by the other

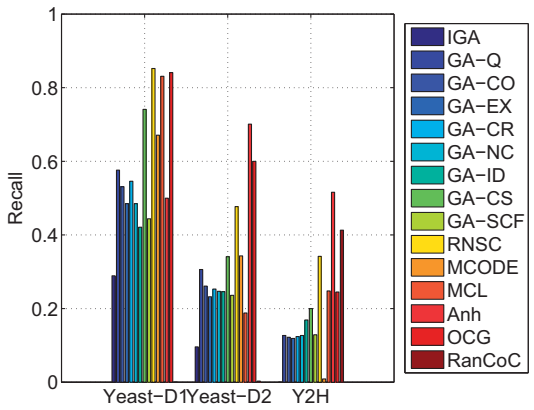

Fig. 2. Recall values for Yeast-D1, Yeast-D2 and Y2H ($\sigma_{OS} = 0.2$)

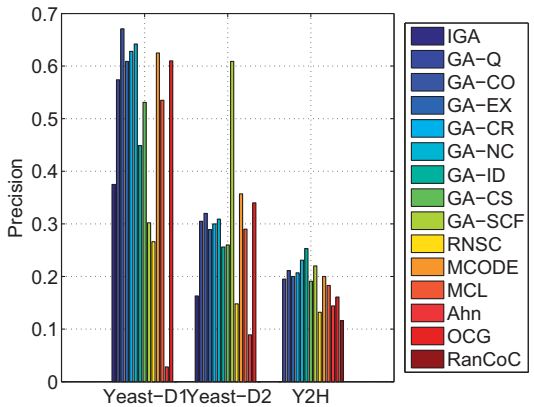

Fig. 3. Precision values for Yeast-D1, Yeast-D2 and Y2H ($\sigma_{OS} = 0.2$)

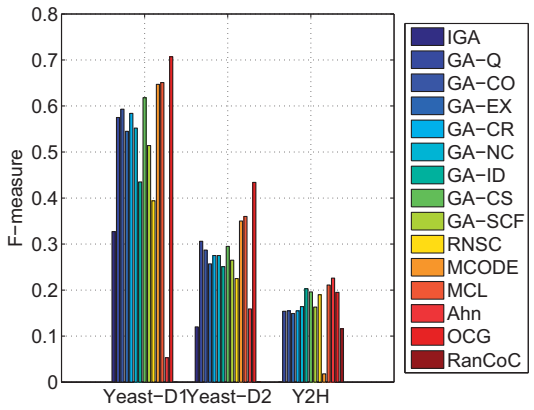

Fig. 4. F-measure values for Yeast-D1, Yeast-D2 and Y2H ($\sigma_{OS} = 0.2$)

methods. Higher recall means that a method predicts a higher number of complexes of all the true complexes. However, high values of precision indicate a more accurate prediction, as the predicted complexes are composed by a high percentage of proteins belonging to the true complex, thus the fraction of false positive is low. In this case, GA-CO, GA-CR and GA-NC are superior to all the other approaches on Yeast-D1, whereas all the

Table 2. Robustness analysis for GA-Q on Yeast-D1 ($\sigma_{OS} = 0.2$)

	Original network	Addition (+)/removal (−) of edges					Complete randomization
			20%	40%	60%	80%	
R	0.58	+	0.54	0.5	0.46	0.4	0.0
		−	0.6	0.59	0.59	0.58	
P	0.57	+	0.57	0.63	0.54	0.56	0.0
		−	0.5	0.46	0.41	0.4	
F_m	0.57	+	0.55	0.56	0.5	0.46	0.0
		−	0.55	0.52	0.49	0.47	

Note: The average values of 10 runs are shown.

