International Journal of Machine Learning and Cybernetics (2024) 15:2131–2147
https://doi.org/10.1007/s13042-023-02020-3

ORIGINAL ARTICLE

A novel overlapping community detection strategy based on Core-Bridge seeds

Gaolin Chen[1] · **Shuming Zhou**[2,3,4]

Received: 27 July 2022 / Accepted: 23 October 2023 / Published online: 4 December 2023

Abstract
The last decade has witnessed the advance of overlapping community detection based on local expansion. In this paper, we propose a novel local expanding-based overlapping community detection algorithm, denoted by Core and Bridge Seeds Extension, that aims to improve the quality of communities. Instead of the traditional approaches to select the cores of communities as seeds, a new Core-Bridge triplet strategy is suggested to select seeds to generate the initial backbone and framework of the community. In the optimization stage, a stepwise refinement approach is adopted to solve the issue of unreasonable division and unassigned node allocation. A merge index is designed to merge communities reasonably. The comparisons about the methods to improve accuracy of community numbers based on the known algorithms are also presented. Experimental results on synthetic networks and real networks show that our strategy outperforms the state-of-art algorithms in stability and effectiveness.

Keywords Social network · Overlapping community detection · Core-Bridge · Seed selection · Community optimization · Community merge

1 Introduction

Community detection of social networks has attracted immense attention over last decades due to its enormous applications in different domains. Community is the one of the most important structure characteristics of social networks, and community detection is a fundamental task in network analysis with applications in functional prediction of protein network [1], social network [2], web network [3], personalized recommendation system [4] among others. Therefore, how to identify communities effectively is of great significance to master and command networks.

Since one social network can be modelled as a graph with nodes and edges, communities are clustered into groups of nodes with higher probability of being connected to each other than that of being connected to nodes of distinct groups [5]. Traditional community detection strategy is to partition a network such that every node belongs to exactly one cluster, usually named non-overlapping community detection [6–10]. However, nodes ordinarily participate in multiple communities, which is more consistent with actual situations. For example, the clustering can be based on an individual's interests, while a person's interests may have more aspects. The problem of overlapping community detection is to find communities, which are allowed to overlap and some nodes are allowed to belong to more than one cluster. A plenty of studies have been conducted on this issue [11–18]. Existing overlapping algorithms involve clique percolation [11, 19, 20], improved label propagation [13, 15, 18, 21], and edge partition [16]. Although these methods have put forward some effective strategies, some of them suffer from either low accuracy or high computation complexity.

With the successive increase of network scale, more efficient overlapping community detection methods have been proposed [22–24]. Local expansion methods [14, 25–28]

✉ Shuming Zhou
zhoushuming@fjnu.edu.cn

Gaolin Chen
gaolinchen@fjnu.edu.cn

1 School of Computer and Cyber Security, Fujian Normal University, XueFu South Street, Fuzhou 350117, China

2 School of Mathematics and Statistics, Fujian Normal University, XueFu South Street, Fuzhou 350117, China

3 Center for Applied Mathematics of Fujian Province, Fujian Normal University, XueFu South Street, Fuzhou 350117, China

4 Key Laboratory of Analytical Mathematics and Applications (Ministry of Education), Fujian Normal University, XueFu South Street, Fuzhou 350117, China

have made great progress due to their efficiency for mining overlapping communities with high quality. Lancichinetti et al. [14] proposed the fitness function for local optimization. Whang et al. [17] suggested two completely different seed strategies to find overlapping communities. Gao et al. [29, 30] applied the difference of conductance to locate nodes accurately, and improved the seed selection and expansion to increase accuracy subsequently. Basuchowdhuri et al. [31] employed community nodes and broker nodes for traversal-based expansion. Ding et al. [32, 33] exploited information contained in the neighborhood of nodes or communities to define a similarity such that the repeatedly removal of nodes with low similarity ensures the quality of the community. Cheng et al. [34] provided a local connectedness strength to correct node's attribution and improve partition performance. Jiang et al. [35] proposed a method to detect the community by the central node-based link prediction strategy. For large-scale networks, overlapping community detection usually swings around in the community quality and efficiency, and it is always difficult to achieve a balance. Therefore, it is worthwhile detecting overlapping communities with high efficiency and accuracy in large-scale networks.

