PAPER • OPEN ACCESS

Defining and identifying cograph communities in complex networks

New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics

Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

OPEN ACCESS

RECEIVED
14 July 2014

ACCEPTED FOR PUBLICATION
11 December 2014

PUBLISHED
20 January 2015

PAPER

Defining and identifying cograph communities in complex networks

Songwei Jia[1], **Lin Gao**[1], **Yong Gao**[2], **James Nastos**[2], **Yijie Wang**[3], **Xindong Zhang**[1] **and Haiyang Wang**[1]

[1] School of Computer Science and Technology, Xidian University, Xi'an 710071, People's Republic of China
[2] Department of Computer Science, University of British Columbia Okanagan, Kelowna, British Columbia, V1V 1V7 Canada
[3] Department of Electrical & Computer Engineering, Texas A&M University, College Station, TX 77843, USA

E-mail: lgao@mail.xidian.edu.cn

Keywords: complex networks, community dection, centrality

Abstract

Community or module detection is a fundamental problem in complex networks. Most of the traditional algorithms available focus only on vertices in a subgraph that are densely connected among themselves while being loosely connected to the vertices outside the subgraph, ignoring the topological structure of the community. However, in most cases one needs to make further analysis on the interior topological structure of communities to obtain various meaningful subgroups. We thus propose a novel community referred to as a *cograph community*, which has a well-understood structure. The well-understood structure of cographs and their corresponding cotree representation allows for an immediate identification of structurally-equivalent subgroups. We develop an algorithm called the *Edge P_4 centrality*-based divisive algorithm (EPCA) to detect these cograph communities; this algorithm is efficient, free of parameters and independent of additional measures mainly due to the novel local edge P_4 centrality measure. Further, we compare the EPCA with algorithms from the existing literature on synthetic, social and biological networks to show it has superior or competitive performance in accuracy. In addition to the computational advantages over other community-detection algorithms, the EPCA provides a simple means of discovering both dense and sparse subgroups based on structural equivalence or homogeneous roles which may otherwise go undetected by other algorithms which rely on edge density measures for finding subgroups.

1. Introduction

As one hotspot and keystone of the research on complex networks, community or module detection has been heavily developed in the past few decades [1]. While a range of algorithms have been proposed to focus mainly on how to detect a cohesive group of vertices as a rough community, they primarily use the macroscopic property of communities, since they are internally edge-dense while being sparse outside and pay little attention to the interior topological structure. The fact that these traditional algorithms do not reveal a specific structure in their detected communities means that extra work will have to be done in order to identify the important subgroups or modules within the community. In applications of complex networks, one often needs to investigate the next-level structure of sub-communities or modules. For example, while protein complexes (modeled as modules) detected in protein–protein interaction (PPI) networks can help us understand biological networks, they still cannot provide enough information due to the fact that we also want to obtain the core components of the complexes [2] or to identify the essential proteins [3]. Additionally, for communities detected on practical networks we also want to know not only which vertices are grouped together from a network partition but also the relationships among the individual members of the obtained communities such as the hierarchical organization of actors in a social network [4]. Traditional algorithms cannot meet such requirements without extra tools from network analysis.

While the main approach to community detection has been to find the resulting network clusters via partitive algorithms, there has been some work done in attempting to characterize the topological structure of the community, which leads to an alternate algorithmic approach of attempting to find these special structures.

This approach has been used to attempt to detect structures such as the clique [5], quasi-clique [6, 7], n-club, n-clan, k-plex, etc [1] as the expected community or module structure in complex networks or to characterize the topological structures of communities based on statistical methods [8]. These algorithms can obtain specific graceful topological structures, but they suffer from prohibitive computational complexity due to the inherent combinatorial complexity of the prime graphs on large-scale practical complex networks. The familial groups in social networks proposed by Nastos and Gao [4] and their corresponding comparability tree arrangements of the groups are one example in which the structural definition of a community reveals much interior structure in the communities, but they also show that the computational problem of detecting these groups is NP-complete. It does, however, open new strategies for defining communities or modules by structural analyses.

From the viewpoint of structural analyses, we consider not only the traditional macroscopic clustering property of communities being internally dense while being externally sparse but also the topological structure of the communities found. We propose a polynomial-time approach of network partitioning, called the EPCA, which detects connected cograph communities in a network. A graph (network) is a cograph when it excludes a specific subgraph configuration called a P_4, defined in the next section. Cographs have attracted persistent attention lately [9–13]. Our algorithm uses an edge-centrality measure called P_4 centrality, defined in section 3, and the resulting cograph communities reveal superior or competitive accuracy in community detection when compared to communities obtained by the state-of-the-art algorithms, as shown in the experiments in section 4.1. Most importantly, cographs have a unique cotree representation, which is efficiently constructed (an example is displayed in appendix A); this allows us to analyze the topological structure of our communities. By this nontraditional structural analysis, we can obtain various meaningful subgroups within cograph communities which the traditional algorithms cannot detect since the sub-modules may be sparsely connected.

This paper is organized as follows: section 2 introduces several terminologies used in the latter part of this paper. Section 3 first presents the EPCA based on the novel edge P_4 centrality and then demonstrates the properties of cograph communities. The accuracy analysis and nontraditional structural analysis of cograph communities using their corresponding cotrees are given in section 4. The conclusions and discussion are presented in section 5.

2. Terminologies and definitions

The terminology used in this paper is compatible with [9]. A network will be equivalently referred to as a *graph*. The nodes of a network can be referred to as *vertices*. A connection joining two nodes u and v is an *edge*, written as uv or (u, v). If a set of objects V are nodes in a network, and the edges joining these nodes are E, we refer to this network as $G = (V, E)$. We define several relevant terms here:

2.1. Induced subgraphs and the P_4

An *induced subgraph* of a network is specified by a set of vertices, and all of the edges that exist on those vertices in the network are also part of the induced subgraph. More formally, for a network $G = (V, E)$, a subnetwork $H = (V', E')$ is an induced subgraph of G if $V' \subseteq V, E' \subseteq E$, where for every pair u and v of V', uv is in E' only if uv is in E. A P_4 is an induced graph on four ordered vertices, which are connected as a simple path [9]. That is, it contains three consecutive edges and, just as importantly, there are no additional edges within these four vertices. An example of a P_4 $a - b - c - d$ is shown in figure 1(a), and these four vertices would not be a P_4 in a network if the network contained an edge joining a and c, for example.

2.2. Cographs

A graph is called a cograph (also known as a P_4 *restricted* graph), if it does not contain a P_4 as an induced subgraph [9]. A single vertex is a trivial cograph, as is any network with three or fewer vertices. An example of a cograph is shown in figure 1(b), and we reiterate that while vertices b, d, c and w form a path, they do not *induce* a path since those four vertices also contain edges bw and dw.

2.3. Cograph community

Cograph communities are defined as the connected components of a network that has no P_4 subgraph. As will be seen in the following section, algorithm EPCA will delete edges that have high P_4-centrality until our modified network is a cograph. The resulting connected components will define the cograph communities.

2.4. Cotree

The rooted tree representing the parse structure of a cograph in normalized form is referred to as a cotree. The leaves of a cotree are the vertices of the corresponding cograph, and each internal tree vertex represents the

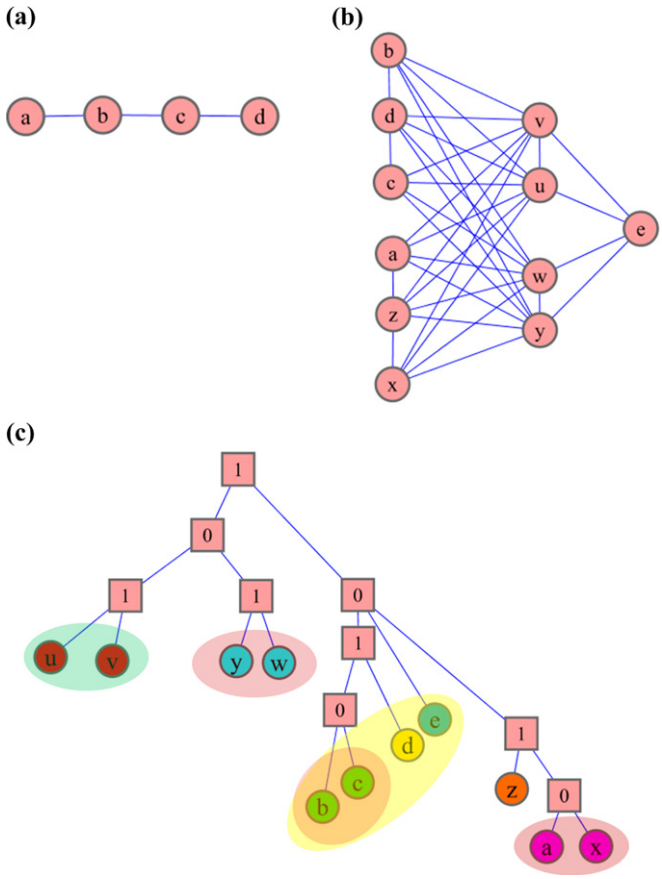

Figure 1. Illustration of the terminologies; (a) a P_4 consists of four vertices a, b, c and d; (b) a cograph consists of 11 vertices and 34 edges; (c) a cotree corresponding to the cograph shown in (b).

union or joint operation. In order to establish various properties about cographs we label each internal vertex of a cotree as follows: the root is labelled **1**, the children of a vertex with label **1** are labelled **0** and the children of a vertex labelled **0** are labelled **1** [14, 15]. Figure 1(c) illustrates the cotree for the cograph depicted in figure 1(b). The set of cographs is exactly the set of graphs which can be represented as a cotree, and every cograph has a unique cotree representation.

2.5. Siblings

For a given vertex x in the complex network $G(V, E)$, the neighbourhood of x denoted by $N(x)$ is $\{y \in V \,|(x, y) \in E\}$. Vertices x, y are called siblings if $N(x) - \{x, y\} = N(y) - \{x, y\}$. The siblings are called *strong* if the vertices are adjacent and are called *weak* otherwise [9]. For example, as shown in figure 1(c), vertices v and u are strong siblings, while vertices b and c are weak siblings. Strong and weak siblings have also been called *true twins* and *false twins* in other contexts. Cographs can also be characterized as graphs which can be generated by repeatedly adding strong and weak siblings to a single vertex.

3. The approach EPCA and cograph communities

To detect the cograph communities of a complex network efficiently, we give an algorithm called the EPCA, a typical divisive algorithm based on edge P_4 centrality. In the following, we first introduce edge P_4 centrality and the approach EPCA; then, we demonstrate the properties of cograph communities.

3.1. EPCA based on P_4 centrality

3.1.1. P_4 centrality

The set of edges that link the vertices of the same community (also called intra-links) are generally expected to be denser than the set of edges that link different communities (also called inter-links). That is, the inter-links are relatively sparser than the intra-links. Intuitively, there are many more cycles embedded in intra-links, while one does not expect to find many cycles using inter-links. This means that the inter-links tend to belong to more

Table 1. A table with the details of the networks[a], the complex golden standards and the high-level GO terms.

Network	Vertices	Edges	Communities	MIPS	SGD	PCDq	CORUM	$\|GO\|$[b]
SNs	1000	—	—	—	—	—	—	—
ZKCN	34	78	2	—	—	—	—	—
PBN	105	441	3	—	—	—	—	—
BDN	62	159	2	—	—	—	—	—
FN	115	613	12	—	—	—	—	—
*Sce*DIP	4980	22 076	—	203	305	—	—	1050
*Hsa*HPRD	9269	36 917	—	—	—	1204	1294	4457

[a] The networks are the largest components of the original datasets.
[b] $|GO|$ is the number of the GO terms that have *IC* of more than 2.

paths. Since a P_4 is a very simple induced path there are no small cycles among those four vertices of a P_4. Thus, the inter-links tend to belong to more P_4s since they tend to be part of paths, while the intra-links are inclined to compose fewer P_4 since they tend to belong to more cycles. From these facts, we define the edge P_4 centrality, which is a score assigned to edges which counts the number of P_4s to which that given edge belongs. This definition of edge P_4 centrality gives us a way to quantitatively measure the fact that an edge ij is more inter-link-like than intra-link. If its P_4 centrality is large it is more likely to be an inter-link, while if its P_4 centrality is smaller it is more likely to be an intra-link.