GA-PPI approaches overcome RNSC, Ahn and RanCoC. IGA also performs better than RNSC and obtains higher precision values than those obtained by GA-PPI when the scaled cost function is used. Regarding Yeast-D2, the results are comparable, though RNSC, Ahn and RanCoC are the worst performing, whereas GA-SCF obtains the best value of precision. This result is due to the fact that GA-SCF finds a big cluster that involves many true complexes. On Y2H, the GA-PPI approaches present precision values always higher than all the other contestant methods, except for MCODE, which is, however, overcome by GA-Q, GA-CR, GA-NC and GA-CS. As regards F-measure, OCG obtains the bests results on Yeast-D1 and Yeast-D2, whereas on Y2H the values returned by GA-NC, GA-ID, MCL and Ahn are the highest.

For $\sigma_{OS} \geq 0.2$, MCL is the best performing on Yeast-D1. As regards Yeast-D2, the best methods are MCODE, Ahn and MCL. On Y2H, MCL obtains the best recall values, however, the values of precision and F-measure returned by MCODE are the highest (see further details in Section 2 of the Supplementary Material).

We tested the robustness of the GA-PPI approaches through negative controls, according to Brohée and van Helden (2006). We randomly generated new networks by both adding/removing an increasing percentage of edges from Yeast-D1. The random addition of edges simulates the introduction of noise caused by spurious inter-complex interactions. Removing edges means testing the performances of the approach when useful information is increasingly missed. Table 2 shows the results obtained for GA-Q for different addition/removal percentages (the results for the other functions and networks follow exactly the same behavior). In particular, the values of recall, precision and F-measure do not change significantly by adding or removing edges. Therefore, the algorithm is robust, as it is able to keep the original clustering (Brohée and van Helden, 2006). We also performed a complete randomization of Yeast-D1, by shuffling the edges between nodes so that each node preserves the same number of links as in the original graph. In this case, all the validation measures equal zero, showing that the algorithm does not return results when there is nothing to be found.

Finally, in Section 4 of the Supplementary Material, we show a further analysis we performed to assess the statistical significance of the clustering results through the hypergeometric test [e.g. Solava *et al*. (2012)]. In particular, the GA-PPI approaches are those presenting the best performances on Y2H, whereas RanCoC, Ahn and OCG are the best on the other two networks.

4 DISCUSSION

The goal of this review is 2-fold: (i) providing a compact overview of the main techniques presented in the literature for PPI networks clustering and (ii) presenting an experimental campaign to show the capability of GAs in extracting clusters from PPI networks, according to different topology-based fitness functions, and with respect to the main other approaches. Both aspects allow us to draw interesting considerations and conclusive remarks.

A first observation is that some of the methods discussed here (almost in the LD category) obtain modules one at a time by the selection of a seed node that is expanded until a condition, generally related to the cluster density, is satisfied. Thus, they can be considered bottom-up approaches: individual nodes are grouped together until all the graphs have been examined. Approaches that simultaneously find the clusters (e.g. PS ones) are instead top-down approaches. They consider the whole graph and try to partition it in connected components by cutting edges. Owing to the threshold constrains that many LD methods require to satisfy, to decide when a group of connected nodes is a cluster, nodes with few interactions are often discarded. The presence of these proteins would reduce the value of the function to optimize. The elimination of sparsely connected nodes has two main drawbacks. First, it prevents the possibility of obtaining topological shapes different from maximally dense subgraphs. Second, important information on the network structure could be lost. In Cho *et al*. (2006) and Hwang *et al*. (2006), the authors observe that bottom-up approaches discard a high percentage of nodes, though the returned clusters may have a more accurate *P*-value. On the other hand, top down approaches produce a lower number of unassigned proteins, yielding modules with larger size but with lower *P*-value. Results on the network coverage of PPI clustering techniques are provided in Pizzuti and Rombo (2012a), where it is shown that some of the LD approaches are able to reach a good compromise between the percentage of input network that is included in the output clustering and the overall accuracy of the clustering. Population-based methods follow a top-down approach, however, they rely on the evolution of solutions, guided by fitness maximization and their combination, to either connect or disconnect pairs nodes. They simultaneously extract the output clusters, thus guaranteeing a total coverage of the input network, as no protein is unassigned.