Based on previous idea, in this paper, we propose a novel overlapping community detection algorithm based on local expansion, denoted by *CBSE* (Core and Bridge Seeds Extension), to improve community quality. The motivation of our work is to make the communities obtained with higher aggregation by improving the performance of each stage, while ensuring efficiency. The main contributions of the paper are summarized as follows.

1. In order to improve the accuracy of seeds, we employ the roles of the core and the bridge of nodes. These two types of nodes play distinct role in constructing the framework and backbone of communities.
2. In community optimization stage, we improve the quality of the community. According to the change of conductance, nodes with low quality in the community are eliminated. Then, the unassigned nodes are allocated to different communities by shift-in (denoted as *SI*) and final allocation (denoted as *FA*) strategies to ensure quality.
3. In community merge stage, when all nodes are partitioned, communities are merged based on the merge index (denoted as *MI*) to reduce community inclusion and overlapping degree of communities, so to improve effectiveness.

The rest of this paper is organized as follows. Relevant works are outlined in Sect. 2. Section 3 proposes the novel algorithm and explains the stages in detail. Section 4 empirically analyzes the experimental results, and Sect. 5 further optimizes the approach to improve quality. Section 6 concludes the work.

2 Related work

A large volume of previous works have attempted to address the overlapping community detection issue in social networks, while the local expansion methods has developed rapidly.

2.1 Overlapping community detection

To address overlapping community detection problem, the clique percolation method (*CPM*) [11] combines two k-cliques with $k-1$ common neighbors to form a community. Cliques belonging to different communities may share nodes to realize community overlapping [19, 20]. Xie et al. [13] proposed an overlapping community algorithm based on label propagation (*SLPA*), which records the historical candidate labels of nodes during the propagation process, and selects the label with the most occurrences to update the target node label. Another variant of label propagation algorithm, *COPRA* [12], added a label list of length l to each node. By updating labels with the membership, nodes are divided into different communities such that a node belongs to l communities. Ahn et al. [16] developed an efficient link partitioning algorithm, called *LC*, to convert overlapping community detection to non-overlapping community detection.

2.2 Local expansion methods

To deal with larger-scale networks, some scholars attempt to use local information to partition communities. Among them, the most notable one is to mine communities based on local expansion. Lancichinetti et al. [14] proposed a method, named *LFM* (Local Fitness Maximization), to search only those nodes that have not been classified, in which the quality of communities is determined by the parameter in fitness function. Padrol-Sureda et al. [36] added or removed nodes from the community in terms of the maximum gain of the fitness function. Whang et al. [17, 37] established a framework of local expansion, including removing whiskers, determining seeds, and then expanding communities by virtue of personalized PageRank vectors [38]. Wang et al. [39] clustered and expanded seeds based on neighborhood information.

3 Our proposed method

3.1 Motivation

The stages in community detection based on local extension are relatively independent. We expect to improve performance at each stage for the purpose of improving the quality of communities.

First, seed selection strategy greatly determines the quality of subsequent community expansion. Selecting seeds randomly is of a great risk. There exists the possibility of error accumulation, which leads to unstable results. The goal of the selection strategy based on node importance is to choose the core of a community. However, in practice, the higher-importance node is not necessarily the centre of the community, but sometimes acts as a bridge node to connect two communities. Therefore, most of the seed selection methods have the premise that the found seeds are in the center of the community. Sometimes, if the pseudo-centres, which are actually bridges with similar features, are selected, mistakes in community mining may occur. It is urgent to find seeds in a more scientific way to improve the quality of seeds. In fact, the node role as the core or the bridge is often difficult to be distinguished perfectly because some nodes have both roles. For example, node 2 (see Fig. 1a) in Karate network is both a high-degree node in the central area and a bridge to connect two communities. This observation prompts us to apply a new strategy to choose the seed set.