Formally, the edge P_4 centrality of an edge ij is defined as the number of pairs of vertices $\{x, y\}$ for which the set $\{i, j, x, y\}$ induces a P_4. Note that these four vertices can extend edge ij to a P_4 in a number of ways: $x-i-j-y, y-i-j-x, i-j-x-y, i-j-y-x, j-i-x-y, j-i-y-x$ and can extend all of their reversals. If any of these configurations occur, this 4-set $\{i, j, x, y\}$ contributes a score of 1 to the P_4 centrality of the edge ij and to the two other edges on these four vertices.

One can check if four vertices induce a P_4 if the induced subgraph on these four vertices contains two vertices of degree 1 and two vertices of degree 2. So, one could write a function $IsP_4(a, b, c, d)$ easily (but we omit the details as this highly depends on the data structures one uses to store and access the elements of their graph). Using such a function, a simple algorithm to compute the P_4 centrality of all of the edges would be to enumerate all sets of 4-distinct vertices and test $IsP_4(a, b, c, d)$ and, if it is true, increment the centrality score for the three involved edges. Of course, there are a number of improvements that can be added to this process, for example, shortcutting the inner loops when the first three vertices induce degrees of 0, 0, 0 or 2, 2, 2, as these configurations cannot extend to a P_4. One can also limit the search for the next candidate vertex by only choosing from the neighbourhood set of the appropriate vertices already in consideration. Of course, obtaining the neighbourhood of a vertex is, once again, dependent on the graph data structure used; so, we do not discuss the finer details of this process here.

We note that updating the P_4 centrality of all of the edges upon an edge deletion is much faster than the initial calculation of all of the centrality scores, since after removing edge ab, we must only search over pairs of candidate vertices $\{c, d\}$ to find the affected changes rather than recalculate a search over all of the quadruplets.

To understand the edge P_4 centrality better, we compare it with edge anti-triangle centrality [16], the edge clustering coefficient [17] and edge betweenness [18, 19], respectively. We choose edge anti-triangle centrality for comparison because it has a similar definition, the edge clustering coefficient as it is a typical and efficient local centrality and edge betweenness for its well-known accuracy in identifying an edge as being inside or outside a community. We plot the scatters of the logarithm of edge P_4 centrality and edge anti-triangle centrality, the edge clustering coefficient and the logarithm of edge betweenness, respectively, on the Zachary karate club network (ZKCN) [20], the LFR synthetic network [21] with the mixing parameter $mu = 0.5$ and the *S.cerevisiae* PPI networks (*Sce*DIP) [22] obtained from the DIP. The details of these three network comparisons are demonstrated in table 1. We compare these four centralities for the ability of discriminating inter-links from intra-links on the ZKCN, the LFR synthetic network ($mu = 0.5$) and the *Sce*DIP, successively. In particular, we compare them for two important quantities: the first one is the fraction of vertices contained in the giant component, denoted by RGC [23]. A sudden decline of RGC is observed if the network disintegrates after the deletion of a certain fraction of edges. The second quantity is the so-called normalized susceptibility [23], defined as

$$\tilde{S} = \sum_{s<s_{\max}} \frac{n_s s^2}{N}, \tag{1}$$

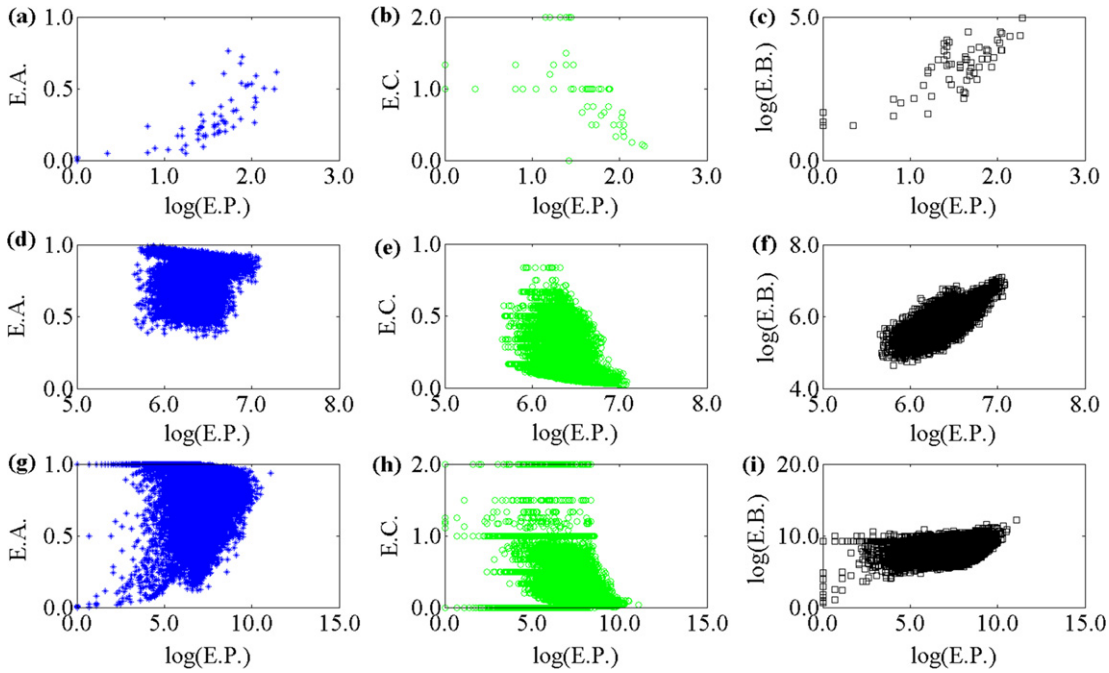

Figure 2. Scatter plots of the centralities for comparison. Figures 2(a), (d) and (g) are the scatter plots of edge anti-triangle centrality (E.A.) and the logarithm of edge P_4 centrality (log(E.P.)) on the ZKCN for 78 edges, the LFR synthetic network ($mu = 0.5$) for 7811 edges and the *Sce*DIP for 22076 edges, respectively; figures 2(b), (e) and (h) are the scatter plots of the edge clustering coefficient (E.C.) and also the log(E.P.) on the three networks, respectively; figures 2(c), (f) and (i) are the scatter plots of the logarithm of edge betweenness (log(E.B.)) and the log(E.P.), respectively.

where n_s is the number of components with size s, N is the size of the whole network and the sum runs over all of the components except the largest one. When $\tilde{S}$ is a function of the fraction of removed edges f an obvious peak can be observed that corresponds to the precise point at which the network disintegrates [23, 24].

In figures 2(a), (d) and (g), we plot the scatters of edge anti-triangle centrality and the logarithm of edge P_4 centrality on the ZKCN, the LFR synthetic network ($mu = 0.5$) and the *Sce*DIP, respectively. Figures 2(b), (e) and (h) show the scatters of the edge clustering coefficient and the logarithm of edge P_4 centrality, also on these three networks. Figures 2(c), (f) and (i) demonstrate the scatters of the logarithm of edge betweenness and logarithm of edge P_4 centrality. Here, we utilize the logarithm function for edge betweenness and edge P_4 centrality to obtain the same quantitative order with the edge anti-triangle centrality and edge clustering coefficient. As expected on all three classical networks, the edge P_4 centrality is positively correlated to the edge anti-triangle centrality and edge betweenness, while it is negatively correlated to the edge clustering coefficient, although these relations are not revealed very rigorously from visual inspection. In particular, notice that the edges with the highest edge P_4 centrality are neither always those with the highest edge anti-triangle centrality and edge betweenness nor those with the lowest edge clustering coefficient. By figure 2, the positive relations between the edge P_4 centrality and edge betweenness are revealed more distinctly than with the other two centrality measures.

Figures 3(a)–(c) compare the edge P_4 centrality, edge anti-triangle centrality, edge clustering coefficient and edge betweenness for the ability of discriminating inter-links from intra-links from the point of view of RGC on the ZKCN, the LFR synthetic network ($mu = 0.5$) and the *Sce*DIP, respectively, and correspondingly from figures 3(d)–(f) from the point view of normalized susceptibility $\tilde{S}$. Due to the high computational cost of edge betweenness on the *Sce*DIP we do not make comparisons to it in figures 3(c) and (f). As figure 3 shows, on the ZKCN, edge P_4 centrality can gain better performance from the point of view of RGC and $\tilde{S}$, while on the synthetic network and on the *Sce*DIP it can be slightly poorer than the edge anti-triangle centrality and edge betweenness, and it is better than the edge clustering coefficient on all three networks. Although edge P_4 centrality is slightly poorer than edge anti-triangle centrality and edge betweenness from the point of view of RGC and $\tilde{S}$ on synthetic networks and on the *Sce*DIP, the results of the communities obtained by the EPCA based on the edge P_4 centrality are much better than the algorithms based on edge anti-triangle centrality and edge betweenness. This is explained by the fact that we observed that the edges of the highest edge P_4 centrality do not correspond to the edges of the highest edge anti-triangle or edge betweenness; so, the respective algorithms, which delete these edges of high centrality, will make different deletion choices early in their execution.

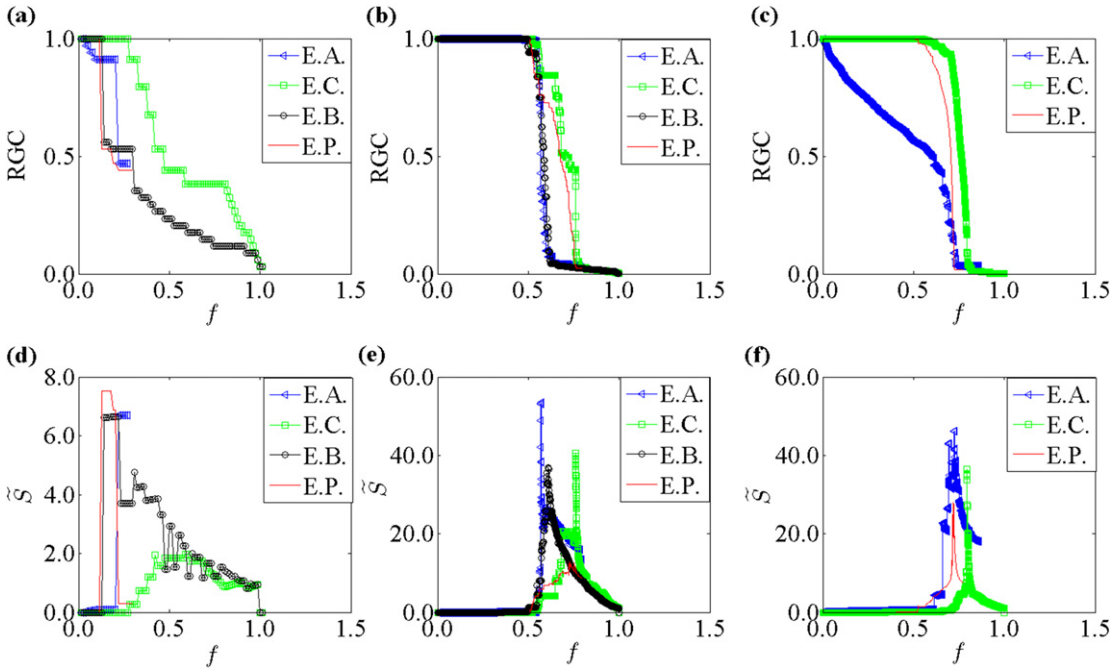

Figure 3. Comparison for centralities on the ability of discriminating inter-links from intra-links. A comparison of the edge P_4 centrality (E.P.), the edge anti-triangle centrality (E.A.), the edge clustering coefficient (E.C.) and the edge betweenness (E.B.) pairwise. Figures 3(a)–(c) are from the point of view of RGC on the ZKCN, the LFR synthetic network ($mu = 0.5$) and the *Sce*DIP, respectively; figures 3(d)–(f) are also from the point of view of normalized susceptibility $\tilde{S}$ on the three networks, respectively, whereas in figures 3(c) and (f) no comparison is performed for edge betweenness due to its high computational cost on the *Sce*DIP.