Another point is that proteins, generally, may participate in multiple biological processes. Thus, methods that assign a protein to only one group, such as the GAs tested here, hamper the possibility of proteins to be clustered in several groups, on the basis of the different functions they have in the cell. This limits their potentiality in describing the complexity of biological systems, as also proved by the experimental campaign provided here. The experimental results described in Section 3 show that the application of GAs to detect protein complexes in PPI networks is promising concerning the accuracy of the discovered groups, but it deserves further study to improve the number of

predicted true complexes. Generalization to allow overlapping clusters would be desirable to enhance the prediction capability of such approaches. An interesting direction to investigate would be that of combining LC and GAs, to make the latter ones able to detect overlapping clusters. To this aim, the link similarity measure by Solava *et al*. (2012) (see also Section 2.4) could be applied.

Note also that the performances of the considered approaches vary with respect to the different considered interaction datasets. Interestingly, OCG, that in Becker *et al*. (2012) has been shown to overcome Ahn on human datasets, is outperformed by the latter one on Y2H for both recall and F-measure. However, all the networks considered here refer to yeast PPI data, as *Saccharomyces cerevisiae* is one of the best characterized organisms. As proved by the experimental campaign provided in Pizzuti and Rombo (2012a), the performances of PPI network clustering techniques may become different when they are applied on less complete networks, and only a few of them are able to keep a good accuracy. Future work may include also analyzing how GA-PPI algorithms perform on the PPI networks of organisms such as human and fly, which are less characterized than yeast.

In conclusion, the investigation of population-based methods for PPI networks clustering is relatively recent and not yet enough explored, presenting interesting potentialities. Thus, additional research is necessary and desirable to improve the predictive power of these approaches.

ACKNOWLEDGEMENTS

The authors are grateful to Hamid Ravaee for providing IGA results and to Ryan W. Solava and Tijana Milenković for useful explanations about their approach.

Funding: 'MERIT: MEdical Research in Italy' financed by the Italian Ministry of Education, Universities and Research (MIUR) (in part to C.P.). Project 'Approcci composizionali per la caratterizzazione e il mining di dati omici' ('Compositional approaches for the characterization and mining of omics data') financed by the Italian Ministry of Education, Universities and Research (MIUR), and Progetto di Ateneo dell'Università degli Studi di Palermo 2012-ATE-0298 'Metodi Formali e Algoritmici per la Bioinformatica su Scala Genomica' (to S.E.R.).

Conflict of interest: none declared.

REFERENCES

Adamcsek,B. *et al*. (2006) CFinder: locating cliques and overlapping modules in biological networks. *Bioinformatics*, **22**, 1021–1023.

Ahn,Y.-Y. *et al*. (2010) Link communities reveal multiscale complexity in networks. *Nature*, **466**, 761–764.

Aittokallio,B. and Schwikowski,B. (2006) Graph-based methods for analyzing networks in cell biology. *Brief. Bioinform.*, **7**, 243–255.

Altaf-Ul-Amin,M. *et al*. (2006) Development and implementation of an algorithm for detection of protein complexes in large interaction networks. *BMC Bioinformatics*, **7**, 207.

Asburner,S. *et al*. (2000) Gene ontology: tool for the unification of biology. *the gene ontology consortium. Nat. Genet.*, **25**, 25–29.

Atias,N. and Sharan,R. (2012) Comparative analysis of protein networks: hard problems, practical solutions. *Commun. ACM*, **55**, 88–97.

Bader,G. and Hogue,H. (2003) An automated method for finding molecular complexes in large protein-protein interaction networks. *BMC Bioinformatics*, **4**, 2.

Becker,E. *et al*. (2012) Multifunctional proteins revealed by overlapping clustering in protein interaction network. *Bioinformatics*, **28**, 84–90.