Next, when seeds have been locally expanded, communities may present the different situations, such as the problem that the community is too large or too small, which needs to be further optimized. We employ a stepwise refinement for community optimization. The purpose of this approach is to correct previous mistake of classification, divide the remaining nodes properly, and improve the quality of communities constantly. Overall, our goal for community optimization is to ensure that the division is stable and robust.

3.2 Preliminaries

An unweighted and undirected network is represented by a graph, which is denoted by $G = (V, E)$, where V denotes node set and E denotes edge set. We denote the number of nodes in G by $|G|$, the number of edges in G by $|E(G)|$. The overlapping community detection is formally defined as follows.

Given a network G, the objective is to pursue a high-quality community partition (usually called a cover) $Cover = \{C_1, C_2, \ldots, C_l\}$ that maximizes (or minimizes) an evaluating metric (such as EQ, $ONMI$ and so on.), which will be introduced in Sect. 4. We denote by C_i a community ($1 \le i \le l$) that satisfies the following constraints: $\emptyset \neq C_i \subsetneq V (1 \le i \le l)$, $C_1 \cup C_2 \cup \ldots \cup C_l = V$, and $C_i \cap C_j \neq \emptyset$ for some $i \neq j$.

3.3 Algorithm

The algorithm adopts the basic framework of local expansion, including three stages:

1. Seed initialization: we choose high-quality initial seeds by leveraging the core and the bridge.
2. Local expansion: the first expansion of seeds is performed by virtue of the strong community, and then the expansion is carried out by the help of the node's PageRank vector.
3. Community optimization: low-quality nodes are eliminated, and two strategies *SI* and *FA* are used to divide unallocated nodes, and finally the communities are merged by the merge index.

Algorithm 1 presents the main procedure of our proposed strategy.

