Large-scale community detection based on node membership grade and sub-communities integration

Shang, Ronghua; Luo, Shuang; Li, Yangyang; Jiao, Licheng; Stolkin, Rustam

DOI:
10.1016/j.physa.2015.02.004

Accepted Manuscript

Large-scale community detection based on node membership grade and sub-communities integration

Ronghua Shang, Shuang Luo, Yangyang Li, Licheng Jiao, Rustam Stolkin

PII: S0378-4371(15)00100-4
DOI: http://dx.doi.org/10.1016/j.physa.2015.02.004
Reference: PHYSA 15876

To appear in: *Physica A*

Received date: 25 August 2014
Revised date: 28 December 2014

Please cite this article as: R. Shang, S. Luo, Y. Li, L. Jiao, R. Stolkin, Large-scale community detection based on node membership grade and sub-communities integration, *Physica A* (2015), http://dx.doi.org/10.1016/j.physa.2015.02.004

Large-scale community detection based on node membership grade and sub-communities integration

Ronghua Shang[a], Shuang Luo[a], Yangyang Li[a], Licheng Jiao[a] and Rustam Stolkin[b]

([a] Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education of China, Xidian University, Xi'an, China; [b] School of Mechanical Engineering, University of Birmingham, UK)

***Abstract*:** Community detection plays an important role in research on network characteristics and in the mining of network information. A variety of algorithms have previously been proposed, but with the continuous growth of network scale, few of them can detect community structure efficiently. Additionally, most of these algorithms only consider non-overlapping community structures in networks. This paper addresses these problems by proposing a new algorithm, based on node membership grade and sub-communities integration, to detect community structure in large-scale networks. The proposed algorithm firstly introduces two functions based on the local information of each node in networks, namely neighboring inter-nodes membership function $f_{MS\text{-}NN}$ and node-to-community membership function $f_{MS\text{-}NC}$. Firstly, local potential complete sub-graphs are efficiently mined using the function $f_{MS\text{-}NN}$, and then these small graphs are merged into larger ones in light of local modularity. Secondly, incorrectly divided nodes are modified according to function $f_{MS\text{-}NN}$. Additionally, by adjusting the parameters in $f_{MS\text{-}NC}$, we can accurately obtain both non-overlapping communities and overlapping communities. Furthermore, the proposed algorithm employs a framework resembling label propagation, which has low time complexity and is suitable for detecting communities in large-scale networks. Experimental results on both artificial networks and real networks indicate the accuracy and efficiency of the proposed algorithm.

Keywords: Large-scale network; node membership function; sub-communities integration; overlapping community

1. Introduction

Complex networks are prevalent throughout the natural world, human interactions and computer systems, e.g. the World Wide Web, interpersonal networks, biological networks, and many other examples [1-6]. An important property that exists in many of these networks is community structure [7-9]. A community is a set of nodes that connect more closely with each other than they do with other nodes in different communities [8-9]. Individuals of the same community usually have common characteristics [10] [45]. For example, web pages with similar subjects compose a community in the worldwide web network [11]. Additionally, it can be seen that individuals with similar characteristics often share more dense connections with each other

than they do with other parts of the same network. Thus, detecting community structures is helpful in understanding the structure and functioning of networks [22] and can also help to detect potentially useful information within a network, through mining relations between individuals.

The problem of community detection is an area of rapidly growing interest within the complex network analysis research community, and a variety of methods have been proposed for its solution. Well-known approaches can be broadly categorized as graph partitioning methods [12-14], hierarchical clustering algorithms [15-17], and evolutionary algorithms [18-19].

Kernighan-Lin algorithm [12] is a well-known graph partitioning method, which works by randomly dividing a network into two communities, and then iteratively exchanging the nodes of the two communities until a modularity measure Q (see [23]) is maximized. Spectral bisection [13] also works by separating the network into two parts, using a Laplace matrix. Both of these algorithms rely on accurate prior knowledge of community size; they can only perform a simple binary division of a network into two communities; also their time complexity is high.

The hierarchical clustering algorithm is based on notions of similarity between the nodes and edge betweenness. This class of algorithms is "hierarchical" in the sense that clusters are recursively merged (agglomerative methods) or split (divisive methods) as one moves up or down the hierarchy respectively. For example, GN [15], proposed in 2002, recursively removes whichever residual edge has the largest edge betweenness, thereby progressively decomposing a network into a number of smaller clusters. However, computing the betweenness of all the edges is time-consuming.

In 2008, Clara Pizzuti [18] first proposed the use of evolutionary algorithms to solve the problem of community detection. The algorithm uses a single objective evolutionary algorithm to optimize community fraction *CS* as its objective function. Inspired by this method, Gong [20] et al proposed a memetic algorithm to optimize modularity density *D* to extract multilevel community structures. In 2013, Shang et al [21] improved on [20] by incorporating additional kinds of prior knowledge and using simulated annealing as a local search strategy to optimize a modularity measure, Q. In addition to the above methods, other algorithms have recently been proposed for detecting overlapping community structures. Some of these methods firstly extract maximal sub-graphs from the original networks and then merge small sub-graphs according to some index or strategy [57-59]. Other methods detect overlapping nodes in bipartite networks using key bi-communities and free-nodes [60]**.**

However, with the rapid growth in worldwide computer prevalence and connectedness, the corresponding expansion of individual's social circles, and the era of big data, the scale of networks is increasing, engendering a growing need for algorithms which are fast and efficient. In this lights some of the above methods are no longer suitable for community detection in large scale networks, for example the time complexity of KL [12] is $O(n^2)$ and GN [15] is $O(n^3)$.

Although evolutionary algorithms have shown potential for finding global optima, and are not constrained by the type of optimization function, they often take a long time to converge. Besides, the detection results still have some room for improvement, such as improving the detection precision and finding more multilevel solutions.

To overcome the limitations of the above algorithms, this paper proposes a large scale community detection algorithm based on node membership grade and sub-communities integration. First, we propose a neighboring inter-nodes membership function $f_{MS\text{-}NN}$ to evaluate the closeness of each node with its neighbors. Through merging the couple-node with highest $f_{MS\text{-}NN}$ value, this method can quickly find the potential complete sub-graph structures and effectively obtain a preliminary partitioning for the network. Next, those sub-communities achieved from the above steps are integrated by optimizing modularity Q. However, once these sub-communities have been merged together, it is difficult to correct nodes that have been wrongly placed. Therefore we propose another membership function $f_{MS\text{-}NC}$ which is used to estimate the intimacy of nodes that connect with different adjacent communities and can modify misclassified nodes, thereby preventing the result from falling into local optima. Finally, through adjusting the parameters of $f_{MS\text{-}NC}$, the proposed algorithm can be used to detect overlapping nodes and find overlapping community structures at different levels. Because the proposed algorithm adopts a learning strategy similar to label propagation, which involves only local information in each iteration, our method has low time complexity and is therefore suitably efficient for detecting communities in large and medium scale networks.

The remaining part of this paper is arranged as follows. In section 2, related algorithms are introduced and the motivations for the proposed algorithm are explained. In section 3, the details of is the proposed algorithm are described. Section 4 presents the results of experiments on both artificial and real networks. Section 5 discusses the results and presents conclusions.

2. Related works and motivation

In this section, we will introduce some related strategies employed for community detection in large scale networks, and discuss the motivations for designing the new algorithm proposed in this paper.

2.1 Local modularity

Modularity Q, proposed by Newman [23], is used as a general evaluation index of the partitioning result. A variety of algorithms have been proposed for dividing a network into communities by maximizing Q. However, calculating modularity Q requires global information of a network, which causes fundamental problems for community detection as the scale of networks becomes large. Therefore, to improve the detection efficiency in large-scale networks, more recent

algorithms have been proposed which exploit the local information of each node, [24-29], especially those which are based on local modularity optimization [25-29].

This paper is particularly concerned with networks which are unweighted and undirected, so that the local modularity incremental function can be reduced to:

$$\Delta Q_{i\to j} = \frac{l_{i,j}}{m} - \frac{d_i d_j}{2m^2} \tag{1}$$

Equation (1) shows the increment of modularity Q when a node *i* (or community *i*) merges with node *j* (or community *j*), where $l_{i,j}$ represents the connections between node *i* (or community *i*) and node *j* (or community *j*), d_i and d_j donates the degrees of all the nodes in node *i* (or community *i*) and node *j* (or community *j*) respectively, while *m* is the number of edges in the whole network.

2.2 Efficient ways of optimizing local modularity

Methods for optimizing local modularity can be broadly divided into two categories：local node search strategies and sub-communities integration strategies. Local node search focuses on information about each node's neighbours in the network, and divides nodes into communities according to an optimization function. For example, LPAm [26] employs the framework of LPA [25], which treats each node as a separate community with its own label. In each iteration, LPAm updates each node's label according to equation function which is equivalent to local modularity, converging on an optimized set of communities corresponding to an increase in modularity optimization. LPAm has low time complexity, O(*m*), and is more stable than LPA. In order to overcome the vulnerability of LPAm to local optima convergence, Liu et al [27] extended LPAm by incorporating the sub-communities integration strategy of multistep greedy, similar to MSG [28]. The sub-communities obtained by LPAm are iteratively merged according to the local modularity function until no further improvement can be made. The optimization result is greatly improved, and its time complexity is only $O(m\log^2 n)$.