Brohèe,S. and van Helden,J. (2006) Evaluation of clustering algorithms for protein-protein interaction networks. *BMC Bioinformatics*, **7**, 488.

Cho,Y.-R. *et al*. (2007) Semantic integration to identify overlapping functional modules in protein interaction networks. *BMC Bioinformatics*, **8**, 265.

Cho,Y.-R. *et al*. (2006) Identification of overlapping functional modules in protein interaction networks: Information flow-based approach. In: *Proceedings of the Sixth International Conference on Data Mining-Workshops*, 18-22 December 2006, Hong Kong.

Chua,H. *et al*. (2007) Using indirect protein-protein interactions for protein complex prediction. In: *Proceedings of Computational Systems Bioinformatics Conference (CSB07)*. pp. 97–109.

De Virgilio,R. and Rombo,S.E. (2012) Approximate matching over biological RDF graphs. In: *Proceedings of the ACM Symposium on Applied Computing*. pp. 1413–1414.

Derenyi,I. *et al*. (2005) Clique percolation in random networks. *Phys. Rev. Lett.*, **94**, 160–202.

Enright,A. *et al*. (2002) An efficient algorithm for large-scale detection of protein families. *Nucleic Acids Res.*, **30**, 1575–1584.

Evans,T. and Lambiotte,R. (2009) Line graphs, link partitions, and overlapping communities. *Phys. Rev. E*, **80**, 016105:1–016105:8.

Evans,T.S. and Lambiotte,R. (2010) Line graphs of weighted networks for overlapping communities. *Eur. Phys. J. B*, **77**, 265–272.

Farutin,V. *et al*. (2006) Edge-count probabilities for the identification of local protein communities and their organization. *Proteins*, **62**, 800–818.

Ferraro,N. *et al*. (2011) Asymmetric comparison and querying of biological networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **8**, 876–889.

Fortunato,S. (2010) Community detection in graphs. *Phys. Rep.*, **486**, 75–174.

Fortunato,S. and Barthélemy,M. (2007) Resolution limit in community detection. *Proc. Natl Acad. Sci. USA*, **104**, 36–41.

Gavin,A.C. *et al*. (2006) Proteome survey reveals modularity of the yeast cell machinery. *Nature*, **440**, 631–636.

Georgii,E. *et al*. (2009) Enumeration of condition-dependent dense modules in protein interaction networks. *Bioinformatics*, **25**, 933–940.

Girvan,M. and Newman,M.E.J. (2002) Community structure in social and biological networks. *Proc. Natl Acad. Sci. USA*, **99**, 7821–7826.

Goldberg,D.E. (1989) *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Pub.

Hartuv,E. *et al*. (2000) An algorithm for clustering cdna fingerprints. *Genomics*, **66**, 249–256.

Hartwell,L.H. *et al*. (1999) Clustering algorithm based graph connectivity. *Nature*, **402**, C47–C52.

Holland,J.H. (1975) *Adaptation in Natural and Artificial Systems*. Ann Harbor MI, University of Michigan Press.

Hwang,W. *et al*. (2006) A novel functional module detection algorithm for protein-protein interaction networks. *Algorithms Mol. Biol.*, **1**, 24.

Jain,A.K. and Dubes,R.C. (1988) *Algorithms for Clustering Data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Jancura,P. *et al*. (2011) A methodology for detecting the orthology signal in a PPI network at a functional complex level. *BMC Bioinformatics*, **13** (Suppl. 10), S18.

King,A.D. *et al*. (2004) Protein complex prediction via cost-based clustering. *Bioinformatics*, **20**, 3013–3020.

Kovacs,A.I. *et al*. (2010) Community landscapes: an integrative approach to determine overlapping network module hierarchy, identify key nodes and predict network dynamics. *PLoS One*, **5**, e12528.

Kuchaiev,O. *et al*. (2011) Graphcruch 2: Software tool for network modeling, alignment and clustering. *BMC Bioinformatics*, **12**, 24.