Comparing edge P_4 centrality with edge anti-triangle centrality, edge clustering coefficient and edge betweenness by plotting scatters and from the point of view of RGC and $\tilde{S}$, we can summarize that edge P_4 centrality can be appropriate for community detection and obtain a significant competitive advantage over other processes that remove edges.

3.1.2. EPCA for cograph community detection

We assume that the network $G(V, E)$ is connected, undirected and unweighted. The EPCA repeatedly removes the edge with the highest edge P_4 centrality score until the scores of the remaining edges are all zero. The EPCA is described in detail as follows:

Input: $G(V, E)$
Output: cograph communities
Calculate the P_4 centrality score for each available edge
While the highest score $\neq 0$ ***do***
Remove the edge with the highest score
Recalculate the scores of those edges affected by the removal
End

The EPCA is a typical divisive algorithm for community detection, while it possesses two significant differences to the general divisive algorithm. First, the EPCA does not need to remove the edges one by one until there is no edge left in the complex network. It just removes part of the whole edge set until the P_4 centrality of the remaining edges are all zero; this is sometimes only a small portion of the edge set. This makes it more computationally efficient and free of any parameters. Second, unlike the general divisive algorithms, which depend on additional measures to decide the community structure, the EPCA does not depend on any additional measures, and it outputs the current components as the expected cograph communities. The remaining components are cographs since they do not contain a P_4 and they possess additional algorithmic and structural properties.

The P_4 centrality is a local centrality measure. The complexity of the EPCA is the same as that of EACH [16], and the total space complexity of the EPCA is $O(|E|)$. The computational time-complexity is $O\left(\bar{k}^2|E| + \bar{k}^4 T\right)$, where $|E|$ is the number of edges, $\bar{k}$ is the average degree of the networks and T is the maximum number of iterations. Here, we want to emphasize that T is not a real parameter of the approach EPCA and does not need to be fixed *a priori*. The condition of the highest P_4 centrality score of the available edges equaling zero is the only

condition used for ending the loop. The value T is just used to represent the maximum number of iterations to express the complexity of the EPCA for convenience. The space and time complexities of other state-of-the-art algorithms are listed in [16], and the complexity of the EPCA is much lower than the others and is the same as that of the algorithm EACH.

3.2. Properties of a cograph community

A cograph community is a connected cograph, which is a special structure that has graceful topological properties evidenced by its unique cotree. The sub-communities or modules of a cograph community are not just a group of tightly-cohesive vertices, like in the case of traditional communities, they are also sparsely-connected subgroups of vertices which are structurally identical. From the definitions of cographs and P_4, the diameter of a cograph community is at most a two-hop since there is no induced P_4, and as a whole community it reveals more intensive social roles or biological functions than those obtained by other traditional algorithms, as demonstrated in section 4.1. Cographs possess a range of algorithmic and structural properties, for example, they can solve the (otherwise NP-hard) problems such as coloring, clique detection, hamiltonicity, etc, which can be done in polynomial time problems on cographs [14, 15].

Here, however, we are especially interested in the property that we can construct, which is its corresponding unique cotree representation, and we are interested in extracting information on subgroups from it. Almost all of the properties of cographs are revealed by the corresponding cotree; so, constructing the cotree is a prerequisite to making a comprehensive and deep study on cographs. The vertices of a cograph community, organized by the cotree, can reveal more lucidly the interior structure and provide a convenient framework for making a next-level analysis. One can construct a cotree in linear time (that is, in time proportional to the time required to simply read the graph) by the algorithms given in the papers [14, 15] and note that the cotree for a particular cograph is unique up to a permutation of the children of the internal vertices.

The cotree possesses the advantage of revealing various subgroups of the vertices of the corresponding community. The vertices belonging to the same subgroup are characterized as those having the same adjacency behavior to the remaining nodes in the community. That is, two vertices are in the same subgroup if their neighbours outside of the subgroup are equivalent. The trivial subgroups are the whole community and each of the isolated vertices. In addition to these trivial ones, the most basic subgroups are the strong siblings or weak siblings. In a general subgroup, if vertex u of this subgroup is adjacent to some vertex v which is not in the subgroup, all of the vertices of this subgroup are adjacent to v; this means the vertices of this subgroup possess the same connecting pattern. Similarly, if vertex u (in the subgroup) is not adjacent to vertex v (outside of the subgroup), none of the vertices in this subgroup are adjacent to v; that means they possess the same disconnecting pattern. These subgroups have been called *homogeneous sets*, *modules* or *indistinguishable sets* due to the fact that all of the vertices in the subgraph interact with the rest of the vertices (outside of the subgraph) in identical ways.

As shown clearly in figure 1(c), the vertices v and u are strong siblings in the pale-green region, and the vertices w and y form another pair of strong siblings in the hazel region, while the vertices b and c are weak siblings in the light-pink region, and vertices a and x are weak siblings. The larger subgroup of b, c, d and e is determined by the outer topological environment by being completely connected to the subgroup of v, u, w and y while being disconnected to the subgroup of z, a and x.

Finding various meaningful subgroups of cograph communities according to their neighbours and non-neighbours indeed brings new angles to investigating the relationships among the members of the communities. What we want to emphasize is that these meaningful subgroups within cograph communities cannot be detected by the traditional hierarchical algorithms, which focus only on obtaining hierarchical communities. The essential difference is that a set of vertices can form a structurally-equivalent subgroup even when there is no edge joining any two of them. Traditional community-detection algorithms that depend on identifying dense clusters will never associate such a group of nodes together.

The vertices revealing similar functions or roles in real networks are not always densely linked by edges but are sparsely linked by edges, as introduced by [25]. For these reasons, traditional hierarchical community-detection methods cannot always find the sparse groups very accurately. It is reasonable and novel to investigate the subgroups within cographs according to their outer topology.

The cotree has the natural advantage of demonstrating various subgroups according to their structural similarity, and we can identify these sets easily. Thus, analyzing cograph communities based on their cotrees is novel and very fascinating. Several examples of a cotree analysis of the cograph communities or modules obtained by the EPCA on the practical networks are made in detail in section 4.2.

4. Experiments and analyses

We produce the experiments and analyses in this section. In section 4.1 we compare the performance of the EPCA with algorithms from the existing literature on synthetic, social and biological networks. In section 4.2 we perform a next-level analysis of the cograph communities based their corresponding cotrees to obtain various meaningful subgroups.

4.1. Accuracy analyses

Before making comparisons, we first introduce why we select these state-of-the-art algorithms; how we implement them for performance comparison; where we obtain the synthetic, social and biological networks, the protein complex golden standard sets and the high-level GO term sets; and what criteria we use to evaluate the performance of the selected algorithms. After that, we compare all of the algorithms on synthetic, social and biological networks to show that the EPCA has comparative and superior performance.

Considering the EPCA as a typical non-overlapping community or module detection algorithm, the main compared algorithms here are non-overlapping identification algorithms. The state-of-the-art algorithms GN [18, 19], $ECCA_Q$ [16, 17], $ECCA_D$ [16, 17], EAC [16] and EACH [16] are selected, as these algorithms are edge-centrality-based. In particular, GN is based on edge betweenness centrality, which is a typical global centrality, and depends on the Q value [18] to decide the community structure. Both the $ECCA_Q$ and $ECCA_D$ are based on the edge clustering coefficient, which is a typical local centrality. The $ECCA_Q$ and $ECCA_D$ indicate the ECCA based on the additional measures of the Q and D values [26], respectively. The EAC and EACH are based on the anti-triangle centrality, whereas the EACH varies from the EAC by just including an added isolated vertex handling strategy. Neither of them depends on an additional measure while deciding the community structure. The NMF [27, 28] and SC [29, 30] possess matrix theory supports, while the CNM [31] attempts to optimize the additional measures to decide the community structure. The MCL [32] is based on random walks and is well known for its robustness. The INFOMAP [33] has significant accuracy performance, as reported in [34]. The OSLOM [35] is the only algorithm for detecting overlapping communities; here, we use OSLOM2, a much faster version from http://oslom.org/software.htm instead. The LOUVAIN [36] is very fast and widely used for community detection. We use the NodeXL (http://nodexl.codeplex.com/) implementations of the GN and CNM. The ECCA is implemented according to [17]; the NMF and SC are implementations in the R packages NMFN [37] and clusterSim [38], respectively. Lastly, we obtain the source code of the MCL (http://micans.org/mcl). The INFOMAP is implemented by the R package igraph [39], and we get the MATLAB version for the LOUVAIN from http://perso.uclouvain.be/vincent.blondel/research/louvain.html. All of the parameters are at default, as set in the corresponding tools or packages for the available algorithms.

For the sake of convenience, we first list several networks used in the experiments in table 1: the series of LFR synthetic networks (SNs) [21], the Zachary karate club network (ZKCN) [20], the Political books network (PBN) (http://www.orgnet.com/), the Bottlenose dolphins network (BDN) [40] and the Football network (FN) [19, 41]. The parameters of the LFR synthetic network are: average degree $\bar{k} = 15$, mixing parameter $mu = 0.5$, minimum community size $minc = 20$ and maximum community size $maxc = 50$. Here, we set $mu = 0.5$ since its median is 0.5. In fact, aside from mu, all of the other parameters are defaults from the original code (http://santo.fortunato.googlepages.com/inthepress2). Here, the *Sce*DIP represents the *S. cerevisiae* PPI networks obtained from the DIP [22], and *Hsa*HPRD represents the *H. sapiens* PPI networks extracted from HPRD [42]. We use the largest components of these two networks as the input of the algorithms. There are four protein complex golden standards: for the *Sce*DIP we use the Munich Information Center for Protein Sequences (MIPS) [43] and the Saccharomyces Genome Database (SGD) [44] golden standards, while for *Hsa*HPRD the golden standards are the Human Protein Complex Database with a Complex Quality Index (PCDq) [45] and the Comprehensive Resource of Mammalian Protein Complexes (CORUM) [46]. We remove the golden standard protein complexes, which consist of less than 2 proteins. The GO terms are not all of the terms but are instead the high-level GO terms, which have information content that is more than 2 [47]. The definition of the information content (IC) of a GO term g is $IC = -\log\left(\frac{|g|}{|root|}\right)$, as given in the literature [47], where '*root*' is the corresponding root GO terms (molecular function (MF), biological process (BP) or cellular component (CC)) of g. In addition, the GO terms that contain less than 2 proteins are removed. Lastly, we remove the protein complexes or GO terms of which no members appear in the corresponding PPI networks. The details of the *Sce*DIP and *Hsa*HPRD, the complex golden standards and the GO terms are also listed in table 1.

4.1.1. Synthetic networks and social networks

To quantify the accuracy performance of the compared algorithms on synthetic and social networks, we adopt the widely used *normalized mutual information* (NMI) [48, 49] to measure the similarities between the obtained communities and the real ones. The details of this issue are introduced in appendix B.

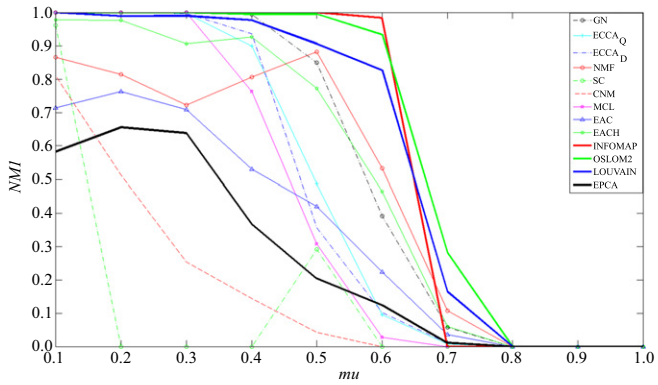

Figure 4. Illustration of the average NMI values obtained by the compared algorithms on the series of LFR networks as *mu* from 0.1 to 1.0 with a step of 0.1.

Table 2. Performance comparison with NMI on ZKCN, PBN, BDN and FN.