An alternative approach to optimizing local modularity are sub-communities integration strategies, which merge existing sub-communities in a greedy way. FM [29], initializes each node as a separate small community, and iteratively merges whichever pair of current sub-communities causes the largest increment of local modularity. This procedure is repeated until no pair of communities can be merged to make a positive improvement in local modularity. The algorithm has a time complexity of $O(n\log^2 n)$. BGLL [30] is another algorithm which uses a sub-communities integration strategy. In contrast to FM [29], in BGLL the pair of sub-communities to be merged need not be globally optimal, but is only required to cause a locally optimal increase in modularity, Q. This method has a near linear time complexity for sparse networks and achieves good detection results. However, like LPAm, BGLL is prone to local optima convergence because, during the merging of sub-communities, if a single node is wrongly

assigned, it will become part of a larger community, after which it cannot be divided back out of that community in future iterations. To overcome this problem, a correction method was proposed by Rotta et al [31]. In this method, a multi-level correction strategy is employed, that employs a local node search strategy during each iteration.

From the above discussion it is apparent that the idea of combining local nodes search and community integration together, can help algorithms overcome vulnerability to local optima, while offering the potential for low computational complexity. Hence, in this paper, the idea of combining these two strategies is employed for detecting community structures in large scale networks.

2.3 Pre-processing method

Unfortunately, the algorithms discussed in section 2.2 share a common problem. After initialization, when every node is individually labeled as a separate community, in accordance with the formula (2), $l_{i,j}$ will be equal to unity. Consequently, the maximum increment of local modularity will correspond to two nodes with smaller degrees, d_i and d_j, so that such nodes are more likely to be partitioned into the same community [32]. This tends to contradict the principle that individuals with closer connections should be partitioned together. Thus, inspired by literature [37-40] which adopts pretreatment methods, we firstly employ a neighboring inter-nodes membership function named as $\boldsymbol{f_{MS\text{-}NN}}$ to generate a preliminary community division for the network. This function, based on the neighboring nodes membership relation, divides those nodes with more close connections into the same community. At the same time, by adjusting the parameters of $\boldsymbol{f_{MS\text{-}NN}}$, potential complete sub-graphs can be found, which helps to improve the accuracy of the algorithm. The details of this pre-processing step are introduced in Section 3.2.

2.4 Detecting overlapping communities based on non-overlapping structures

Overlapping communities are widely prevalent in real networks, but are comparatively under-explored in the community detection literature. Existing methods for overlapping community detection in large-scale networks include COPRA [33] and LFM [34]. These are comparatively fast algorithms with low computational (using objective functions based on the local node information or optimization of local fitness), however their classification accuracy is relatively poor. Other algorithms, that are suitable for large-scale networks, detect overlapping communities based on already having prior knowledge in the form of existing accurate detection results for non-overlapping communities. For example, the CONA algorithm [35] efficiently detects overlapping communities based on using the BGLL and Infomap [36] division results as a starting point. High quality overlapping community detection with this method, is often dependent on first establishing accurate knowledge of the non-overlapping communities within networks.

3. The proposed method

This section introduces the design of neighboring inter-nodes membership function $\boldsymbol{f_{MS\text{-}NN}}$, the procedure for merging of sub-communities based on local modularity, the node-to-community membership function $\boldsymbol{f_{MS\text{-}NC}}$, and the detection of overlapping communities. Finally, the overall framework of the algorithm is presented and the time complexity of the algorithm is analyzed.

3.1 Representation and decoding

Consider a network $G=\{V, E\}$, where V represents the vertex set and E is the edge set, $|V|=n$ is said to be the number of nodes in the network and $|E|=m$ is the total number of edges in the network. Here we use the real coded representation as the network partition:

$$\boldsymbol{g} = [x_1, x_2, ...x_i, ...x_n], \text{where } i = 1,2,3,...n, x_i \in [1,n] \tag{2}$$

Where $\boldsymbol{g}$ means a partition of the network and x_i is an integer representing the label of the community to which node i belongs. According to expression (2), if $x_i = x_j$, for i, j=1, 2, ..., n, then node i and node j belong to the same community. For example, for a network with 12 nodes shown in Fig.1(a), if the partitioning result is $\boldsymbol{g}_1$=[2, 2, 1, 1, 2, 2, 1, 1, 2, 3, 3, 3], that means node set {3, 4, 7, 8} is in the community 1 and node set {1, 2,5,6,9} is in community 2, and the rest nodes are in community 3. The corresponding community structure can be displayed by Fig. 1(b), which is shown in different colors and shapes.

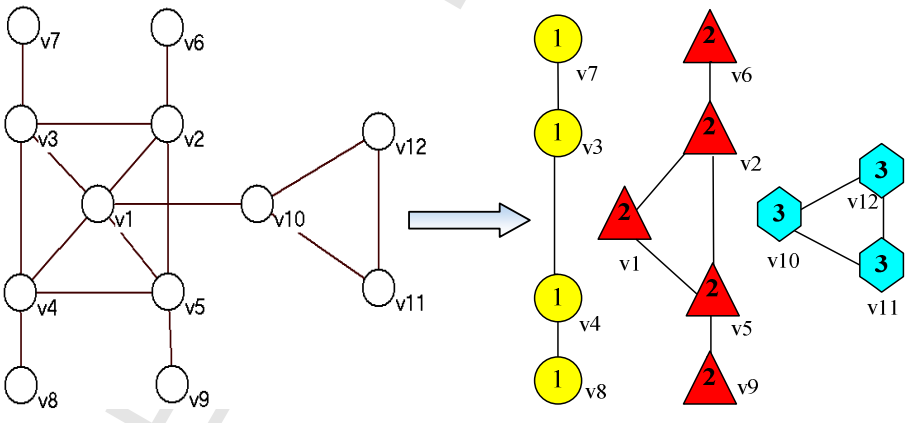


Fig.1. Test network 1

3.2 Preprocessing based on neighboring inter-nodes membership function $\boldsymbol{f_{MS\text{-}NN}}$

The question of how to judge the similarity of nodes in a network is an important problem. Commonly, the intimacy between any two nodes is decided according to the number of their common neighbors, e.g. cosine similarity [61] or Jaccard similarity [62]. As a simple example, consider that two people could be defined as knowing each other well, if they share many

common friends. If most people connected with individual A are also known to individual B, then we can infer that A belongs to the community of B. The extent to which B is connected to the individuals that connect with A, represents the membership degree with which A belongs to B. In light of this phenomenon, we propose a neighboring inter-nodes membership function $\boldsymbol{f_{MS\text{-}NN}}$ which indicates the membership degree with which *i* belongs to *j*:

$$f_{MS-NN}(i,j)=\frac{\left|\Gamma_i \cap \Gamma_j\right|+1}{d_i} \qquad (i=1,2,...n,\ \ j\in \Gamma_i) \tag{3}$$

Where *n* represents the number of nodes in the network, Γ_k is the neighbor set of node *k*, and d_k denotes its node degree. Greater values of $\boldsymbol{f_{MS\text{-}NN}}(i,j)$ indicate higher likelihoods that node *i* belongs to node *j*, and suggest that node *i* should be partitioned into the community of node *j*. If all the neighbors of node *i* are connected to node *j*, namely $\boldsymbol{f_{MS\text{-}NN}}(i,j)=1$, then node *i* is completely attributed to node *j*. TABLE 1 presents the procedure for network pretreatment using the function $\boldsymbol{f_{MS\text{-}NN}}$.

TABLE 1: The framework of pretreatment of a network based on function $\boldsymbol{f_{MS\text{-}NN}}$

Input: The node number *n*; initialization representation $\boldsymbol{g}$=[1,2,3…*n*];number of each node's neighbors *Neilen*; neighboring node information *Neiglist*={$\Gamma_1,\Gamma_2,\ldots\Gamma_k,\ldots\Gamma_n$}, *k*=1,2,…n, where Γ_k represents the neighbor nodes set of node *k*; parameter α.

Output: The preliminary partition result $\boldsymbol{g}$.