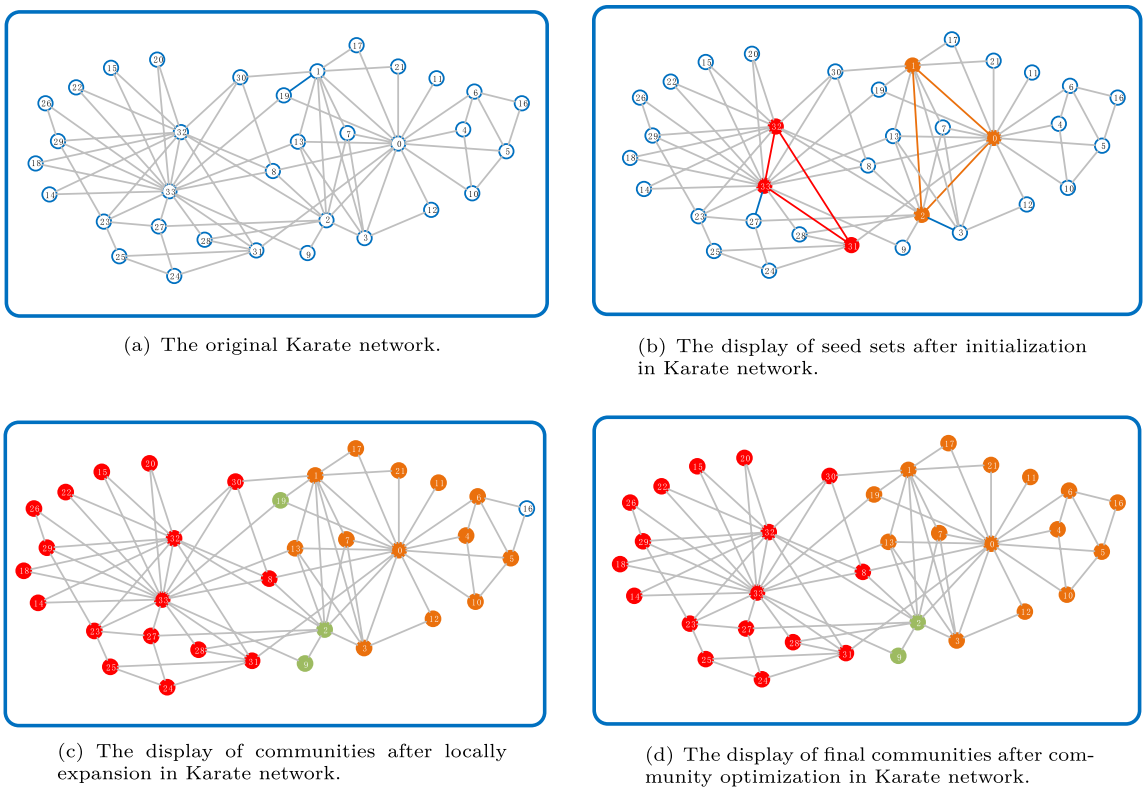
3.3.1 Seed initialization

Seed selection strategy determines the accuracy of community detection. A seed can be a node, an edge, or a set of jointed or disjointed nodes. Lancichinetti et al. [14] randomly selected a node, which does not belong to any known community, as a seed. Baumes et al. [25] selected one random edge as a seed, known as iterative scan (*IS*) algorithm. Shen et al. [40] used the maximal clique as seeds with more expensive complexity. Su et al. [41] chose three nodes with local maximal degree as seeds. In addition, some seed selection algorithms allow overlap between seeds. But if two seeds are highly overlapping, there is a great chance that they will expand into the same or highly similar community.

Therefore, we want to select seeds that are of high quality and do not overlap each other. In a community, cores

Input: The network $G = (V, E)$.
Output: The overlapping community set $Cover$.
1: $Cover \leftarrow \emptyset$;
2: Seed set $S \leftarrow$ execute Algorithm 2;
3: Local Cover $LCover \leftarrow$ execute Algorithm 3;
4: Final Cover $Cover \leftarrow$ execute Algorithm 4;
5: **return** $Cover$;

Algorithm 1 The framework



(a) The original Karate network.

(b) The display of seed sets after initialization in Karate network.

(c) The display of communities after locally expansion in Karate network.

(d) The display of final communities after community optimization in Karate network.

Fig. 1 The process of overlapping community detection in Karate network. Blue hollow dots represent unassigned nodes, red and orange solid dots are community nodes. Different colors represent different communities. The green solid dots are overlapping nodes (color figure online)

and bridges play prominent roles. In the communication of the network, bridges act as a hub between multiple communities, and play a pivotal role in boundary generation in community detection. As a seed, the core forms the backbone to agglomerate other nodes in the community, while the bridge constructs the boundary between communities. Meanwhile, overlapping nodes are largely derived from bridges. Therefore, we utilize the topological features of a node's *k_egonet* and the community-aware bridge centrality [42] to select candidate seeds.

We denote by $k\_egonet(i)$ a subnetwork induced by the k-hop neighbors of node i including i itself, where i is the center of this subnetwork. $1\_egonet(i)$ is a subnetwork composed of center node i and all direct neighbors. Generally, when k is fixed, we name it the *egonet* for short. Meanwhile, we call $del\_k\_egonet(i)$ a decentralized $k\_egonet(i)$, which means the subnetwork formed by the $k\_egonet(i)$ after deleting the central node i.

Berahamand et al. [42] defined a new Degree and Clustering coefficient and Location method (*DCL*) with the advantage of identifying important bridges and hubs.