Algorithms	NMI_ZKCN	NMI_PBN	NMI_BDN	NMI_FN
GN	0.3084	0.4060	0.3534	0.6163
$ECCA_Q$	0.4456	0.0366	0.1000	0.8100
$ECCA_D$	0.2841	0.0373	0.1126	0.7794
NMF	**1.0000**	0.4204	**0.8006**	0.7058
SC	**1.0000**	0.07874	0.0013	0.7080
CNM	0.4778	0.4035	0.3609	0.4305
MCL	0.8333	0.3359	0.1140	0.8332
EAC	0.6270	0.1042	0.0872	0.5960
EACH	**1.0000**	0.4255	0.2441	0.8159
INFOMAP	0.5032	0.2798	0.2262	0.8332
OSLOM2	0.9167	**0.4547**	0.7026	0.8150
LOUVAIN	0.2290	0.1665	0.1963	**0.8361**
EPCA	0.5906	0.0857	0.1144	0.6197

We compare the NMI values of the results obtained by the compared algorithms on the synthetic networks, as shown in figure 4. Each node of the figure corresponds to the average NMI value of over 20 LFR networks constructed with the same parameters. The NMI values of all of the algorithms decrease as the mixing parameter *mu* increases. The reason for this is that the community structures of the LFR networks become increasingly fuzzier and thus are more difficult to be detected correctly as *mu* increases. As figure 4 shows, the black line represents the NMI value of the EPCA, and the results of the other algorithms are indicated by the corresponding color lines with signs. The INFOMAP can obtain the best performance among these compared algorithms, as reported in [34], and OSLOM2, LOUVAIN are not far behind. As figure 4 shows, the EPCA has superior performance compared with the algorithms SC and CNM across all of the networks, while as $mu \geqslant 0.6$, it is better than MCL, $ECCA_D$ and $ECCA_Q$.

Note that the algorithms INFOMAP, OSLOM2, LOUVAIN, GN, $ECCA_D$, $ECCA_Q$ need to set the number of output communities or must depend on the additional measures to decide the structure of the communities. As well as the global edge betweenness centrality GN based has a very high computational cost, while the inflation parameter of MCL affects the granularity of communities directly. By accounting for these factors, the EPCA can gain impressive performance in general on synthetic networks since it does not need additional measures, is free of parameters and is based on a typical local edge centrality at a very low computational cost.

Then, we compare the NMI values of the results obtained by the compared algorithms on the social networks, as shown in table 2. As depicted in table 2, the EPCA has comparative performance on the four social networks. Although the GN, $ECCA_D$, $ECCA_Q$, MCL, SC, NMF, INFOMAP and OSLOM2 algorithms obtain slightly better NMI values on some networks, they all depend on additional measures or a series of parameters; among them, SC and NMF also need to set the number of expected communities, which may bring in great difficulties if we do not have a prior knowledge of the networks. The NMI values of the EAC and EPCA reveal that the edge P_4 centrality and the edge anti-triangle centrality [16] have similar performances for community detection on small networks, while the EPCA may produce fewer isolated vertices on these social networks. The better performance of EACH is just due to its isolated vertex handling strategy. However, the communities

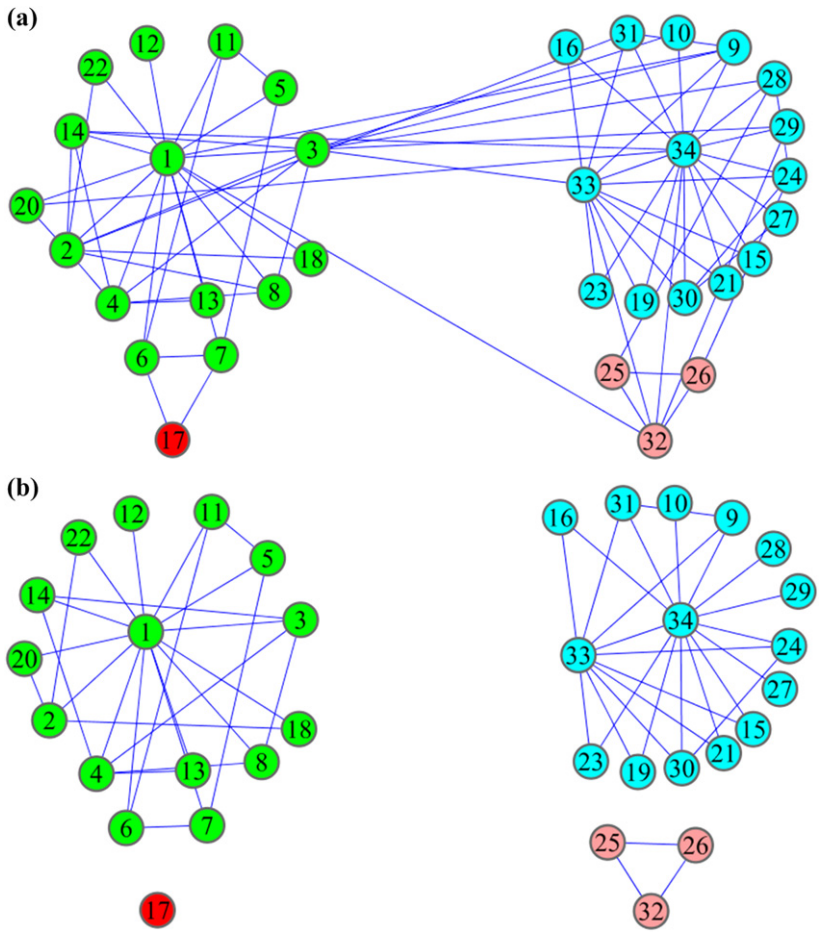

Figure 5. Illustration of the real groups and the cograph communities on the ZKCN. (a) the ZKCN consisting of 2 real communities; (b) the cograph communities obtained by EPCA after removing just 23 edges.

obtained by EACH are no longer cographs after the isolated vertex handler is used since the diameter of the communities obtained by EACH may be as high as a four-hop. So, the results obtained from EACH are not conducive to a cotree-based deeper analysis of those communities. Here, the EPCA is free of any parameters and has lower computational cost. Detecting communities by the EPCA is not only beneficial from the comparative performance on accuracy but also provides cograph communities on which we can perform a deeper analysis into the meaningful subgroups.

Figure 5 shows the cograph communities obtained by the EPCA on the ZKCN in detail. As figure 5(a) shows, the ZKCN consists of 34 vertices and 78 edges, representing 34 members and 78 social relationships among the members of the karate club. The club suffered a division which split the club into two, and the split very closely corresponds to a mini-cut that separates the two opposing individuals of the largest influence of the vertices 1 and 34. As shown in figure 5(b), the EPCA obtains 3 communities and one isolated vertex by just removing 23 edges. In essence, the 3 cograph communities obtained by the EPCA match the practical division of two communities comparatively well. Furthermore, the EPCA purifies the community headed by vertex 1 by removing the attachment vertex 17 since it does not directly connect with the leader vertex 1, as shown in figure 5(b). The EPCA partitions the community, led by vertex 34, by removing the sub-community, which consists of vertices 32, 25, 26, since vertices 25 and 26 do not directly connect with the leader vertex 34, as shown in figure 5(b).

Figure 6 depicts the cograph communities obtained by the EPCA on FN in detail. The FN consists of 115 vertices and 613 edges, representing 115 teams and 613 games played against each other, as shown in figure 6(a). The 115 teams are grouped into 11 conferences, with a 12th group of independent teams. The EPCA gains 13 communities and 10 isolated vertices after removing 291 edges. Surprisingly, we find the 13 cograph communities matching the 12 groups comparatively well in general. Here, we focus on the nontrivial cograph communities and ignore the isolated vertices. In fact, the two cograph communities in the shaded area of figure 6(b) mainly correspond to the group presented by the red triangle in figure 6(a). The group presented by the red triangles in figure 6(a) is realized as two smaller sub-communities because the group has the most

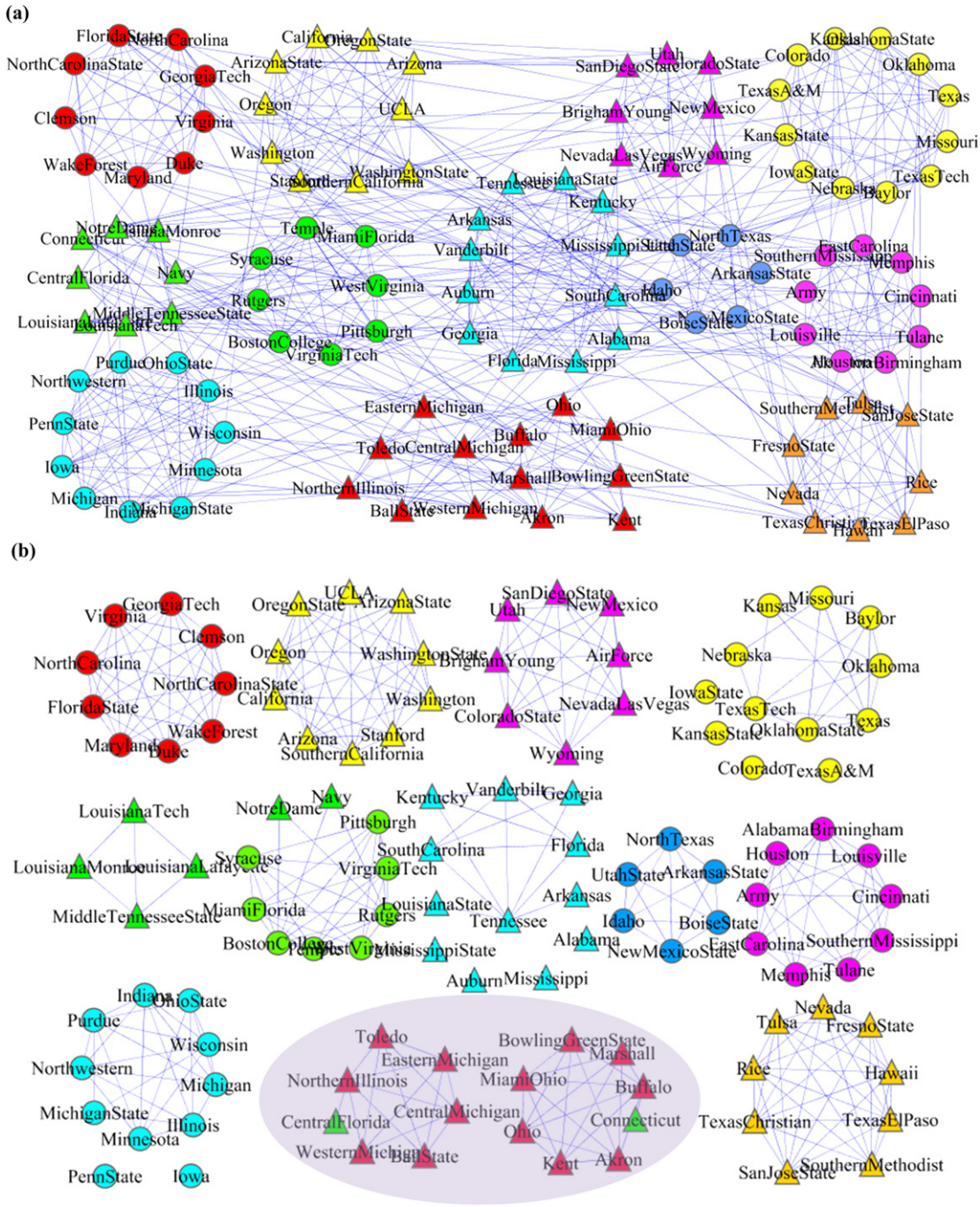

Figure 6. Illustration of the real groups and the cograph communities on FN. (a) The FN consisting of 12 groups; (b) the cograph communities obtained by the EPCA after removing 291 edges.

members, and the members tend to be led by the two leaders Kent and BallState, respectively. The ten isolated vertices emerge from the yellow, bright-blue circles and the triangle groups, as shown in figure 6(b), all of which are relatively looser than the others. The 12th group consists of 8 independent teams, presented by green triangles, as shown in figure 6(a); since they are the independent teams, two of them are likely mismatched into the green circle group, and another two are arranged into the two cograph communities in the shaded area, as shown in figure 6(b). Other than the independent and isolated teams, all of the other teams are arranged correctly.

4.1.2. Biological networks

In the following, we perform experiments on biological networks, and we test the quality of the algorithm for community or module detection by how well it can be applied to make predictions for protein complexes and GO terms. Protein complexes typically have a dense modular structure within which proteins are highly

connected. To examine whether the detected modules capture protein functional relationships other than just protein complexes, we use the high-level GO terms in all three domains (MF, BP and CC) as the golden standards for GO term prediction.