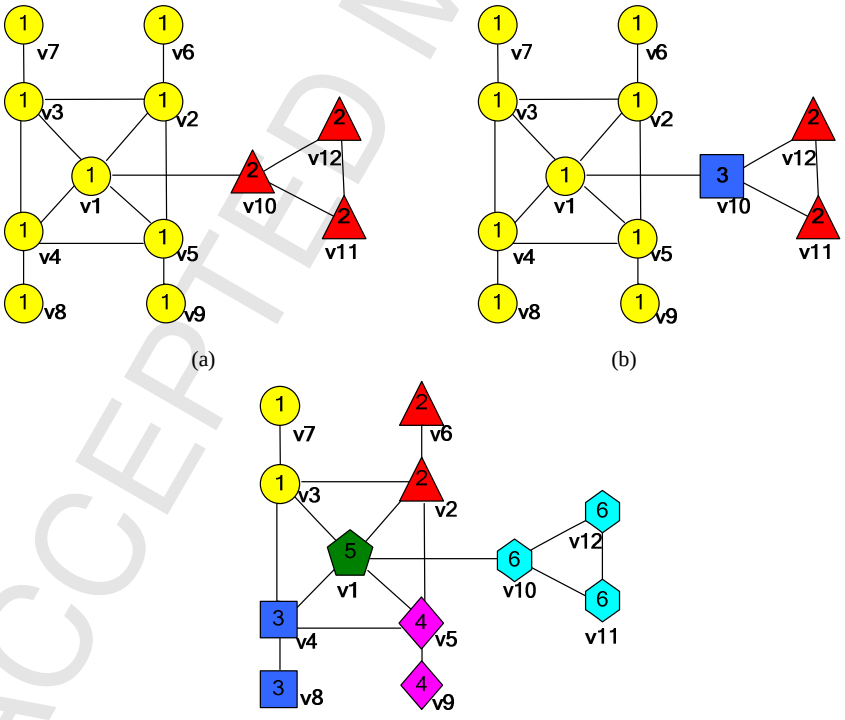
Step1: for i=1 to *n* do
Step2: *Neilen* <— the number of node *i*'s neighbors: $|\Gamma_i|$;
Step3: if *Neilen*≠0
Step4: for *j*=1 to *Neilen* do
Step5: $F_{MS\_NN}(j)$ <— Compute $f_{MS\text{-}NN}(i,j)$;
Step6: end for
Step7: end if
Step8: if max(F_{MS_NN})≥ α
Step9: Attribution node i_{ms_n} <— $\arg\max_l$ ($f_{MS\text{-}NN}(i,l)$), $l\in\Gamma_i$, (breaking ties randomly if more than one *l's* satisfy the condition) ;
Step10: Community label of node *i* : $\boldsymbol{g}(i)$ <—Community label of node i_{ms_n}: $\boldsymbol{g}(i_{ms\_n})$;
Step11: end if
Step11: end for
Step12: $\boldsymbol{g}$ <—Decode($\boldsymbol{g}$);

For example, Figure 1(a) is a simple network which contains 12 nodes and its community structure is clear. Intuitively, we can conclude that node *v*1 to node *v*9 belong to one community while nodes *v*10 to node *v*12 belong to a separate community. According to the framework shown in TABLE 1, we can pretreat the network presented in Fig.1. The corresponding membership function value of $\boldsymbol{f_{MS\text{-}NN}}$ for each node is shown in TABLE 2.

TABLE 2: The value of function $f_{MS\text{-}NN}$ of each node in test network 1

node i	node j	$f_{MS\text{-}NN}(i,j)$	node i	node j	$f_{MS\text{-}NN}(i,j)$
1	2	**0.60**	4	8	0.25
1	3	**0.60**	5	1	**0.75**
1	4	**0.60**	5	2	0.50
1	5	**0.60**	5	4	0.50
1	10	0.20	5	9	0.25
2	1	**0.75**	6	2	**1**
2	3	0.50	7	3	**1**
2	5	0.50	8	4	**1**
2	6	0.25	9	5	**1**
3	1	**0.75**	10	1	0.33
3	2	0.50	10	11	**0.67**
3	4	0.50	10	12	**0.67**
3	7	0.25	11	10	**1**
4	1	**0.75**	11	12	**1**
4	3	0.50	12	10	**1**
4	5	0.50	12	11	**1**

According to the results in Table 2, assuming that the membership function $f_{MS\text{-}NN}$ satisfies the condition that $f_{MS\text{-}NN}(i,j) \geq \alpha$, we will put node i into the community for which adjacent node j has the highest value of $f_{MS\text{-}NN}(i,j)$. For example, if α is set to be 0.75, the the node $v1$ will stay in its own community since no $f_{MS\text{-}NN}(1,j)$ exceeds the threshold. Node $v2$ is divided into the community of node $v1$ since $f_{MS\text{-}NN}(2,1)$ is the highest of all $f_{MS\text{-}NN}(2,j)$, where j=1,3,5,6,others followed by analogy. Ties are broken randomly if there is more than one highest value. Fig. 2(a), Fig. 2(b), Fig. 2(c) show the network pre-treatments corresponding to α set at 0.5, 0.75 and 1 respectively (different communities are shown in different colors):



(a) (b)

(c)

Fig.2. Preliminary partition results of test network 1. (a) Pre-treatment result when α=0.5. (b) Pre-treatment result when α=0.75. (c) Pre-treatment result when α=1.

As we can see from Figure 2, when α = 0.5, the preliminary partition result of network 1 is the most compatible with the intuitive result. When α increased to 0.75, node *v*10 becomes regarded as a sole community as it has the trend to be an overlapping one. If we set α = 1, which has the most stringent membership relation, then only those nodes who are completely affiliated to their neighbors become partitioned into the same community. Thus, changing the membership function threshold α can usefully lead to multi-level pretreatment results.

3.3 Sub-communities integration based on local modularity

Based on the preliminary partition result obtained in section 3.2, a sub-communities integration strategy is next employed. This method is similar to the second step in BGLL [30]. The first thing to do is to agglomerate those nodes in the same community as a new node. The method for agglomerating the nodes is as follows. Firstly select those nodes within a single common community as a whole group. Then this group is re-labeled as a "big new node". This "big new node" has both a self-link and external links. Its self-link is set to be twice the number of internal links of nodes within this group, and its external links are those that connect with other communities. The process of sub-communities integration of the pre-proceed network of Fig. 2(c) is shown in Fig. 3 (unlabeled lines connection degree is 1).

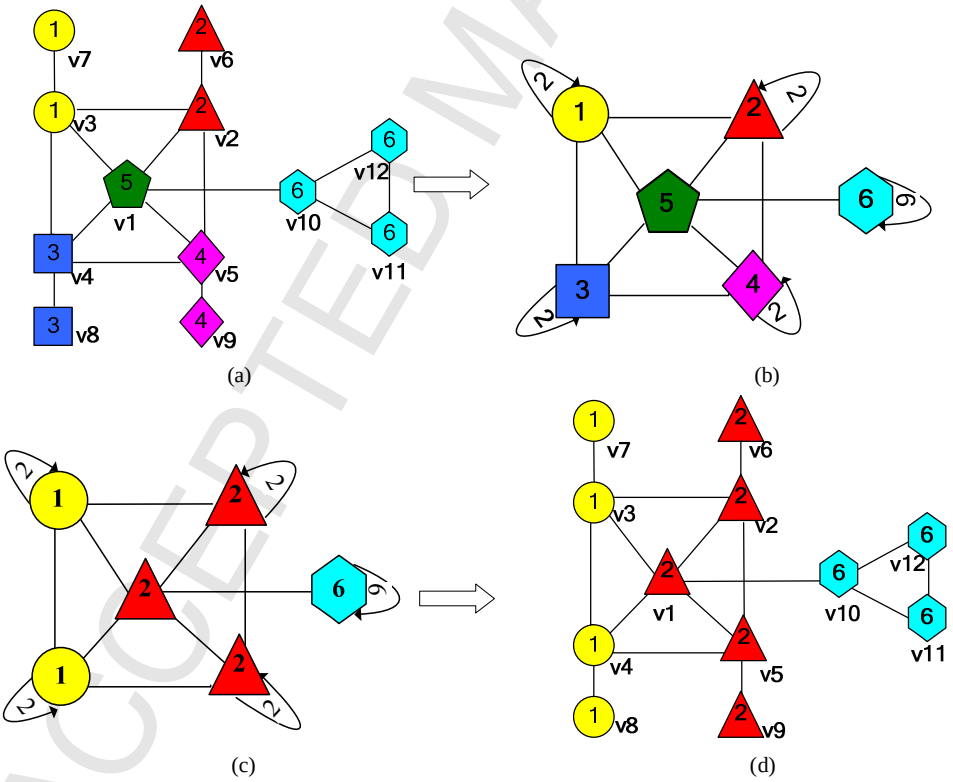


Fig.3. The process of sub-communities integration of network in Fig. 2(c). (a) The initial network before integration. (b) Agglomerate the node in same community. (c) Merge sub-communities according to local modularity. (d) The final result corresponding to the original network.

Figure 3 shows the process of using sub-communities integration, based on local modularity, to partition the network shown in Fig. 2(c). As shown in Figure 3 (a), *v*2 and *v*6 are in the same community, and the connection number between two nodes is 1. Therefore, these two nodes agglomerate into one node and its self-link becomes 2. After all nodes within the same community have been agglomerated into a single node, these new nodes will merge according to formula (1) to increase the local modularity. As shown in Figure 3(b) and Figure 3(c), new nodes 2, 4 and 5 agglomerate into one community 2, and new nodes 1, 3 become merged into one. Finally, the result is re-drawn corresponding to the nodes of the original network. At this stage, the partitioning result is the best (with Q=0.3223).