Definition 1 [42] *DCL* is defined by

$$DCL(i) = deg(i)\frac{1}{CC_i + \frac{1}{deg(i)}} + \frac{\sum_{j \in 1\_egonet(i)} deg(j)}{|E(1\_egonet(i))| + 1}, \quad (1)$$

where $deg(i)$ is the degree of i, the second item is the sum of local clustering coefficient [43] and the inverse of degree of i, and the third item represents the average degree of edges within $1\_egonet(i)$. The local clustering coefficient $CC_i = \frac{2|E(del\_1\_egonet(i))|}{|1\_egonet(i)|(|1\_egonet(i)|-1)}$.

Combining the above strategies, we define the Core-Bridge Triplet ($CBT(i)$) to measure the ability of node i as a community core and bridge.

Definition 2 $CBT(i)$ of node i is defined by

$$CBT(i) = (|del\_k\_egonet(i)|, DCL(i), \omega(del\_k\_egonet(i))). \quad (2)$$

where $\omega(del\_k\_egonet(i))$ is the number of components in the subnetwork of $del\_k\_egonet(i)$.

Here, $del\_k\_egonet(i)$ reflects the cohesion ability of the center node in neighborhood. When node i is deleted from $k\_egonet(i)$, if there are more remaining nodes in $del\_k\_egonet(i)$, then more nodes are connected to the core; if there are more components in $del\_k\_egonet(i)$, then there are denser connections around the core. Therefore, the ability of a node to act as a core can be measured by the characteristics of $del\_k\_egonet(i)$. While a node act as the bridge, DCL provides an efficient measure based on local information. By the aid of features of $del\_k\_egonet(i)$ and DCL, we will measure the core and the bridges reasonably.

Traditional *egonet* only considers one-hop neighbors of a node [44]. The $k\_egonet(i)$ contains k-hop neighbors of node i, which takes into account the farther range. For each node in the network, according to Definition 2, we compute

Table 1 Taking Karate network as an example, the value of Core-Bridge triplet of all nodes (where k in k_egonet is 1)

No.s	CC_i	$N(i)$	$DCL(i)$	$\omega(i)$	*No.s*	CC_i	$N(i)$	$DCL(i)$	$\omega(i)$
0	0.15	16	977.94	4	17	1.0	2	2.47	1
1	0.33	9	155.36	2	18	1.0	2	2.49	1
2	0.24	10	252.12	4	19	0.33	3	12.6	2
3	0.67	6	38.22	1	20	1.0	2	2.49	1
4	0.67	3	7.96	1	21	1.0	2	2.47	1
5	0.5	4	18.39	1	22	1.0	2	2.49	1
6	0.5	4	18.39	1	23	0.4	5	37.04	2
7	1.0	4	11.66	1	24	0.33	3	10.97	2
8	0.5	5	32.92	1	25	0.33	3	11.12	2
9	0	2	7.45	2	26	1.0	2	2.43	1
10	0.67	3	7.96	1	27	0.17	4	34.46	3
11	0	1	0.94	1	28	0.33	3	12.38	2
12	1.0	2	2.44	1	29	0.67	4	15.71	1
13	0.6	5	28.77	2	30	0.5	4	19.52	2
14	1.0	2	2.49	1	31	0.2	6	88.36	3
15	1.0	2	2.49	1	32	0.19	12	429.28	1
16	1.0	2	2.13	1	33	0.11	17	1354.59	4

To simplify, in the table, *No.s* represents the node's number, CC_i is local clustering coefficient, $N(i)$ is the number of nodes in $del\_k\_degnet(i)$, and $\omega(i)$ is the number of components in $del\_k\_egonet(i)$ [1]

Algorithm 2 The process of seed initialization

Input: The network $G = (V, E)$, the parameter k.
Output: The seed set S.

$S \leftarrow \emptyset$;
for $i \in V$ **do**
  compute $CBT(i)$ according to Eq. (2);
end for
$CBT' \leftarrow$ sorting $CBT(i)$;
while $CBT' \neq \emptyset$ **do**
  get the first node v from CBT' and remove v from CBT';
  if v has at least two neighbors **then**
    take the first two neighbors v_1 and v_2 with the largest DCL value;
    if v_1 and v_2 has an edge **then**
      $S \leftarrow [v, v_1, v_2]$;
      remove v_1 and v_2 from CBT';
    end if
  end if
end while
return S;

$CBT(i)$. For example, we calculate $CBT(i)$ of each node in Karate network (Fig. 1a) and list them in Table 1.