To evaluate the performance for the complex prediction, we use two independent quality measures [50] to assess the similarities between the predicted complexes and the golden standard reference complexes. In our experiments, we do not consider the one-protein module for all of the compared algorithms. The first measure counts the number of predicted modules that match the golden standards. A predicted module N_1 with V_{N_1} proteins or genes is thought to match with a reference module N_2 with V_{N_2} proteins or genes when the neighborhood affinity is

$$NA\left(N_1, N_2\right) = \frac{\left|V_{N_1} \cap V_{N_2}\right|^2}{\left|V_{N_1}\right| \times \left|V_{N_2}\right|} \geqslant \omega, \tag{2}$$

where the threshold ω is usually set as 0.2 or 0.25 [51, 52]. The second measure is the geometric mean of two other measures, which are the cluster-wise sensitivity (Sn) and the cluster-wise positive predictive value (PPV) [52]. Given that r is predicted and s is the reference complexes, let t_{ij} denote the number of proteins that exist in both predicted complex i and reference complex j, and w_j represents the number of proteins in reference complex j. Then, Sn and PPV can be defined as

$$Sn = \frac{\sum_{j=1}^{s} \max_{i=1,\cdots,r}\left\{t_{ij}\right\}}{\sum_{j=1}^{s} w_j}, \tag{3}$$

$$\mathrm{PPV} = \frac{\sum_{i=1}^{r} \max_{j=1,\cdots,s}\left\{t_{ij}\right\}}{\sum_{i=1}^{r}\sum_{j=1}^{s} t_{ij}}, \tag{4}$$

respectively. Since Sn can reach its maximum by grouping all proteins in one module, and PPV can be maximized by putting each protein in its own module, we use their geometric mean

$$Acc = \sqrt{Sn \times \mathrm{PPV}} \tag{5}$$

as 'accuracy' to balance these two measures [50, 52], whereas higher Acc scores the better results.

To investigate the functional significance of identified modules, we follow the same strategy as used in the literature [25, 47] to compute the F-measure based on high-level GO term prediction. The neighborhood affinity score between a predicted module p and a real GO term rg, $NA(p, rg)$ is used to determine whether they match each other. If $NA(p, rg) \geqslant \omega$, they are considered to be matched with each other. Here, we set ω to 0.20, as was done in the literature [51]. We assume that PC and RG are the sets of modules predicted by a computational method and by real GO terms, respectively. N_{cp} is the number of correct predictions which match at least a real GO term, and N_{crg} is the number of real GO terms that match at least a predicted one. Precision (P) and recall (R) are defined as follows [53]

$$N_{\mathrm{cp}} = \left|\{p \mid p \in PC, \exists\, rg \in RG, NA(p, rg) \geqslant \omega\}\right|, \tag{6}$$

$$N_{\mathrm{crg}} = \left|\{rg \mid rg \in RG, \exists\, p \in PC, NA(p, rg) \geqslant \omega\}\right|, \tag{7}$$

$$P = \frac{N_{\mathrm{cp}}}{|PC|}, \tag{8}$$

$$R = \frac{N_{\mathrm{crg}}}{|RG|}. \tag{9}$$

The F-measure (F) is the harmonic mean of precision and recall, and it is depicted as follows

$$F = \frac{2 \times P \times R}{P + R}. \tag{10}$$

Among the compared algorithms used in the previous section, here, GN, NMF and SC are not used to test on the *Sce*DIP and *Hsa*HPRD. GN is excluded, for it is too slow on large networks due to the expensive edge betweenness calculation. Both the NMF and SC need to fix the number of expected modules, which brings an inconvenient ambiguity when we face different golden standards. Although we can follow the same strategy to

implement a grid search using the number of expected modules $k = 500 \sim 3000$ with an interval of 100, as in the literature [25], we thought the step length to be too big. For uniformity in the comparisons, we exclude these three algorithms.

We first depict the results of the protein complex prediction in table 3; then, we display the results of the GO term prediction in figure 7. Table 3 shows the performance of complex predictions on the *Sce*DIP and *Hsa*HPRD in detail. The column headings of table 3 include the network for testing, the golden standard, the algorithms for comparison, the number of coverage proteins, the number of modules predicted, the average size of modules, the number of matched protein complexes, the cluster-wise sensitivity (Sn), the cluster-wise positive predictive value (PPV) and the accuracy score (Acc). We compare the results on the *Sce*DIP according to the *S. cerevisiae* protein complex golden standards MIPS and SGD, while on *Hsa*HPRD we compare the results according to the *H.sapiens* protein complex golden standards PCDq and CORUM.

As depicted in table 3, the *Acc* scores of the EPCA are obviously better than the ones of other algorithms on SGD, and the scores of the EPCA on the MIPS is 0.3749 lower than the highest 0.3960 of INFOMAP; the scores of the EPCA on the PCDq is 0.4607, which is just slightly lower than the highest 0.4613 of the $ECCA_D$, while on CORUM the score is 0.3088, which is also slightly lower than the highest 0.3153 of the MCL. The numbers of matched protein complexes of the EPCA are the largest ones among the compared algorithms on MIPS and PCDq, respectively. The average size of the modules of the EPCA on the PCDq is 4.53, which is close to the average size of the reference protein golden standards 4.51. Considering the *Acc* scores and the matched numbers of the compared algorithms, EPCA, MCL and $ECCA_D$ are the competitive ones, and they all outperform others dramatically. As for the aspect of the GO term prediction, shown in figure 7, this aspect (a) illustrates the F-measure of the compared algorithms and (b) shows the percentage of GO terms that are considered to be correctly matched to at least one of the identified modules by different algorithms on *Sce*DIP and *Hsa*HPRD, respectively. Figure 7 also clearly illustrates that the EPCA, MCL and $ECCA_D$ are competitive, as they also outperform others since the EPCA can obtain better F-measure scores on both the *Sce*DIP and *Hsa*HPRD while having slightly fewer matched GO terms than $ECCA_D$ on *Sce*DIP. Summarily, the EPCA is more attractive than MCL and $ECCA_D$ since the EPCA is free of any parameters and has comparatively lower computational cost. Namely, the inflation parameter of MCL can affect the granularity, and the $ECCA_D$ depends on the additional measure: the D value. Also, the $ECCA_D$ performs better than the $ECCA_Q$, emphasizing the drawback that the same algorithm operating with different additional measures can lead to different results.

To demonstrate the comparison intuitively, we display, for instance, the Arp2/3 complex predicted by the compared methods in figure 8. The Arp2/3 complex consists of seven-subunit proteins, which play a major role in the regulation of the actin cytoskeleton. As figure 8 shows, the EPCA can detect the complex perfectly, while MCL can obtain a module that includes ten proteins, with three additional proteins. The $ECCA_D$ can obtain four proteins of the Arp2/3 complex. None of the algorithms (EAC, EACH, INFOMAP and LOUVAIN) can extract a candidate complex, including YDL029W, which is an essential protein member of the Arp2/3 complex. The OSLOM2 obtains a candidate complex which includes 20 proteins, while 13 are not the correct members. Unfortunately, the CNM and $ECCA_Q$ cannot obtain a valuable candidate complex for the Arp2/3 complex.

4.2. Various meaningful subgroups realized as cotree subgroups

In this section we mainly make a next-level analysis for the cograph modules obtained by the EPCA based on their corresponding cotrees to obtain various meaningful subgroups. As introduced in section 3.2, the structure of a cograph module can be demonstrated more explicitly by the corresponding cotree. The reasons for detecting subgroups this way are that the vertices belonging to a subgroup must possess structural similarity to the rest of its network. This strategy lets us find not only the dense subgroups but also the sparse subgroups and even those with no connections within the subgroup. The simple fact is that finding sparse or non-connected subgroups is a remarkable property that exhibits the superiority of analyzing cograph modules by cotrees. In the following, as shown in figure 9, we analyze four typical cograph modules obtained from *Hsa*HPRD based on their cotrees in detail.

Figures 9(a), (c), (e) and (g) depict the first, second, third and fourth cograph module, while (b), (d), (f) and (h) display the corresponding cotrees, respectively. Figures 9(a) and (b) show the first cograph module, which consists of 6 genes and its corresponding cotree. The three genes PIWIL1, PIWIL4 and PIWIL2 are partitioned into the weak sibling subgroup in the light-blue region, which are all adjacent to DICER1 and nonadjacent to the strong siblings of TARBP2 and PRKRA. In fact, the weak sibling subgroup of PIWIL1, PIWIL4 and PIWIL2 perfectly matches the GO term 'piRNA binding', numbered by GO:0034584. Here, we want to emphasize that among these three genes, there are no interactions among them. So, if we use traditional algorithms that use density, we can never identify the subgroup matching the term GO:0034584, while the three genes TARBP2, PRKRA and DICER1 compose a subgroup shown in figure 9(b), which perfectly matches the term 'RNA interference, production of siRNA', numbered by GO:0030422. The strong siblings of TARBP2 and PRKRA

Table 3. Performance comparison for complex prediction on *Sce*DIP and *Hsa*HPRD.

N.w.[a]	G.s.[b]	Alg.[c]	Cov.[d]	Mo.n.[e]	A.s.[f]	Ma.n.[g]	*Sn*	PPV	*Acc*
S.[h]	M.[j]	—	1061	203	12.52	—	—	—	—
		$ECCA_Q$	4980	737	6.76	46	0.4486	0.2604	0.3418
		$ECCA_D$	4980	1563	3.19	75	0.2436	**0.4096**	0.3159
		CNM	4980	42	118.57	4	**0.6045**	0.1058	0.2529
		MCL	4736	928	5.10	69	0.3125	0.3689	0.3395
		EAC	1691	98	17.26	29	0.2916	0.2854	0.2885
		EACH	4980	98	50.82	20	0.4455	0.2191	0.3124
		INFOMAP	4980	441	11.29	47	0.4915	0.3190	**0.3960**
		OSLOM2	5442	85	64.02	21	0.5053	0.2382	0.3469
		LOUVAIN	4980	675	7.38	35	0.5081	0.2571	0.3614
		EPCA	4687	1019	4.60	**82**	0.3530	0.3982	0.3749
	S.[k]	—	1211	305	5.70	—	—	—	—
		$ECCA_Q$	4980	737	6.76	106	0.5549	0.3933	0.4672
		$ECCA_D$	4980	1563	3.19	**166**	0.4048	**0.6288**	0.5045
		CNM	4980	42	118.57	4	**0.7320**	0.0986	0.2687
		MCL	4736	928	5.10	124	0.5026	0.5585	0.5298
		EAC	1691	98	17.26	36	0.3899	0.3897	0.3898
		EACH	4980	98	50.82	34	0.5854	0.2720	0.3990
		INFOMAP	4980	441	11.29	74	0.6354	0.4447	0.5316
		OSLOM2	5442	85	64.02	24	0.6475	0.2745	0.4216
		LOUVAIN	4980	675	7.38	79	0.6538	0.3598	0.4850
		EPCA	4687	1019	4.60	129	0.5348	0.5943	**0.5638**
H.[i]	P.[l]	—	3433	1204	4.51	—	—	—	—
		$ECCA_Q$	9269	1464	6.33	194	0.4285	0.3717	0.3991
		$ECCA_D$	9269	2601	3.56	372	0.3590	**0.5927**	**0.4613**
		CNM	9269	96	96.55	21	**0.6426**	0.0486	0.1768
		MCL	8903	1789	4.98	316	0.3992	0.5322	0.4609
		EAC	3506	251	13.97	66	0.3041	0.2130	0.2545
		EACH	9269	251	36.93	56	0.4482	0.1514	0.2605
		INFOMAP	9269	668	13.88	150	0.5192	0.3266	0.4118
		OSLOM2	10016	208	48.15	19	0.5262	0.1686	0.2978
		LOUVAIN	9269	1097	8.45	226	0.5385	0.2944	0.3981
		EPCA	8807	1946	4.53	**377**	0.3856	0.5504	0.4607
	C.[m]	—	1955	1294	5.06	—	—	—	—
		$ECCA_Q$	9269	1464	6.33	166	0.4251	0.1907	0.2847
		$ECCA_D$	9269	2601	3.56	**278**	0.3212	**0.2720**	0.2956
		CNM	9269	96	96.55	12	**0.7333**	0.0334	0.1566
		MCL	8903	1789	4.98	190	0.4041	0.2460	**0.3153**
		EAC	3506	251	13.97	34	0.3650	0.0856	0.1768
		EACH	9269	251	36.93	20	0.4743	0.0793	0.1939
		INFOMAP	9269	668	13.88	73	0.5251	0.1591	0.2890
		OSLOM2	10016	208	48.15	20	0.5425	0.0970	0.2294
		LOUVAIN	9269	1097	8.45	95	0.5663	0.1310	0.2724
		EPCA	8807	1946	4.53	196	0.3772	0.2529	0.3088

[a] N.w. denotes the network.
[b] G.s. denotes the Golden standard.
[c] Alg. denotes the algorithm.
[d] Cov. denotes the number of coverage proteins.
[e] Mo.n. denotes the number of modules predicted.
[f] A.s. denotes the average size of the modules.
[g] Ma.n. denotes the number of matched modules.
[h] S. denotes *Sce*DIP.
[i] H. denotes *Hsa*HPRD.
[j] M. denotes MIPS.
[k] S. denotes SGD.
[l] P. denotes PCDq.
[m] C. denotes CORUM.

interact with the gene DICER1; together, they compose the dense larger subgroup of TARBP2, PRKRA and DICER1, a clique in fact, as shown in figure 9(a).