3.4 Modifying the network using node-to-community membership function $f_{MS\text{-}NC}$

From the analysis of Section 2, it is apparent that sub-communities integration strategies based on incremental local modularity can efficiently obtain partitions in large-scale networks. However, nodes that are wrongly partitioned once can never be correctly recovered, leading to a sub-optimal final result. Hence, we propose a strategy for modifying the partitions based on a node-to-community membership function $f_{\boldsymbol{MS\text{-}NC}}$. In the spirit of LPA [25], we begin with the assumption that the possibility of a node belonging to its adjacent nodes is in proportion to their connection number. However, such an approach is "node-centric", and ignores the extent to which a community might actually be receptive (or not) to its adjacent nodes. Therefore, we take both sides into consideration and propose a new membership function $f_{\boldsymbol{MS\text{-}NC}}$ to measure the intimacy of a node and its neighboring communities:

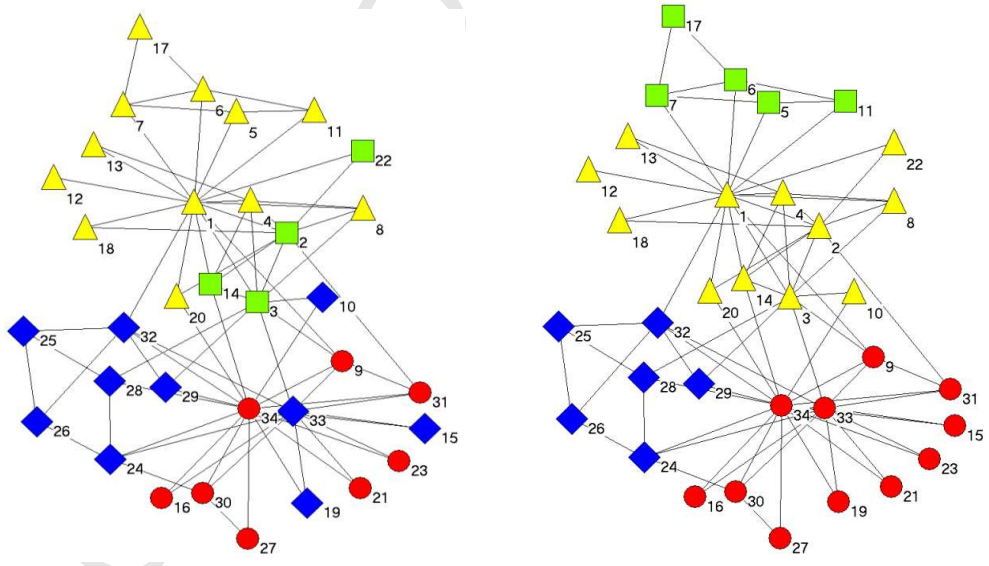
$$f_{MS-NC}(i,c) = \lambda \cdot \frac{l_{i,c}}{d_i} + \beta \cdot \frac{l_{i,c}}{|c|} \quad (i = 1,2,...n;\ \lambda, \beta \in [0,1], \lambda + \beta \neq 0) \tag{4}$$

Where $l_{i,c}$ represents the number of links between node i and its adjacent community c; $|c|$ represents the number of nodes in community c; λ and β are the parameters of this function and their values lie in the range [0,1]. The first term (before the "+") represents the possibility of a node belonging to its adjacent community c and the second term shows how likely community c is to accept node i. Since $0<l_{i,c}\leq d_i$, and the node number of community c *is* $|c|$, then the value of the whole equation range is (0,1]. Initially, the size of sub communities is small and $f_{MS\text{-}NC}$ changes mainly with the connection number of each node to its adjacent communities. With the growth of these sub-communities, the gaps between communities widening, and the link numbers being equal, the value of $f_{MS\text{-}NC}$ representing the intimacy of a node with a smaller community is higher, and the node is more easily partitioned into a small community. At the same time, adjusting the parameters λ and β can also adjust the proportion of the first and second item in the formula (4). The overall procedure is presented in TABLE 3.

TABLE 3: Modify the network based on node-to-community membership function $f_{MS\text{-}NC}$

Input: Node number n; Network representation after pretreatment and sub-communities integration $\boldsymbol{g}=[r_1,r_2,\ldots r_n]$, $r_k \in [1,n]$, $k=1,2,\ldots n$; Parameters λ, β; Iteration number *Iter*.
Output: Detection result $\boldsymbol{g}$.
Step1: for *loop*=1 to *Iter* do
Step2: for i=1 to n do
Step3: Find all the adjacent sub-communities of node i : $N_c=\{c_1,c_2,\ldots c_p\}$, where p is the number of sub-communities;
Step4: for j=1 to p do
Step5: $F_{MS\_NC}(j)$ <— Compute $f_{MS\text{-}NC}(i, c_t)$,where t=1,2,3…p;
Step6: end for
Step7: Attribution community index : i_{ms_c} <— $\arg\max_u (F_{MS\_NC}(u))$, u=1,2,…p (breaking ties randomly if more than one u's satisfy the condition) ;
Step8: Community label of node i: $\boldsymbol{g}(i)$ <— Community label of sub-community $\boldsymbol{g}(N_c(i_{ms\_c}))$;
Step9: end for
Step10: end for
Step11: $\boldsymbol{g}$ <—Decode($\boldsymbol{g}$);

Figure 4 shows the detection result on the Zachary Club network [41] after using our proposed strategy based on node-to-community membership function $f_{MS\text{-}NC}$. Fig. 4(a) shows the detection result of using only node search strategy and sub-communities integration, Fig. 4(b) represents the network modified by our proposed algorithm as summarized in TABLE 3.



(a) (b)

Fig.4. Partitioning result of Zachary's karate club. (a) Detection result of using only node search strategy and sub-communities integration. (b) Detection result using our proposed method with $f_{MS\text{-}NC}$.

As you can see from Figure 4, after modification, a number of misclassified nodes have been corrected. For example, the 33rd node, which has more connections with the 34th node, is wrongly divided into the same community as the 29th node. Once the 33rd node has been modified, the 19th node and 15th node are corrected successfully. Similarly, note that the 2nd node has a node degree of 9, while its number of connections with the community denoted as triangles is 5, which accounts for more than half of the total links. Thus the 2nd node is corrected by being assigned to the community of the 1st node. After that, the 22nd node, 3rd node, 14th node and 10th node are also modified. In addition for node correction, some potential small community structure has also been identified. For example, nodes 5, 6, 7, 11 and 17 form a closely connected small network. After correction, the value of Q also increases from 0.276 (shown in Fig. 4 (a)) to 0.419 (shown in Fig. 4 (b)).

3.5 Detecting overlapping communities using the node-to-community membership function $f_{MS\text{-}NC}$

In real networks, those nodes which belong to multiple communities are known as overlapping ones. Since the definition of $f_{MS\text{-}NC}$ implicates the membership grade of a node and its neighboring communities, we can assume that if the membership value between a node and several of its adjacent communities are the same, then this node can be regarded as an overlapping node. Additionally, as stated in Section 2, high quality prior knowledge of the non-overlapping community structure usually contributes to accurate detection results for overlapping communities. Therefore, our proposed algorithm makes use of the membership function introduced in Section 3.4 and mines overlapping nodes based on the non-overlapping communities obtained by the previous steps. The overall framework of the detection procedure is shown in TABLE 4.

TABLE 4: Overlapping community detection based on $f_{MS\text{-}NC.}$

Algorithm 3: Overlapping community detection based on $f_{MS\text{-}NC}$
Input: Number of nodes in network n; Detection results of non-overlapping community $\boldsymbol{g}=[r'_1, r'_2,\ldots, r'_n]$, where $r'_k \in [1,n]$, $k=1,2,\ldots n$; parameters λ, β; Max iteration number I_{ter1}.
Output: Overlapping node list ***Nod_ov***.
Step1: ***Nod_ov*** <— {};
Step2:for *loop*=1 to *Iter*1 do
Step3: for i=1 to n do
Step4: Find all the adjacent sub-communities of node i : $\boldsymbol{N_{c1}}=\{\boldsymbol{co_1, co_2,\ldots, co_p}\}$, where p is the number of sub-communities;
Step5: if not all the community label of sub-communities in N_{c1} are the same;
Step6: for j=1 to p do
Step7: $F_{MS\_NC}(j)$ <— Compute $f_{MS\text{-}NC}(i, \boldsymbol{co_t})$,where t=1,2,3…p;

Step6: end for

Step7: Membership index set $\boldsymbol{v}_{ov} \leftarrow \arg\max_{v} (F_{MS\_NC}(v))$, v=1,2,…p;

Step8: if $|\boldsymbol{v}_{ov}| > 1$

Step9: i is an overlapping node, hence ***Nod_ov***=***Nod_ov***$\cup\{i\}$;

Step10: else

Step11: i is a non-overlapping node and ***Nod_ov*** =***Nod_ov*** $\setminus \{i\}$ if i is in ***Nod_ov***

Step12: end if

Step13: node i belongs to community $\boldsymbol{co}_v$, namely $\boldsymbol{co}_v = \boldsymbol{co}_v \cup \{i\}$, v=1,2,…p;

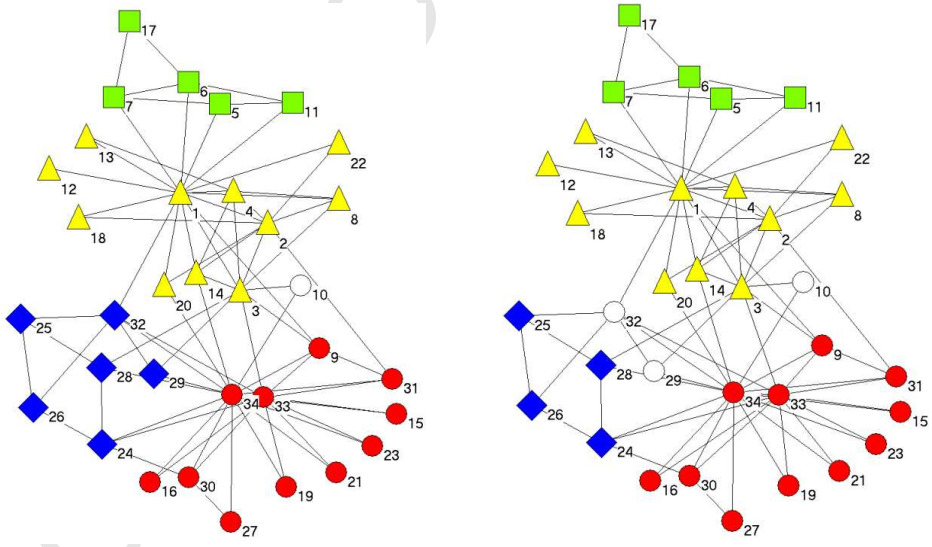
Step14: end if

Step15: end for

Step16: end for

Step17: $\boldsymbol{g} \leftarrow$Decode($\boldsymbol{g}$);