Now, we rank $CBT(i)$ by lexical sorting method, which is proposed by Şimşek et al. [45]. Because there are multiple sorting metrics, the idea of this strategy is to sort elements according to the evaluation with the highest priority. When there are several elements with the same ranking under this metric, they are sorted according to the evaluation with the second priority, and so on. We define the priority in $CBT(i)$ as $|del\_k\_egonet(i)|$, $DCL(i)$, $\omega(del\_k\_egonet(i))$ from high to low. For example, in Karate network, the sorted list of all nodes is [33, 0, 32, 2, 1, 31, 3, 23, 8, 13, 27, 30, 5, 6, 29, 7, 19 , 28, 25, 24, 4, 10, 9, 14, 15, 18, 20, 22, 17, 21, 12, 26, 16, 11].

The calculation of $CBT(i)$ is not complicated because only local information is used. In sorted list, we prefer to use the higher-ranked nodes. When selecting the candidate seeds, we proceed from the top node. In order to maximize the aggregation of powerful nodes, a clique is the ideal structure. In fact, to find all cliques is an NP-hard problem. Therefore, we choose the triangle as candidate seeds. Once the top-ranked node v is selected, we scan the neighbors of v. If v has at least two neighbors, two of them with the highest DCL value are selected. In order to form a triangle, two selected neighbors must be adjacent. When a seed is chosen, we remove v and the nodes that have become seeds from the sorted list. The successively search for candidate seeds starts from the highest-ranked node in the remaining list, and so on. When the sorted list completes one scanning round, we get some triangle seeds. The nodes in the candidate seed set have high Core-Bridge functions, and they do not overlap. In Karate network as shown in Fig. 1, using 1_egonet, the top-1 node 33 in the sorted list is selected. Scanning the neighbors of 33, two neighbors 32 and 31 meet conditions (lines 11–13 in Algorithm 2), and span a triangle $\{33, 32, 31\}$. Then, we remove these three nodes from the sorting list. Next, we find the top-1 node 0 in the sorting list, and get the second seed $\{0, 2, 1\}$. Until the entire sorting list is scanned, the final seed sets are $\{33, 32, 31\}$ and $\{0, 2, 1\}$, as shown in Fig. 1b. The process of initializing seeds is shown in Algorithm 2.

3.3.2 Local expansion

At present, there are generally two seed expansion methods: one is local expansion based on the neighborhood, and the other is random walk based on the PageRank vector. Kloumann and Kleinberg confirmed that the method based on PageRank vector outperformed neighborhood-based extensions [46]. In this work, we expand all seed sets with approximate PageRank vectors [38].

The selected seed sets are all triangles. We hope that the seeds will continue to expand as much as possible under strict standards. Strong community [47] is utilized to expand neighbors of the seed sets. A strong community is a subnetwork where each node has a higher probability of being connected with a node inside the subnetwork than that with any other node of the network. If one neighbor satisfies the conditions of Definition 3, it first joins the seed set. Meanwhile, the conductance is given in Definition 4.

Definition 3 [47, 48] A community C is called a strong community, if any node v in the community C satisfies the following equation,

$$k_{in}^{v} > k_{out}^{v}, \tag{3}$$

where k_{in}^{v} is the number of edges that v connects to nodes inside the community, and k_{out}^{v} is the number of edges that v connects to the outside of the community.