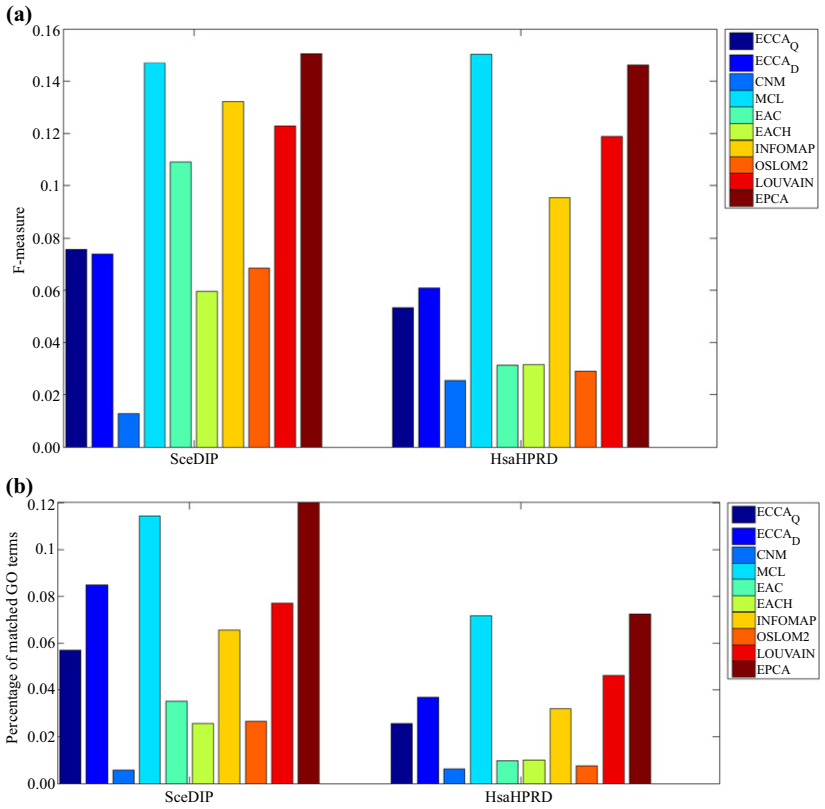

Figure 7. The bar plots illustrating the performance of algorithms for GO term prediction; (a) the comparison results based on the F-measure on *Sce*DIP and *Hsa*HPRD; (b) the comparison of the percentages of matched GO terms in the complete set of selected high-level GO terms.

The second cograph module and its cotree are shown in figure 9(c) and (d), respectively. The two distinct subgroups of weak siblings are revealed clearly by the cotree. One of the two subgroups consists of NPY1R, NPY2R, PPYR1 and NPY5R in the light-yellow region, perfectly matching the 1139th term of the 4457 golden standard terms. Furthermore, three genes of this subgroup are just the members of the 1396th term. We again emphasize that there are no interactions among these genes; so, density-based clustering methods would fail to detect them.

The third cograph module and its cotree are shown in figure 9(e) and (f), respectively. The cograph module consists of 9 genes that form a typical star-shaped subgraph, and we can obtain two subgroups intuitively from its cotree. One is the center BTK; the other is the weak siblings, including the rest of the 8 genes. Among the 8 genes, there are three genes in the light-blue region matching the term '1-phosphatidylinositol-5-phosphate 4-kinase activity', numbered by GO:0016309 perfectly.

The fourth cograph module and its cotree are shown in figure 9(g) and (h), respectively. The cograph module consists of 10 genes, and distinct subgroups are revealed by the cotree. Surprisingly, the strong siblings of BID and BAK1 in the light-blue region just match the three terms simultaneously. The three matched terms are the 'activation and oligomerization of BAK protein', numbered by REAC:111452; the 'tBID activates BAK protein', numbered by REAC:139895; and the 'tBID binds to inactive BAK protein', numbered by REAC:168848, respectively. Also, the genes BIK, BOK from the weak siblings and the gene BAK1 from the strong siblings together compose the subgroup which just matches the 1353th term. The cotree structure predicts more meaningful subgroups and especially predicts the weak siblings of PMAIP1 and BBC3 to have the same or similar functions since they are structurally-equivalent.

In summary, analyzing the cograph modules based on their corresponding cotrees can lead to an immediate prediction of distinctive subgroups. The vertices organized by the cotree reveal very fascinating subgroups which are not only dense but also sparse, even when there are no connections in the subgroups. Most importantly, some of the subgroups revealed in this manner have significant biological meanings and also match the corresponding GO terms perfectly.

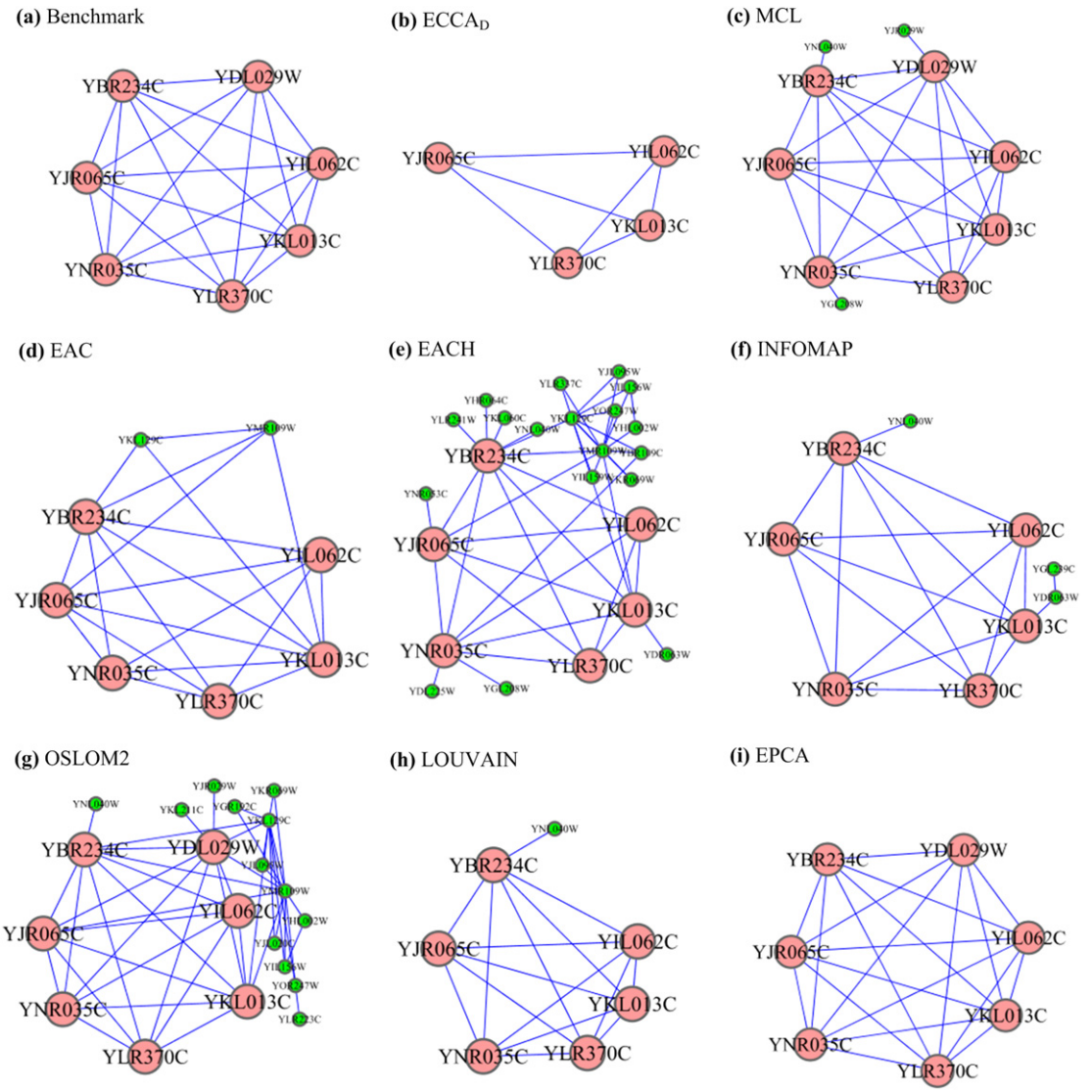

Figure 8. Illustration of the results predicted by the compared algorithms about the Arp2/3 complex. (a) The real Arp2/3 complex and (b-i) the candidate Arp2/3 complex predicted by the ECCA$_D$, MCL, EAC, EACH, INFOMAP, OSLOM2, LOUVAIN and EPCA, respectively, where the proteins in the orange color are the members of the real Arp2/3 complex, and those in the green color are not. CNM and ECCA$_Q$ cannot extract a valuable candidate complex for the Arp2/3 complex.

5. Conclusion and discussion

In this paper, we propose the novel cograph community and develop an approach (EPCA) for extracting cograph communities based on edge P_4 centrality. We compare the EPCA with algorithms from the existing literature on synthetic, social and biological networks to show that the EPCA has superior or competitive performance in accuracy and speed, in addition to having the advantages of being free of any parameters and independent of additional measures. The cograph communities have a fine granularity, and their diameters are at most a two-hop. More importantly, cograph communities exhibit a specialized internal structure, which decomposes the community into structurally-equivalent subgroups. The equivalence to cotrees allows a simple pictorial view for performing a next-level structural analysis for the purpose of finding meaningful subgroups that have functional similarity. In particular, these structurally-equivalent subgroups reveal homogeneous roles or functions that cannot be detected by traditional hierarchical clustering algorithms, which depend on edge density for community detection. Analyzing networks with cograph communities can contribute greatly to understanding the global structures and local structures of the networks more easily and distinctly.