According to the procedure of TABLE 4, and using the Zachary's karate network obtained in Fig.4. (b) as an example, we set the parameters λ and β respectively equal to 1 and 0 or 0.2 and 1, with the corresponding overlapping community detection results shown in Figure 5 (a) and Figure 5 (b), in which the overlapping nodes are depicted in white. From Fig. 5 (a) it can be seen that when β=0, whether a node is overlapping is mainly decided by the number of connections between it and its adjacent communities. Thus only the 10th node satisfies the overlapping condition. In contrast, when β=0.2, after considering the acceptance degree between the community and its neighbors, a greater number of overlapping nodes are identified, as shown in Fig. 5 (b). Thus, we can obtain overlapping communities at different levels by adjusting these parameters.



(a) (b)

Fig.5. Overlapping community detection results on the Zachary's karate club network when (a) λ =1, β =0 (b) λ =0.2, β =1.

3.6 Overall framework of the proposed algorithm

According to the descriptions of Sections 3.1 to 3.5, the overall framework of our proposed algorithm is shown in Figure 6.

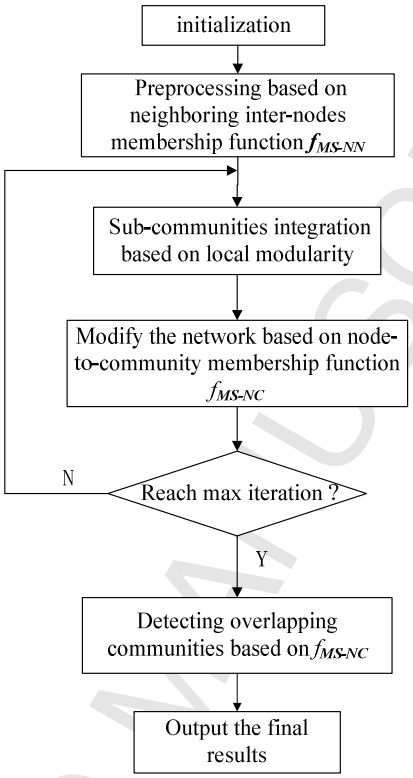


Fig.6. Overall flow chart for the proposed algorithm

3.7 Time complexity analysis of the proposed algorithm

In this section, we analyze the time complexity of the proposed algorithm. Supposing a network with n nodes and m edges, at the stage of the preprocessing introduced in Section 3.2, the membership between each node and its neighbors need to be calculated, with time complexity $O(m)$. The second stage needs $O(m\log n)$ time as stated in [28], and if the procedure runs for for l_1 iterations, the total time used in the second stage is $O(l_1 m\log n)$. The third stage, in which misclassified nodes are modified, has to consider the topological relationship between each node and its adjacent communities, thus it takes a time complexity of $O(k_c n)$ in each iteration, where k_c is the average number of communities that a node may be connected with. The time complexity of the third stage after l_2 iterations is $O(l_2 k_c n)$. The time used in searching overlapping communities is almost the same as that used in the third stage. Supposing it takes l_3 iterations for the second and the third stage to converge, thus the whole complexity of the proposed algorithm is $O(m)+l_3(O(l_1 m\log n)+ O(l_2\, k_c n))+ O(l_2\, k_c n)$. Since $l_1 \approx \log n$ [27], the overall time complexity is only $O(m\log^2 n)$.

4. Experimental results and analysis

This section presents and discusses the results of detecting both non-overlapping and overlapping communities in experiments performed on both artificial and real network examples.

4.1 Evaluation index

To test the detection results of networks whose true partitions are known, here a standard mutual information index (NMI) is introduced, defined as follows:

$$I(h_1,h_2)=\frac{-2\sum_{i=1}^{N_{h1}}\sum_{j=1}^{N_{h2}}H_{ij}\log\left(H_{ij}N/H_{i.}H_{.j}\right)}{\sum_{i=1}^{N_{h1}}H_{i.}\log(H_{i.}/N)+\sum_{j=1}^{N_{h2}}H_{.j}\log(H_{.j}/N)} \tag{5}$$

Here N_{h1} (N_{h2}) is the number of communities in the partition h_1 (h_2). $\boldsymbol{H}$ is the confusion matrix and its element $\boldsymbol{H}_{ij}$ is the number of nodes that belong to community i of partition h_1 that also belong to community j of partition h_2. The element $H_{i.}$ ($H_{.j}$) is the sum of the elements row i (column j) in matrix $\boldsymbol{H}$. When the partitioning results h_1 is the same with h_2, then $I(h_1, h_2)$ =1; otherwise, the larger the difference of the two partitions, the lower the value of $I(h_1, h_2)$. When they are completely opposite, $I(h_1, h_2)$ =0.

For other networks whose true partitions are unknown, modularity Q [23] is employed here as another index to test the detection results for non-overlapping communities. Its definition can be found in [23]. As to the evaluation of the overlapping nodes, Shen [42] et al. proposed a simple function EQ_{ov} for the evaluation of overlapping communities in unweighted and undirected networks. The definition of EQ_{ov} is as follows:

$$\mathrm{EQ}_{\mathrm{ov}}=\frac{1}{2m}\sum_{i,j}\frac{1}{O_iO_j}\left(A_{ij}-\frac{d_id_j}{2m}\right)\delta(c_i,c_j) \tag{6}$$

In the equation (6), A_{ij} represents the link number of nodes i and j. If i is connected with j, then A_{ij}=1; otherwise, A_{ij}=0. O_v represents the number of communities to which node v belongs, d_v represents the degree of node v, m represents the sum edges in the network. From equation (6) we can see that EQ_{ov}=Q if the network does not contain any overlapping nodes.

4.2 Setting of parameters

In the proposed algorithm, we introduced membership function $\boldsymbol{f_{MS\text{-}NN}}$ and node-to-community membership function $\boldsymbol{f_{MS\text{-}NN}}$, in which some parameters should be set. To obtain more accurate experimental results, some prior work has been done on several small networks whose ground truth partition results are known. As Fig.2 shows, different hierarchical network structures are found when parameter α changes from 0.5 to 1. Hence, to mining more multilayered structures, here we set α to 1. Another two parameters, λ and β, are flexible settings according to the detecting results of each network. That means we can get much better detection results through adjusting these two parameters.

4.3 Detection of non-overlapping communities

This section describes the algorithms employed for comparison, the artificial networks and real-world networks used in the experiment, and the corresponding analysis is given.

4.3.1 Algorithms for comparison

In order to fully demonstrate the effectiveness of the proposed algorithm, some representative algorithms such as GA algorithm [18], MODPSO algorithm [44], LPAm [26], LPAm+ [27], Infomap [36] and BGLL [30] (part of the code can be downloaded from [45]). In addition, in order to verify the effectiveness of each component of the proposed algorithm, we will combine the preprocessing strategy introduced in Section 3.2 with sub-communities integration introduced in Section 3.3 as the comparison algorithm, which is denoted as Pre-processing+BGLL.

4.3.2 Detection results on artificial networks

The first artificial network employed in our experiments is the extended GN benchmark networks, proposed by Lancichinetti et al. [43]. This network has 128 nodes, and is divided into 4 communities. μ is a parameter which represents the fraction of the number of links of each node within the community and the degree of the node. When the value of 1-μ becomes large, it suggests that the community structure of this network is much clearer, and can be more easily detected. Therefore, with increasing μ, the difficulty of detection is also increased. The key parameter values set for our algorithm in this experiment are: α =1, λ =0.15, β =1. Parameters in other algorithms are the same as those suggested in their corresponding publications. Figure 7 shows the best results over 30 runs on extended GN benchmark networks for the detecting of non-overlapping communities.

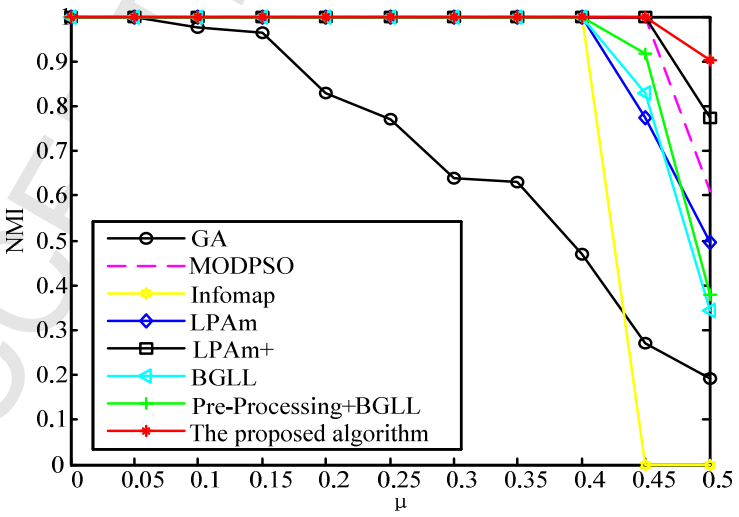


Fig.7. Best results over 30 runs on extended GN benchmark networks.