Leskovec,J. *et al*. (2010) Empirical comparison of algorithms for network community detection. In: *Proccedings of International World Wide Web Conference (WWW)*. pp. 631–640.

Li,M. *et al*. (2008) Modifying the DPClus algorithm for identifying protein complexes based on new topological structures. *BMC Bioinformatics*, **9**, 398.

Li,X.-L. *et al*. (2010) Computational approaches for detecting protein complexes from protein interaction network: a survey. *BMC Genomics*, **11** (Suppl. 1), S3.

Li,X.-L. *et al*. (2005) Interaction graph mining for protein complexes using local clique merging. *Genome Inform.*, **16**, 260.

Li,X.-L. *et al.* (2007) Discovering protein complexes in dense reliable neighborhoods of protein interaction networks. In: *Proccedings of Computer System Bioinformatics Conference. (CSB07)*. pp. 157–168.

Lin,C. *et al.* (2006) Clustering methods in protein-protein interaction network. In: Xiaohua,H. and Pan,Y. (eds) *Knowledge Discovery in Bioinformatics: Techniques, Methods and Application*. John Wiley & Sons, Inc.

Liu,H. and Liu,J. (2006) Clustering protein interaction data through chaotic genetic algorithm. In: T.-D.,Wang *et al.* (eds) *Proceedings 6th International Conference, SEAL 2006, Hefei, China, October 15-18, 2006*. LNCS Vol. 4247, Springer, pp. 858–864.

Lovasz,L. (1996) Random walks on graphs: a survey. In: Miklós,D., Sós,V.T. and Szõnyi,T. (eds) *Combinatorics, Paul Erdos is Eighty*. Vol. 2, János Bolyai Mathematical Society, pp. 353–398.

Lubovac,Z. *et al.* (2006) Combining functional and topological properties to identify core modules in protein interaction networks. *Proteins*, **64**, 948–959.

Macropol,K. *et al.* (2009) RRW: repeated random walks on genome-scale protein networks for local cluster discovery. *BMC Bioinformatics*, **10**, 283.

Madeira,S.C. and Oliveira,A.L. (2004) Biclustering algorithms for biological data analysis: a survey. *IEEE Trans. Comp. Biol. Bioinf.*, **1**, 24–45.

Mewes,H.W. *et al.* (2000) MIPS: a database for genomes and protein sequences. *Nucleic Acids Res.*, **28**, 37–40.

Mewes,H.W. *et al.* (2006) MIPS: analysis and annotation of proteins from whole genomes in 2005. *Nucleic Acids Res.*, **34**, 169–172.

Milenkovic,T. and Pržulj,N. (2008) Uncovering biological network function via graphlet degree signatures. *Cancer Inform.*, **6**, 257–273.

Newman,M.E.J. and Girvan,M. (2004) Finding and evaluating community structure in networks. *Phys. Rev.*, **E69**, 026113.

Pxalla,G. *et al.* (2005) Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, **435**, 814–818.

Panni,S. and Rombo,S.E. (2013) Searching for repetitions in biological networks: methods, resources and tools. *Brief. Bioinform.* Published online December 3, 2013 doi:10.1093/bib/bbt084.

Park,Y. and Song,M. (1989) A genetic algorithm for clustering problems. In: *Proceeding of 3rd Annual Conference on Genetic Algorithms*. pp. 2–9.

Pavlopoulos,G.A. *et al.* (2011) Using graph theory to analyze biological networks. *BioData Min.*, **4**, 10.

Pereira,J.B. *et al.* (2004) Detection of functional modules from protein interaction networks. *Proteins*, **54**, 49–57.

Pizzuti,C. (2008) GA-NET: a genetic algorithm for community detection in social networks. In: *Proccedings of the 10th International Conference on Parallel Problem Solving from Nature*. pp. 1081–1090.

Pizzuti,C. (2009) Overlapped community detection in complex networks. In: *Proceedings of the 11th Annual conference on Genetic and Evolutionary computation*, GECCO'09. pp. 859–866.