Since edge P_4 centrality is defined for unweighted graphs, we cannot currently use the EPCA to detect cograph communities on weighted and directed networks. In a future study, we will attempt to develop an extended version of P_4 centrality for weighted and directed networks and propose a framework for detecting overlapping and hierarchical [6, 48, 54] cograph communities. Being able to identify structurally equivalent

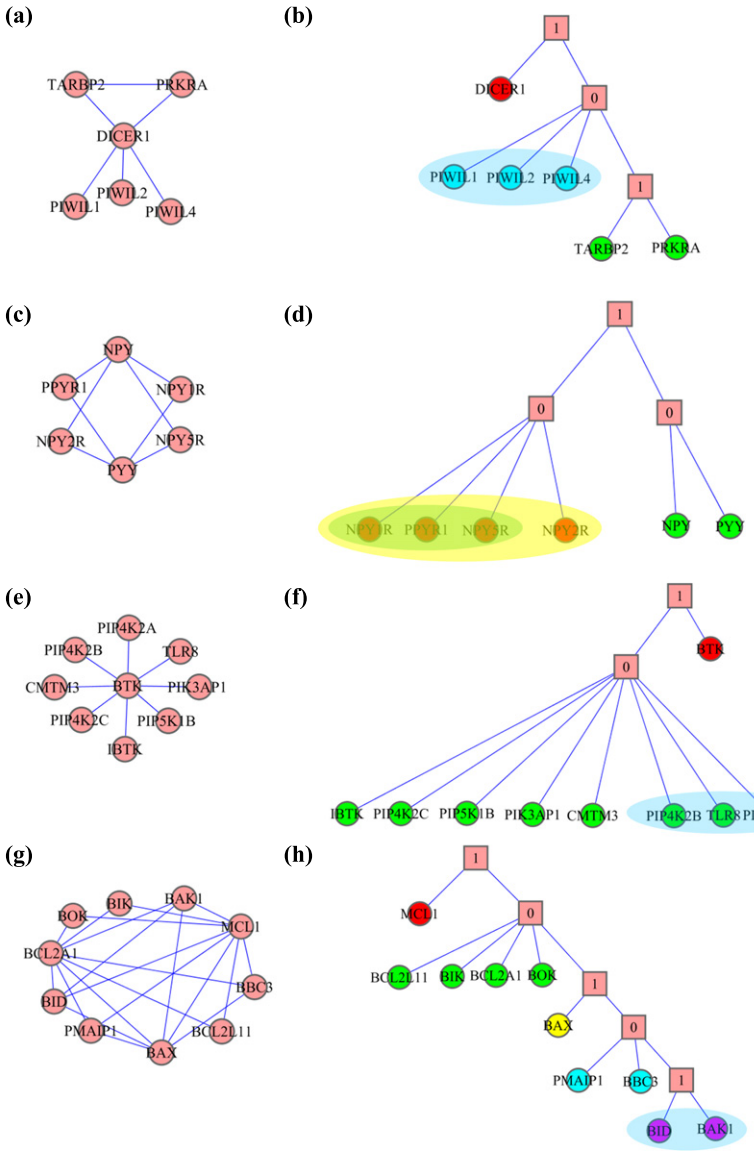

Figure 9. Illustration of cograph modules obtained by the EPCA from *Hsa*HPRD and their corresponding cotrees. (a), (c), (e) and (g) depict the first, second, third and fourth cograph modules, successively, while (b), (d), (f) and (h) display their corresponding cotrees.

subgroups in directed networks may have immediate applications to network controllability [55] by controlling communities with few drivers and thus controlling the whole network efficiently.

We feel that there are many interesting applications obtainable from studying other real biological or complex networks with the EPCA and cotrees. Another avenue of potential research is to explore other uses of these cograph communities and their cotree representations since they possess a multitude of algorithmic and structural properties and benefits besides subgroup detection.

We do, however, stress that although we have shown numerous benefits granted by the EPCA, such as polynomial runtime, ease of implementation, accuracy in finding cograph communities and the inherent ability to detect meaningful subgroups, we also find that it suffers from the transition between the undetectable and detectable regimes like virtually all community-finding algorithms [56–60], and we illustrate the details of this transition in appendix C, where we test the EPCA on Block model networks analogous to those used by Radicchi in [59].

Acknowledgments

We thank PhD candidate Xiaofei Yang and graduate student Jiapeng Yang for their beneficial discussions, PhD candidate Shihua Zhang for his suggestions, the anonymous editors and the referees for their helpful comments. We also gratefully acknowledge Professor Janos Kertesz's important suggestions and Professor F Radicchi's code

of generating the networks for testing the transition in appendix C. This work is supported by the National Natural Science Foundation of China (NSFC) under grant nos. 60933009, 91130006, 61303122, 61303118, 61202175, 61100157, 61202174 and by the Fundamental Research Funds for the Central Universities under Grant nos. BDZ021404, JB140609, BDY181417, K5051303010.

Appendix A. Simple construction of a cotree

We describe the simplest polynomial-time algorithm to compute a cotree of a cograph here [9]. Many existing algorithms in the literature run in linear time (which is $O(N^2)$ for dense graphs), and they rely on advanced techniques in graph algorithms. Here, we describe a very simple $O(N^3)$−time algorithm that will compute a cotree for a given connected cograph. The only extra knowledge required to understand this process is the operation of *graph complementation*: given a graph $G = (V, E)$, the complement of G (denoted co-G) is the graph on the same set of vertices V but with an edge uv in co-G only if there is no uv edge in G. Note that the complement of co-G is G itself.

The name *cograph* originates from the term *complement-reducible graphs*, which describes the original characterization of these graphs as those which can be completely decomposed into single vertices (leaf nodes) through successively taking graph complementation. That is, if a cograph is connected, its complement is always disconnected and thus decomposes into separate connected components.

A cotree is a tree that will contain two types of internal nodes: **1** and **0**, which represent a complete join or a disjoint union of the subgraphs below them. A **1** will only have **0** nodes (or leaf nodes) as its children and a **0** will only have **1** nodes (or leaf nodes) as its children. The leaf nodes of the tree will be the vertices of the graph. In this paper, the EPCA acts on a graph until it is guaranteed to be a cograph; so, for our purposes here, we will assume that we will always begin our cotree-building process on a connected cograph.

We will use the example cograph and cotree in figures 1(b) and (c) to illustrate this process. We initialize with a **1** node as the root and associate with it a set of all nodes of G, which denotes the set of all leaf nodes underneath this root. Since this is a cograph, its complement is disconnected; so, taking the graph complement of this cograph reveals a graph with two connected components: $S_1 = \{u, v, w, y\}$ and $S_2 = \{a, b, c, d, x, z, e\}$. Thus, underneath the **1** root node, we create two **0** nodes: one associated with S_1 and the other associated with S_2.

On each of these **0** nodes, we perform the graph complementation again in order to find further decomposition. However, recall that co-G is simply G; so, to obtain the second complement, we only take the induced graph on these sets: S_1 and S_2. We find that S_1 decomposes into the connected components $\{u, v\}$ and $\{w, y\}$; then, each of these is associated with its own **1** underneath S_1. Similarly, the induced graph of S_2 shows us the connected components $\{a, x, z\}, \{b, c, d\}$ and $\{e\}$; so, each of these is assigned to its own **1** underneath S_2.

When a node is trivially associated with a set of two or fewer vertices, those vertices are placed beneath that node as leaf nodes of the cotree. In our example, this means u and v are under their own **1** node, w and y are under their own **1** node, and e is under the **0** node, which was associated with S_2, as that is where it came from. Since $\{a, x, z\}$ and $\{b, c, d\}$ are associated to their own **1** nodes, we apply graph complementation to these and find the resulting connected components, which are now $\{a, x\}$ and $\{z\}$ in one case and $\{b, c\}$ and $\{d\}$ in the other. We are then left with trivial subgroups; so, all of the leaf nodes are constructed, and the cotree is complete.

Appendix B. The detail of normalized mutual information

In this paper we use $\mathrm{NMI}_{\mathrm{MGH}}$ [49] to evaluate the compared results for $\mathrm{NMI}_{\mathrm{MGH}}$ and to correct the so-called unintuitive behavior of $\mathrm{NMI}_{\mathrm{LFK}}$ [48]; we obtain an available code for $\mathrm{NMI}_{\mathrm{MGH}}$ from https://github.com/aaronmcdaid/Overlapping-NMI. In fact, $\mathrm{NMI}_{\mathrm{MGH}}$ is based on $\mathrm{NMI}_{\mathrm{LFK}}$ [48] in which the authors extend the normalized mutual information for evaluating overlapping communities from evaluating non-overlapping ones.

The definition of corresponding normalized mutual information $\mathrm{NMI}_{\mathrm{MGH}}$ [49] is demonstrated as

$$\mathrm{NMI}_{\mathrm{MGH}} = \frac{I(X{:}Y)}{\max(H(X), H(Y))}, \tag{B.1}$$

where $I(X{:}Y)$ is the mutual information and $H(X), (H(Y))$ are the unconditional entropy of cover $X, (Y)$.

$$I(X{:}Y) := \frac{1}{2}[H(X) - H(X|Y) + H(Y) - H(Y|X)], \tag{B.2}$$

$$H(X) = \sum_{i=1}^{K_X} H\left(X_i\right) = \sum_{i=1}^{K_X}\left(h\left(\sum_{m=1}^{n}\left[X_{im}=1\right], n\right) + h\left(\sum_{m=1}^{n}\left[\left[X_{im}=0\right], n\right]\right)\right), \tag{B.3}$$

where $\sum_{m=1}^{n}\left[X_{im}=1\right]$ counts the number of vertices in cluster i and $\sum_{m=1}^{n}\left[X_{im}=0\right]$ counts the number of vertices not in cluster i.

$$H(X \mid Y) = \sum_{i\in\left\{1,\ldots K_X\right\}} H\left(X_i\middle|Y\right). \tag{B.4}$$

$$H\left(X_i\middle|Y\right) = \min_{j\in\left\{1,\ldots K_Y\right\}} H^{*}\left(X_i\middle|Y_j\right). \tag{B.5}$$

$$H^{*}\left(X_i\middle|Y_j\right) = \begin{cases} H\left(X_i\middle|Y_j\right) \text{if } h(a,\ n) + h(d,\ n) \\ \quad \geqslant h(b,\ n) + h(c,\ n) \\ h(c+d,\ n) + h(a+b,\ n) \ \text{ otherwise} \end{cases}. \tag{B.6}$$

$$\begin{aligned} H\left(X_i\middle|Y_j\right) &= H\left(X_i,\ Y_j\right) - H\left(Y_j\right) \\ &= h(a,\ n) + h(b,\ n) + h(c,\ n) \\ &\quad + h(d,\ n) - h(b+d,\ n) - h(a+c,\ n). \end{aligned} \tag{B.7}$$

$$a = \sum_{m=1}^{n}\left[X_{im}=0 \wedge Y_{im}=0\right]. \tag{B.8}$$

$$b = \sum_{m=1}^{n}\left[X_{im}=0 \wedge Y_{im}=1\right]. \tag{B.9}$$

$$c = \sum_{m=1}^{n}\left[X_{im}=1 \wedge Y_{im}=0\right]. \tag{B.10}$$

$$d = \sum_{m=1}^{n}\left[X_{im}=1 \wedge Y_{im}=1\right]. \tag{B.11}$$

X and Y are matrices of the community membership. There are n objects. The first cover has K_X communities; hence, X is a $n \times K_X$ matrix, and Y is a $n \times K_Y$ matrix. X_{im} is 1 if vertex m is in community i in cover X. More details can be found in the original references [48, 49].