As we can see from Figure 7, our proposed algorithm clearly generates the best testing results. When $\mu \leq 0.4$, apart from GA algorithm, all other algorithms were able to obtain completely accurate results, but when μ=0.45, Infomap is unable to detect the community structure, only LPAm+, MODPSO and the proposed algorithm can get the true partition results, and the value of NMI obtained by other detection algorithms has declined. Meanwhile, because the Pre-Processing +BGLL algorithm makes a preparatory division of network based on the function $f_{MS\text{-}NN}$, thus it generates more accurate results compared to using only the BGLL algorithm. When μ=0.5, the proposed algorithm generates the closest results to the true partition (NMI value is close to 0.9).

Another set of artificial networks is the LFR benchmark networks [43]. Compared to the extended GN benchmark networks, LFR benchmark networks has more adjustable parameters, which control the number of nodes generated, the size of communities and the degree of nodes. For our experiments, the parameters chosen in the LFR benchmark networks are as follows: network node number n=1000, average node degree is 20, maximum node degree is 50, the degree distribution exponents are $\tau1$=2, $\tau2$=1. In this experiment, parameter μ changes from 0 to 0.7 and 17 network are generated. Figure 8 shows the best results over 30 runs on LFR benchmark networks.

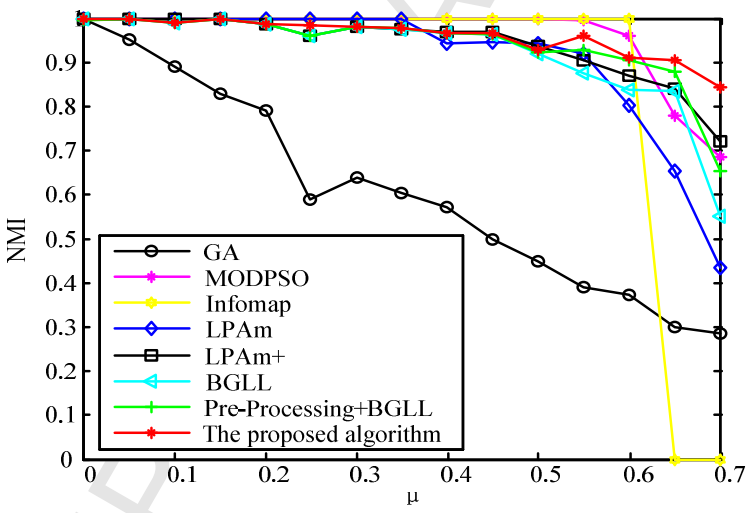


Fig.8. Best results over 30 runs on the LFR networks

It can be seen form Figure 8 that the detection results obtained by the proposed algorithm are not completely optimal and perform very slightly less well than some of the comparison methods (e.g. LPAm and Infomap) for relatively easy detection problems. However, as μ increases, the results remain relatively stable and it obtains the best results when μ is greater than 0.65. In contrast, Infomap produces a completely correct partition of the networks only when μ<0.65. Similarly, the value of NMI obtained by MODPSO declines after μ=0.6. The results obtained by Pre-Processing + BGLL outperform those obtained by BGLL alone, which indicates the effectiveness of the preprocessing strategy proposed in this paper.

4.3.3 Detection results on real-world networks

In this section 9 real-world networks are tested, and their important attributes are as shown in TABLE 5:

TABLE 5: Information of real-world networks

Network	node number	edge number	average degree	Reference
Zachary's karate (N1)	34	78	4.59	[46]
dolphins (N2)	62	159	5.13	[47]
American football (N3)	115	613	10.66	[15]
elegans (N4)	453	2025	8.94	[48]
netscience(N5)	1589	2742	3.45	[49]
power (N6)	4941	6594	2.67	[50]
PGP (N7)	10680	24340	4.55	[51]
Internet (N8)	22963	48436	4.22	[52]
Enron(N9)	36692	367662	20.04	[53]

All algorithms are run 30 times on the 9 real-world networks, and their best results and average results are shown in Table 6 (for conciseness the Pre-Processing+BGLL algorithm is abbreviated as Pre_BGLL):

TABLE 6: The results of all the algorithms run 30 times on 9 real-world networks (the symbol "—" indicates that the algorithm cannot effectively detect communities within the networks).

network	Index	GA	MODPSO	Infomap	LPAm	LPAm+	BGLL	Pre_BGLL	proposed
N1	Q_{max}	**0.4198**	**0.4198**	0.402	0.406	**0.4198**	**0.4188**	0.3949	**0.4198**
	Q_{avg}	0.411	**0.4186**	0.402	0.384	0.4176	0.4172	0.3894	0.4181
N2	Q_{max}	0.5238	0.5268	**0.5285**	0.511	**0.5285**	0.520	**0.5285**	0.5276
	Q_{avg}	0.5138	**0.5249**	**0.5285**	0.501	0.5240	0.518	0.5202	0.5244
N3	Q_{max}	0.5683	**0.6046**	0.6005	0.6044	**0.6046**	0.6044	0.6044	**0.6046**
	Q_{avg}	0.5021	0.6035	0.6005	0.5814	**0.6038**	**0.6036**	0.6032	0.6035
N4	Q_{max}	0.2832	0.3585	0.4168	0.3999	0.450	0.434	0.4156	**0.4505**
	Q_{avg}	0.2732	0.3566	0.4168	0.3796	0.440	0.432	0.4074	**0.4417**
N5	Q_{max}	0.8979	0.9501	0.931	0.8471	0.9513	**0.9517**	0.9481	**0.9579**
	Q_{avg}	0.8581	0.950	0.931	0.8363	0.9436	**0.9504**	0.935	**0.9549**
N6	Q_{max}	0.666	0.8422	0.8298	0.6121	0.9302	0.9349	**0.9363**	**0.9382**
	Q_{avg}	0.6354	0.8385	0.8298	0.6055	0.9289	0.9341	**0.9351**	**0.9366**
N7	Q_{max}	0.645	0.335	0.8135	0.7222	0.8643	**0.8822**	0.8799	**0.8831**
	Q_{avg}	0.604	0.328	0.8135	0.7124	0.8632	**0.8817**	0.8787	**0.8820**
N8	Q_{max}	0.3912	——	0.5755	0.4748	0.6500	0.6608	**0.6668**	**0.6756**
	Q_{avg}	0.3850	——	0.5755	0.4669	0.6381	0.6597	**0.6644**	**0.6742**
N9	Q_{max}	0.1071	——	0.2584	0.2450	0.2716	0.2741	**0.2762**	**0.2780**
	Q_{avg}	0.1068	——	0.2584	0.2297	0.2663	0.2724	**0.2728**	**0.2769**

We can see from TABLE 6 that GA and LPAm struggle to detect useful results, even when the scale of networks is relatively small. The results obtained by Infomap are relatively stable, but it can only achieve the best results in few of the example networks. As the size of the test networks increases, the performance of MODPSO (also based on an evolutionary algorithm) is greatly

improved compared to GA, but fails on several large scale networks. LPAm+ which is based on LPAm using a sub-communities integration strategy, overcomes the vulnerability of LPAm to local optima, and thus generates superior results than those achieved by LPAm. In contrast to LPAm+ (which is based on global sub-communities and multistep greedy integration), BGLL uses integration strategy based on local sub-communities and thereby achieves better results in some large and medium-sized networks as shown in TABLE 6. Pre-BGLL denotes our proposed pre-processing strategy (building on BGLL) which considers the intimacy between each node and its neighbors. TABLE 6 shows that the accuracy of results obtained by Pre-BGLL is improved compared with that obtained by the unmodified BGLL when testing on large scale networks. Furthermore, the proposed algorithm employs the node modification strategy based on the Pre-BGLL algorithm and the accuracy of its detection results is therefore further improved. Thus the proposed algorithm achieves the best detection results when tested on the majority of these benchmark networks.

4.4 Detection of overlapping communities

For detection of overlapping communities, we compare COPRA [33], CFinder [54], CONGA [55], as well as a recent algorithm proposed by Li [58] (Li's Alg), as well as our proposed algorithm on the 9 real-world benchmark networks. COPRA is based on the LPA algorithm, and is suitable for large-scale overlapping community detection. CFinder is a k-clique percolation algorithm, in which a node can belong to multiple k factions, thus achieving the detection of overlapping nodes. The CONGA algorithm is based on the well-known GN algorithm [15], joining the node splitting strategy to make sure that nodes can be accepted to multiple communities. The source code of these algorithms can be obtained from [56]. Li proposed two noble algorithms for the detecting of overlapping communities [58][63]. In paper [58], he employs depth and breadth searching to extract the maximal cliques and then merge sub-graphs according to rules. Through these steps, overlapping nodes can be found and satisfactory results are obtained. Another firstly extracted all the seed communities and absorbed more community members using the absorbing degree function. As this algorithm mining overlapping nodes in weighted networks, which is different from ours, hence here we only take the former one for comparison.