Pizzuti,C. and Rombo,S.E. (2007) Pincoc: a co-clustering based approach to analyze protein-protein interaction networks. In: *Procedings of the 8th International Conference on Intelligent Data Engineering and Automated Learning*. pp. 821–830.

Pizzuti,C. and Rombo,S.E. (2008) Multi-functional protein clustering in ppi networks. In: *Proccedings of the 2nd International Conference on Bioinformatics Research and Development (BIRD)*. pp. 318–330.

Pizzuti,C. and Rombo,S.E. (2012a) A coclustering approach for mining large protein-protein interaction networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.*, **9**, 717–730.

Pizzuti,C. and Rombo,S.E. (2012b) Experimental evaluation of topological-based fitness functions to detect complexes in ppi networks. In: *Genetic and Evolutionary Computation Conference (GECCO)*. pp. 193–200.

Pizzuti,C. and Rombo,S.E. (2013) Restricted neighborhood search clustering revisited: an evolutionary computation perspective. In: *Proccedings of the 8th IAPR International Conference on Pattern Recognition in Bioinformatics (PRIB)*. pp. 59–68.

Pizzuti,C. *et al.* (2012) Complex detection in protein-protein interaction networks: A compact overview for researchers and practitioners. In: *10th European Conference of Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics (EvoBio)*. pp. 211–223.

Pržulj,N. (2005) Functional topology in a network of protein interactions. In: Jurisica,I. and Wigle,D. (eds) *Knowledge Discovery in Proteomics*. CRC Press Taylor & Francis Group, Boca Raton, FL, USA.

Pržulj,N. (2007) Biological network comparison using graphlet degree distribution. *Bioinformatics*, **23**, 177–183.

Radicchi,F. *et al.* (2004) Defining and identifying communities in networks. *Proc. Natl Acad. Sci. USA*, **101**, 2658–2663.

Ravaee,H. *et al.* (2010) Improved immune genetic algorithm for clustering protein-protein interaction network. In: *Proccedings of the 2010 IEEE International Conference on Bioinformatics and Bioengineering*. pp. 174–179.

Ruan,J. and Zhang,W. (2008) Identifying network communities with a high resolution. *Phys. Rev. E*, **77**, 016104.

Samantha,M. and Liang,S. (2003) Predicting protein functions from redundancies in large-scale protein interaction networks. *Proc. Natl Acad. Sci. USA*, **100**, 12579–12583.

Schaeffer,S.E. (2007) Survey: graph clustering. *Comput. Sci. Rev.*, **1**, 27–64.

Shannon,P. *et al.* (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.*, **13**, 2498–504.

Sharan,R. *et al.* (2007) Network-based prediction of protein function. *Mol. Syst. Biol.*, **3**, 88.

Shi,J. and Malik,J. (2000) Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, **22**, 888–905.

Solava,R.W. *et al.* (2012) Graphlet-based edge clustering reveals pathogen-interacting proteins. *Bioinformatics*, **28**, 480–486.

Tornw,S. and Mewes,H.W. (2003) Functional modules by relating protein interaction networks and gene expression. *Nucleic Acids Res.*, **31**, 6283–6289.

Van Dongen,S. (2008) Graph clustering via a discrete uncoupling process. *SIAM J. Math. Anal. Appl.*, **30**, 121–141.

von Mering,D. *et al.* (2002) Comparative assessment of a large-scale data sets of protein-protein interactions. *Nature*, **31**, 399–403.

Wang,J. *et al.* (2010) Recent advances in clustering methods for protein interaction networks. *BMC Genomics*, **11 (Suppl. 3)**, S10.

Yu,H. *et al.* (2008) High-quality binary protein interaction map of the yeast interactome network. *Science*, **322**, 104–110.

Zaki,N. *et al.* (2012) Prorank: a method for detecting protein complexes. In: *Proccedings of the Genetic and Evolutionary Computation Conference*. pp. 209–216.