Appendix C. EPCA suffering from the transition between undetectable and detectable regimes

Recently, a very novel counter-intuitively paradox in-community detection was proposed by F Radicchi [59], which tells us that the detection of well-defined modules is more difficult than the identification of ill-defined communities. The paradox is mainly due to the fact that virtually all algorithms are affected by the so-called detectability threshold [56–60]. It has been shown that community identification algorithms are able to detect a modular structure only when $\Delta > \Delta_c$, where

$$\Delta = \left\langle k_{\text{in}} \right\rangle - \left\langle k_{\text{out}} \right\rangle, \tag{C.1}$$

$$\Delta_c = \sqrt{\left\langle k_{\text{in}} \right\rangle + \left\langle k_{\text{out}} \right\rangle}, \tag{C.2}$$

$\left\langle k_{\text{in}} \right\rangle$ ($\left\langle k_{\text{out}} \right\rangle$) is the average internal (external) degree. Here, we also test our proposed algorithm EPCA, which suffers from the so-called transition between undetectable and detectable regimes from numerical computations. The detectability threshold Δ_c depends not only on the average values of internal and external degrees but also on the correlation between their degree distributions. The correlations between the internal and external degree distribution are independent, positive and negative, respectively. Concretely, the tests are performed on the Block model [36], as used by F Radicchi [59]. As shown in figure C1, we test the EPCA on the Block model composed of two and four communities. We plot the fraction of vertices that the algorithm correctly classified as a function of Δ. Each node of the figure represents the average performance of the EPCA in 20 realizations, (a) the model composed of two communities with 50 vertices and the average degree $\left\langle k_{\text{in}} \right\rangle + \left\langle k_{\text{out}} \right\rangle = 16$ and (b) the model composed of four communities with 30 vertices and the average degree $\left\langle k_{\text{in}} \right\rangle + 3\left\langle k_{\text{out}} \right\rangle = 16$. Figure C1 obviously reveals that the algorithm EPCA does suffer from the transition

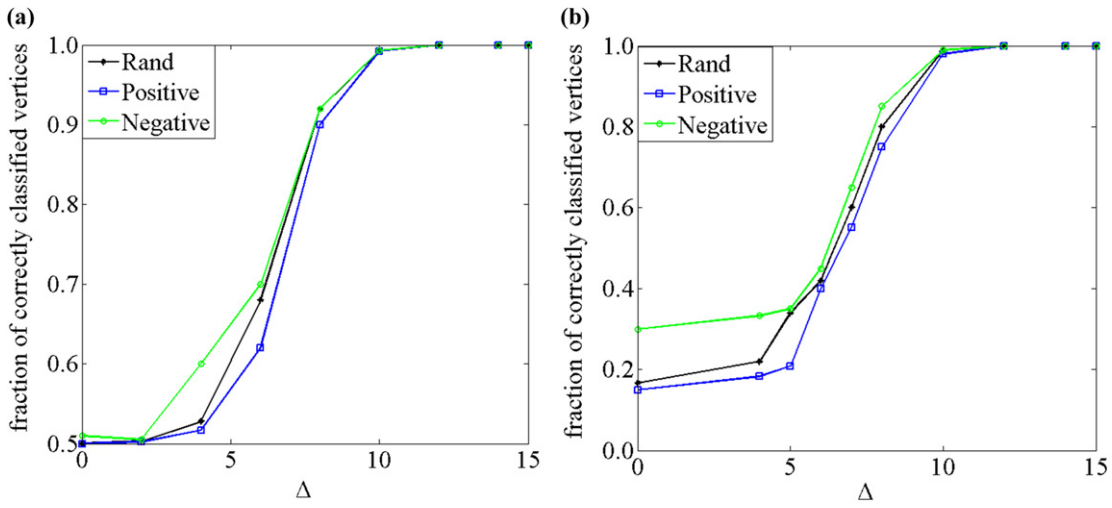

Figure C1. Illustration of how the EPCA suffers from the transition between undetectable and detectable regimes on the Block model; (a) two pre-imposed communities; (b) four pre-imposed communities.

between undetectable and detectable regimes; thus, we indeed need to reconsider the relation with the notation of communities and clusters identified by the algorithms.

References

[1] Fortunato S 2010 Community detection in graphs *Phys. Rep.* **486** 75–174
[2] Zaki N and Mora A 2014 A comparative analysis of computational approaches and algorithms for protein subcomplex identification *Sci. Rep.* **4** 4262
[3] Kovács I A, Palotai R, Szalay M S and Csermely P 2010 Community landscapes: an integrative approach to determine overlapping network module hierarchy, identify key nodes and predict network dynamics *PloS ONE* **5** e12528
[4] Nastos J and Gao Y 2013 Familial groups in social networks *Soc. Networks* **35** 439–50
[5] Martin G and Everett S P B 1998 Analyzing clique overlap *Connections* **21** 49–61
[6] Palla G, Derényi I, Farkas I and Vicsek T 2005 Uncovering the overlapping community structure of complex networks in nature and society *Nature* **435** 814–8
[7] Bu D *et al* 2003 Topological structure analysis of the protein-protein interaction network in budding yeast *Nucleic Acids Res.* **31** 2443–50
[8] Lancichinetti A, Kivelä M, Saramäki J and Fortunato S 2010 Characterizing the community structure of complex networks *PLoS ONE* **5** e11976
[9] Corneil D G, Lerchs H and Burlingham L S 1981 Complement reducible graphs *Discrete Appl. Math.* **3** 163–74
[10] Liu Y, Wang J, Guo J and Chen J 2012 Complexity and parameterized algorithms for cograph editing *Theor. Comput. Sci.* **461** 45–54
[11] Nastos J and Gao Y 2012 Bounded search tree algorithms for parameterized cograph deletion: efficient branching rules by exploiting structures of special graph classes *Discrete Math. Algorithms Appl.* **04** 1250008
[12] Gao Y, Hare D R and Nastos J 2013 The cluster deletion problem for cographs *Discrete Math.* **313** 2763–71
[13] Hellmuth M, Hernandez-Rosales M, Huber K T, Moulton V, Stadler P F and Wieseke N 2013 Orthology relations, symbolic ultrametrics, and cographs *J. Math. Biol.* **66** 399–420
[14] Habib M and Paul C 2005 A simple linear time algorithm for cograph recognition *Discrete Appl. Math.* **145** 183–97
[15] Bretscher A, Corneil D, Habib M and Paul C 2008 A simple linear time LexBFS cograph recognition algorithm *SIAM J. Discrete Math.* **22** 1277–96
[16] Jia S, Gao Y, Gao L and Wang H 2014 Anti-triangle centrality-based community detection in complex networks *IET Syst. Biol.* **8** 116–25
[17] Radicchi F, Castellano C, Cecconi F, Loreto V and Parisi D 2004 Defining and identifying communities in networks *Proc. Natl Acad. Sci. USA* **101** 2658–63
[18] Newman M E J and Girvan M 2004 Finding and evaluating community structure in networks *Phys. Rev.* E **69** 026113
[19] Girvan M and Newman M E J 2002 Community structure in social and biological networks *Proc. Natl Acad. Sci.* **99** 7821–6
[20] Zachary W W 1977 An information flow model for conflict and fission in small groups *J. Anthropol. Res.* **33** 452–73
[21] Lancichinetti A, Fortunato S and Radicchi F 2008 Benchmark graphs for testing community detection algorithms *Phys. Rev.* E **78** 046110
[22] Salwinski L, Miller C S, Smith A J, Pettit F K, Bowie J U and Eisenberg D 2004 The database of interacting proteins: 2004 update *Nucleic Acids Res.* **32** D449–51
[23] Blanc R 1986 Introduction to Percolation theory *Contribution of Clusters Physics to Materials Science and Technology* (*NATO ASI Series*) ed J Davenas and P M Rabette (Dordrecht: Springer) pp 425–78
[24] Bunde A and Havlin S 1991 *Fractals and Disordered Systems* (New York: Springer)
[25] Wang Y and Qian X 2014 Functional module identification in protein interaction networks by interaction patterns *Bioinformatics* **30** 81–93
[26] Ahn Y-Y, Bagrow J P and Lehmann S 2010 Link communities reveal multiscale complexity in networks *Nature* **466** 761–4
[27] Lee D D and Seung H S 1999 Learning the parts of objects by non-negative matrix factorization *Nature* **401** 788–91

[28] Lee D and Seung S 2000 Algorithms for non-negative matrix factorization *Proc. 13th Conf. on Advances in Neural Information Processing Systems* ed T K Leen, T G Dietrich and V Tresp (Cambridge, MA: MIT Press) pp 556–62
[29] Newman M E J 2006 Finding community structure in networks using the eigenvectors of matrices *Phys. Rev.* E **74** 036104
[30] Luxburg U V 2007 A tutorial on spectral clustering *Stat. Comput.* **17** 395–416
[31] Clauset A, Newman M E J and Moore C 2004 Finding community structure in very large networks *Phys. Rev.* E **70** 066111
[32] Enright A J, Van Dongen S and Ouzounis C A 2002 An efficient algorithm for large-scale detection of protein families *Nucleic Acids Res.* **30** 1575–84
[33] Rosvall M and Bergstrom C T 2008 Maps of random walks on complex networks reveal community structure *Proc. Natl Acad. Sci.* **105** 1118–23
[34] Lancichinetti A and Fortunato S 2009 Community detection algorithms: a comparative analysis *Phys. Rev.* E **80** 056117
[35] Lancichinetti A, Radicchi F, Ramasco J J and Fortunato S 2011 Finding statistically significant communities in networks *PLoS ONE* **6** e18961
[36] Blondel V D, Guillaume J-L, Lambiotte R and Lefebvre E 2008 Fast unfolding of communities in large networks *J. Stat. Mech. Theory Exp.* P10008
[37] Li Y and Ngom A 2013 The non-negative matrix factorization toolbox for biological data mining *Source Code Biol. Med.* **8** 1–15
[38] Walesiak M, Dudek A and Dudek M A 2008 ClusterSim: searching for optimal clustering procedure for a data set *R Package Version* 036-1 (available at: http://keii.ue.wroc.pl/clusterSim)
[39] Csardi G and Nepusz T 2006 The igraph software package for complex network research *InterJournal Complex Syst.* **5** 1695
[40] Lusseau D, Schneider K, Boisseau O J, Haase P, Slooten E and Dawson S M 2003 The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations *Behav. Ecol. Sociobiol.* **54** 396–405
[41] Evans T S 2010 Clique graphs and overlapping communities *J. Stat. Mech. Theory Exp.* P12037
[42] Prasad T K, Goel R, Kandasamy K, Keerthikumar S, Kumar S, Mathivanan S, Telikicherla D, Raju R, Shafreen B and Venugopal A 2009 Human protein reference database—2009 update *Nucleic Acids Res.* **37** D767–72
[43] Mewes H-W, Amid C, Arnold R, Frishman D, Güldener U, Mannhaupt G, Münsterkötter M, Pagel P, Strack N and Stümpflen V 2004 MIPS: analysis and annotation of proteins from whole genomes *Nucleic Acids Res.* **32** D41–4
[44] Hong E L, Balakrishnan R, Dong Q, Christie K R, Park J, Binkley G, Costanzo M C, Dwight S S, Engel S R and Fisk D G 2008 Gene Ontology annotations at SGD: new data sources and annotation methods *Nucleic Acids Res.* **36** D577–81
[45] Kikugawa S, Nishikata K, Murakami K, Sato Y, Suzuki M, Altaf-Ul-Amin M, Kanaya S and Imanishi T 2012 PCDq: human protein complex database with quality index which summarizes different levels of evidences of protein complexes predicted from H-Invitational protein-protein interactions integrative dataset *BMC Syst. Biol.* **6** S7
[46] Ruepp A *et al* 2008 CORUM: the comprehensive resource of mammalian protein complexes *Nucleic Acids Res.* **36** D646–50
[47] Shih Y-K and Parthasarathy S 2012 Identifying functional modules in interaction networks through overlapping Markov clustering *Bioinformatics* **28** i473–9
[48] Lancichinetti A, Fortunato S and Kertész J 2009 Detecting the overlapping and hierarchical community structure in complex networks *New J. Phys.* **11** 033015
[49] McDaid A F, Greene D and Hurley N 2011 Normalized mutual information to evaluate overlapping community finding algorithms (arXiv:11102515)
[50] Nepusz T, Yu H and Paccanaro A 2012 Detecting overlapping protein complexes in protein-protein interaction networks *Nat. Methods* **9** 471–2
[51] Wu M, Li X, Kwoh C-K and Ng S-K 2009 A core-attachment based method to detect protein complexes in PPI networks *BMC Bioinformatics* **10** 169
[52] Li X, Wu M, Kwoh C-K and Ng S-K 2010 Computational approaches for detecting protein complexes from protein interaction networks: a survey *BMC Genomics* **11** S3
[53] Chua H N, Ning K, Sung W-K, Leong H W and Wong L 2008 Using indirect protein–protein interactions for protein complex prediction *J. Bioinform. Comput. Biol.* **6** 435–66
[54] Gregory S 2010 Finding overlapping communities in networks by label propagation *New J. Phys.* **12** 103018
[55] Liu Y-Y, Slotine J-J and Barabási A-L 2011 Controllability of complex networks *Nature* **473** 167–73
[56] Decelle A, Krzakala F, Moore C and Zdeborová L 2011 Inference and phase transitions in the detection of modules in sparse networks *Phys. Rev. Lett.* **107** 065701
[57] Nadakuditi R R and Newman M E J 2012 Graph spectra and the detectability of community structure in networks *Phys. Rev. Lett.* **108** 188701
[58] Radicchi F 2013 Detectability of communities in heterogeneous networks *Phys. Rev.* E **88** 010801
[59] Radicchi F 2014 A paradox in community detection *EPL* **106** 38001
[60] Radicchi F 2014 Driving interconnected networks to supercriticality *Phys. Rev.* X **4** 021014