Figure 9 shows the average detection results on overlapping networks over 30 runs within 2 hours. It can be seen that CONGA can hardly detect community structures effectively when parameter μ increase to 0.25. The remaining algorithms, like Cfinder and COPRA can find relative better results, but with the increase of μ, these two algorithms can hardly get satisfying detecting results. Li's Alg can obtain higher value of EQ_{ov} with the increase of μ, but it is not the most efficient one. From Fig.9 we can conclude that the proposed algorithm can effectively mining community structures compared with other algorithms.

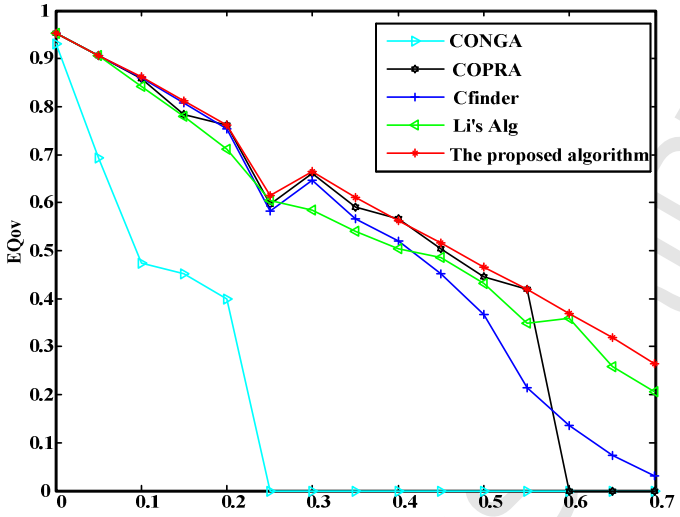
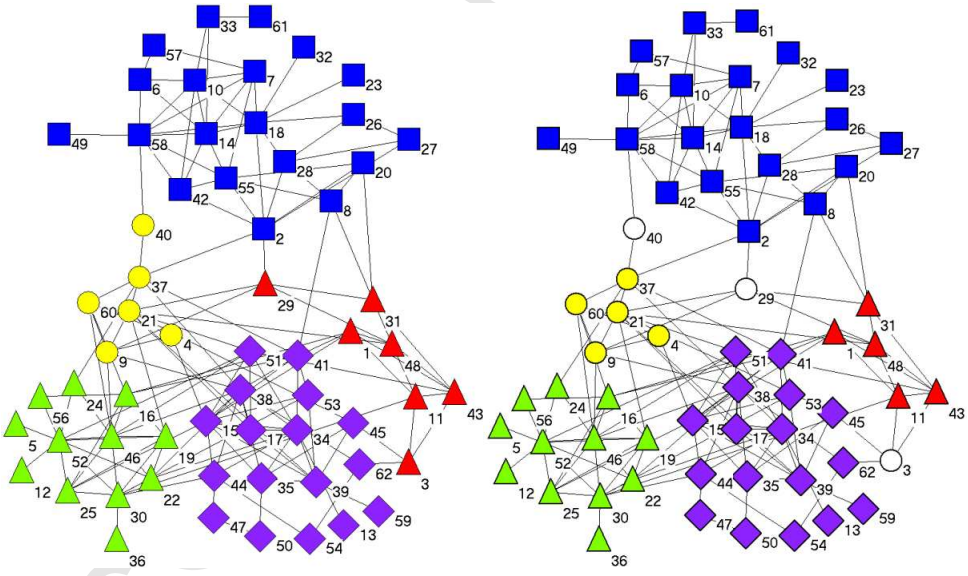


Fig.9. Average detection results on overlapping networks over 30 runs within 2 hours

Figure 10 shows the overlapping nodes detected in a single run on the dolphin network. Figure 9(a) shows the non-overlapping community structures obtained by the proposed algorithm. The color notation shows how the network has been divided into five distinct communities, and the triangles and squares respectively represent the two communities in the ground-truth division of the network. Figure 10(b) shows the detection of overlapping communities with white circles denoting the overlapping nodes.



(a) (b)

Fig.10. Detection results on dolphin network when parameters are set as α =0.7, λ =0.15, β =1. (a) Detection results of non-overlapping communities. (b) Detection results of overlapping communities.

For the detection of overlapping nodes in the dolphin network, the parameters were set as α=0.7, λ=0.15, β=1. As shown in Fig. 10(a), the proposed algorithm divides the single ground-truth triangle node into a number of small communities during the non-overlapping community detection stage, making the network a multi-level structure. On this basis, the 40th node, 29th node and 3rd node are detected as overlapping nodes, because they connect with several different communities which all share the same value of $f_{MS\text{-}NC}$ with them.

In the following experiment, we set the parameters as λ = 0.15 and β =1, the evaluation index is EQ_{ov} introduced in the Section 4.1. Table 7 shows the average value of EQ_{ov} over 30 runs in the 9 real-world networks.

TABLE 7: Each algorithm run 30 times in the real network, the average EQ_{ov} value are as below ("—" indicates that the algorithm cannot effectively detect the overlapping nodes).

algorithm	N1	N2	N3	N4	N5	N6	N7	N8	N9
CONGA	0.278	0.3808	0.3372	0.1695	**0.9506**	**0.9170**	0.4916	——	——
COPRA	0.2576	0.3258	0.5934	0.3233	0.8464	0.75	0.6710	0.0914	0.315
CFinder	0.1858	0.3612	0.5593	0.0957	0.5905	0.1577	0.3788	0.0149	——
Li's Alg	**0.3848**	**0.5077**	**0.5946**	**0.4024**	0.8460	0.8712	**0.8694**	**0.457**	**0.4623**
Proposed	**0.4053**	**0.5238**	**0.5987**	**0.4349**	**0.9541**	**0.9362**	**0.8826**	**0.6621**	**0.6019**

As shown in Table 7, the overlapping community detection results of the proposed algorithm on the 9 real-world networks are significantly better than the other three algorithms. As CFinder needs to extract the maximum complete sub-graphs in each run, the running time is too long to detect community structure in larger networks, and its results are affected by the parameter k in the algorithm, so the value of EQ_{ov} is low. CONGA in some networks, such as the netscience network (N5) and power network (N6) has good detection results. However, CONGA is also unable to detect the overlapping community structure of the last two networks as its time complexity is $O(m^3)$. Although the COPRA algorithm has lower time complexity, and it can accomplish the detection of all the networks, its detection results are not optimal. Li's algorithm utilizes depth and breadth searching methods to extract the maximal cliques which is time-saving, enabling it to discover overlapping nodes in some large-scale networks effectively. However, it cannot achieve the best values of EQ_{ov} in all the networks. The results suggest that our proposed algorithm can effectively detect the overlapping nodes in large and medium-scale networks.

5. Conclusions

In this paper we have proposed a large-scale community detection algorithm based on node membership grade and sub-communities integration. Firstly, considering the relationship between each node and its adjacent nodes, we proposed a neighboring inter-nodes membership function $f_{MS\text{-}NN}$ to extract sub-communities, thus providing fast preprocessing of the network. Next, after

merging these sub-communities based on local modularity, we introduced another node-to-community membership function $f_{MS\text{-}NC}$ to modify any misclassified nodes, preventing convergence on local optima. Additionally, by adjusting the parameters of function $f_{MS\text{-}NC}$, multilevel overlapping communities of high quality can be detected on the basis of the non-overlapping community structures obtained by the proposed algorithm. The experimental results demonstrate that, through the effective combination of the strategies of local node search and sub-communities integration, as well as node correction, the algorithm can not only accurately detect non-overlapping communities, but can also effectively mine the overlapping communities in large and medium scale networks. In addition, the algorithm relies mainly on only the local information of each node, which contributes to a relatively low time complexity ($O(m\log^2 n)$), making this method suitable for community detection in large scale networks.

In future research, we will focus on the detection problem in networks with larger scale, such as networks with hundreds of thousands, or even millions nodes, and endeavor to further improve the detection accuracy while preserving low time complexity, so that the algorithm can detect community structures efficiently.

Acknowledgement

We would like to express our sincere appreciation to the anonymous reviewers for their valuable comments, which have greatly helped us in improving the quality of the paper. This work was partially supported by the National Basic Research Program (973 Program) of China under Grant 2013CB329402, the National Natural Science Foundation of China, under Grants 61371201, 61203303, 61272279, and 61373111, the EU FP7 project (grant no. 247619) on “NICaiA: Nature Inspired Computation and its Applications”, the Fund for Foreign Scholars in University Research and Teaching Programs (the 111 Project) under Grant B07048, and the Program for Cheung Kong Scholars and Innovative Research Team in University under Grant IRT1170.

References

[1] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, A. Arenas, Models of social networks based on social distance attachment, Phys. Rev. E 70 (5) (2004) 056122.

[2] Z.W. Liu, H.G. Zhang, Q.L. Zhang, Novel stability analysis for recurrent neural networks with multiple delays via line integral-type L–K functional, IEEE Trans. Neural Netw. 21 (11) (2010) 1710-1718.

[3] F. Képès, Biological Networks, World Scientific, Singapore, 2007.

[4] T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, Y. Sakaki, A comprehensive two-hybrid analysis to explore the yeast–protein interaction, Proc. Natl.Acad. Sci. USA 98 (8) (2001) 4569-4574.

[5] Y. J. Liu, C.L.P. Chen, G.X. Wen, S.C. Tong, Adaptive neural output feedback tracking control for a class of uncertain iscrete-time nonlinear systems, IEEE Trans. Neural Netw. 22 (7) (2011) 1162-1167.

[6] K.A. Eriksen, I. Simonsen, S. Maslov, K. Sneppen, Modularity and extreme edges of the internet, Phys. Rev. Lett. 90 (14) (2003) 148701.

[7] A. Vazquez, R. Pastor-Satorras, A. Vespignani, Large-scale topological and dynamical properties of the Internet, Phys. Rev. E 65 (6) (2002) 066130.

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

[9] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (3–5) (2010) 75-174.

[10] M. G. Gong, J. Liu, L.J. Ma, Q. Cai, L.C. Jiao, Novel heuristic density-based method for community detection in networks, Physica A 403(2014) 71-84.

[11] C. Shi, Z. Yan, Y. Cai, B. Wu, Multi-objective community detection in complex networks, Applied Soft Computing 12 (2012) 850-859.

[12] B.W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, Bell Syst. Tech. J. 49 (2) (1970) 291-307.

[13] M. Fiedler, Algebraic connectivity of graphs, Czechoslovak Math. J. 23 (98) (1973) 298-305.

[14] A. Pothen, H.D. Simon, K.P. Liou, Partitioning sparse matrices with eigenvectors of graphs, SIAM. J. Matrix Anal. Appl. 11 (3) (1990) 430-452.

[15] M. Girvan, M. E. J. Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences of the USA 99 (2002) 7821-7826.

[16] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, D. Parisi, Defining and identifying communities in networks, Proc. Natl. Acad. Sci. USA 101 (9) (2004) 2658-2663.

[17] S. Fortunato, V. Latora, M. Marchiori, A method to find community structures based on information centrality, Phys. Rev. E 70 (2004) 056104.

[18] C. Pizzuti, Ga-net: a genetic algorithm for community detection in social networks, in: Parallel Problem Solving from Nature C PPSN X, in: Lect. Note Comput. Sc., vol. 5199, Springer, Berlin, Heidelberg, 2008, pp. 1081-1090.

[19] C. Pizzuti, A multi-objective genetic algorithm for community detection in networks, in: Proceedings of the 21st IEEE International Conference on Tools with Artificial Intelligence, Newark, New Jersey, USA, 2009, pp. 379-386.

[20] M. G. Gong, B. Fu, L. C. Jiao, and H. F. Du, Memetic algorithm for community detection in networks, Phys. Rev. E 00 (2011) 006100.

[21] R. H. Shang, J. Bai, L. Jiao, C. Jin, Community detection based on modularity and an improved genetic algorithm, Physica A 392 (2013) 1215-1231.

[22] Q. Chen∗, T. T. Wu, M. Fang, Detecting local community structures in complex networks based on local degree central nodes, Physica A 392 (2013) 529-537.

[23] M. E. J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2004) 026113.

[24] X. Wang∗, J. Li, Detecting communities by the core-vertex and intimate degree in complex networks, Physica A 392 (2013) 2555-2563.

[25] U. N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (2007) 036106.

[26] M. J. Barber, J.W. Clark, Detecting network communities by propagating labels under constraints, Physical Review E 80 (2) (2009) 026129.

[27] X. Liu, T. Murata. Advanced modularity-specialized label propagation algorithm for detecting communities in networks. Physica A, 2010, 389(7):1493-1500.

[28] P. Schuetz, A. Caflisch, Efficient modularity optimization by multistep greedy algorithm and vertex refinement, Phys. Rev. E 77 (2008) 046112.

[29] A. Clauset, M.E.J. Newman, C. Moore, Finding community structure in very large networks, Physical Review E 70 (6) (2004) 066111.

[30] V. Blondel, J. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. 2008 (2008) P1000.

[31] R. Rotta, A. Noack. Multilevel local search algorithms for modularity clustering. Journal of Experimental Algorithmics, (2011).16(2), article 2.3.

[32] A. Mohsen, A. Mohsen. Community detection in social networks using hybrid merging of sub-communities, Journal of Network and Computer Applications 40 (2014) 73-84.

[33] S. Gregory. Finding overlapping communities in networks by label propagation, New Journal of Physics 12 (2010) 103018.

[34] A. Lancichinetti, S. Fortunato, J. Kertész. Detecting the overlapping and hierarchical community structure in complex networks, New Journal of Physics 11 (2009) 033015.

[35] Z. H. Wu, Y. F. Lin, Efficient overlapping community detection in huge real-world networks, Physica A 391 (2012) 2475-249.

[36] M. Rosvall, C.T. Bergstrom. Maps of random walks on complex networks reveal community structure, Proceedings of the National Academy of Sciences of the United States of America 105 (4) (2008) 1118-1123.

[37] D. Lai, H. Lu, C. Nardini. Enhanced modularity-based community detection by random walk network preprocessing, Physical Review E 81 (6) (2010) 066118.

[38] C. H. Mu, Y. Liu, Y. Liu, J. S. Wu, L. C. Jiao, Two-stage algorithm using influence coefficient for detecting the hierarchical, non-overlapping and overlapping community structure, Physica A 408 (2014) 47-61.

[39] B. Yan, S. Gregory, Detecting community structure in networks using edge prediction methods, Journal of Statistical Mechanics: Theory and Experiment (2012) P09008.

[40] P. De Meo, E. Ferrara, G. Fiumara, A. Provetti. Enhancing community detection using a network weighting strategy, Inf. Sci. 222 (2013) 648-668.

[41] W. Zachary, An information flow model for conflict and fission in small groups, Journal of Anthropological Research 33(1977) 452-473.

[42] H. Shen, X. Cheng, K. Cai, M.B. Hu. Detect overlapping and hierarchical community structure, Physica A 388 (2009) 1706.

[43] A. Lancichinetti, S. Fortunato, F. Radicchi. Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (2008) 046110.

[44] M. G. Gong, Q. Cai, X. Chen, L. Ma, Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition, IEEE Transactions on Evolutionary Computation 2014, 18(1): 82-97.

[45] L. Ma, M. G. Gong, J. Liu, Q. Cai, L.C. Jiao, Multi-level learning based memetic algorithm for community detection, Applied Soft Computing 19 (2014) 121-133.

[46] W. Zachary. An information flow model for conflict and fission in small groups, Journal of Anthropological Research 33(1977) 452-473.

[47] D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, S.M. Dawson. The bottlenose dolphin community of Doubtful Sound features a large Proportion of long-lasting associations, Behavioral Ecologyand Sociobiology 54 (2003) 396-405.

[48] J. Duch, A. Arenas, Community detection in complex networks using extremal optimization, Physical Review E 72 (2) (2005) 027104.

[49] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices, Phys. Rev. E, vol. 74, no. 3, p. 036104, Sep.2006.

[50] D. J. Watts, S.H. Strogatz. Collective dynamics of 'small-world' networks, Nature 393 (6684) (1998) 440-442.

[51] M. Boguna, R. Pastor-Satorras, A. Díaz-Guilera, A. Arenas. Models of social networks based on social distance attachment, Physical Review E 70 (5) (2004) 056112.

[52] M. E. J. Newman, Network data, 2013, available at http:// www-personal. umich.edu/ mejn/ netdata/.

[53] J. Leskovec, J. Kleinberg, C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations, in: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, pp. 177-187.

[54] G. Palla, I. Derényi, I. Farkas, T. Vicsek, Uncovering the overlapping community structure of complex networks in nature and society, Nature 435 (2005) 814-818.

[55] S. Gregory. A fast algorithm to find overlapping communities in networks. Machine Learning and Knowledge Discovery in Databases, 2008, 408-423.

[56] www.cs.bris.ac.uk/~steve/networks/.

[57] Y. Z. Cui, X. Y. Wang, J. Q. Li, Detecting overlapping communities in networks using the maximal sub-graph and the clustering coefficient, Physica A 405 (2014) 85-91.

[58] J. Q. Li, X. Y. Wang, Y. Z. Cui, Uncovering the overlapping community structure of complex networks by maximal cliques, Physica A 415 (2014) 398-406.

[59] Y. Z. Cui, X. Y. Wang, J. Eustace. Detecting community structure via the maximal sub-graphs and belonging degrees in complex networks, Physica A 416 (2014) 198-207.

[60] Y. Z. Cui, X. Y. Wang, Uncovering overlapping community structures by the key bi-community and intimate degree in bipartite networks, Physica A 407 (2014) 7-14.

[61] L. Y. Lü, T. Zhou, Link prediction in complex networks: A survey, Physica A 390 (2011) 1150-1170.

[62] L. Hamers, Y. Hemeryck, G. Herweyers, M. Janssen, H. Keters, R. Rousseau, A. Vanhoutte Similarity measures in scientometric research—the jaccard index versus salton cosine formula. Information Processing & Management 25 (1989) 315-318.

[63] J. Q. Li, X. Y. Wang, J. Eustace, Detecting overlapping communities by seed community in weighted complex networks, Physica A 392(23) (2013) 6125-6134.