Algorithm 3 The process of local expansion

Input: The network $G = (V, E)$, seed set S.
Output: The Local Cover $LCover$.

```
1:  LCover ← ∅;
2:  for s ∈ S do
3:      for v ∈ del_1_egonet(s) do
4:          if k_in^v > k_out^v then
5:              add v to S to get S′;
6:          end if
7:      end for
8:  end for
9:  for s ∈ S′ do
10:     use PageRank vector algorithm to get top-k node sets with minimum
    conductance, add them to LCover;
11: end for
12: return LCover;
```

Definition 4 [17, 37, 47] The conductance $\psi(C)$ is defined by

$$\psi(C) = \frac{k_{out}^{C}}{\min\{k_{in}^{C}, k_{in}^{V\backslash C}\}}, \tag{4}$$

where C is a community, $V\backslash C$ is the complement of C to V, k_{out}^{C} is the number of edges between C and the outside, and k_{in}^{C} is the number of edges inside C.

The first step is to strictly expand the seed set. Then, we use the approximate PageRank vector to further expand the community. The node with the larger PageRank value is more likely to be in the same community as the initial node. The algorithm starts from a seed, and uses the smallest conductance (see Definition 4) as the standard. When a source node satisfies the criterion, it is expanded to k nodes. A more detailed description of the algorithm can be found in [17, 37]. After the local expansion phase of Karate network is completed, communities are shown in Fig. 1c. The process of local expansion is shown in Algorithm 3.

3.3.3 Community optimization

When the preceeding stages are completed, there exist some nodes that have not yet been classified into a community. At the same time, there are some inappropriately classified nodes in the community. To address this issue, we take a stepwise optimizing strategy, which involves three sub-processes of removing, assigning and merging. First, we remove incorrectly partitioned nodes. In order to search the wrongly partitioned nodes efficiently, the semi-local conductance is chosen as the criterion. The change of conductance after node deletion is examined. If the conductance decreases, the node is removed from the community.

There exist some nodes in the network that have not been classified into any community, which will be allocated by two strategies. The first strategy, *SI*, tries to add unassigned nodes to the community in terms of quality. If the addition of a node causes the conductance to decrease, then it will be classified into the community. But this method does not guarantee that all nodes are allocated. Then we define the membership function $MF(v, C)$,

Algorithm 4 The process of community optimization

Input: The network $G = (V, E)$, local cover $LCover$, parameters α, β, γ.
Output: The Final Cover $FCover$.

$FCover \leftarrow \emptyset$;
for $C \in LCover$ **do**
  for $v \in C$ **do**
    if $\psi(C - v) < \psi(C)$ **then**
      remove v from C to obtain $Cover'$;
    end if
  end for
end for
for $u \in unassigned\ \ node\ \ set$ **do**
  for $C \in Cover'$ **do**
    if $\psi(C + u) < \psi(C)$ **then**
      add v into C to obtain $Cover'$;
    end if
  end for
end for
for $u \in unassigned\ \ node\ \ set$ **do**
  compute membership function of u according to Eq. (6) to get $Cover'$;
end for
for $C_i \in Cover'$ **do**
  for $C_j \in Cover'$ **do**
    if $C_i \neq C_j$ and $Merger(C_i, C_j) > 0.5$ **then**
      merge C_i and C_j to obtain $FCover$;
    end if
  end for
end for
return $FCover$;

and add node v into the community with the maximum $MF(v, C)$. This strategy is denoted as FA.

Definition 5 The membership function $MF(v, C)$ between node v and adjacent community C is defined as

$$MF(v,C) = \frac{k_{in}^{v}}{deg(v)} + \frac{\sum_{u \in 1\_egonet(v) \cap C \neq \emptyset} \frac{k_{in}^{u}}{deg(u)}}{deg(v)}, \tag{5}$$

where node v is not in community C, and there exists some edges between v and C.

Then, we denote the communities to which v belongs by $Set(C)$ with the largest membership function, i.e.,

$$Set(C) = \arg\max_{C \in Cover} MF(v,C). \tag{6}$$

So far, all nodes have been perfectly classified. Since the initial seed size is small, after multiple rounds of expansion and optimization, a lot of communities may contain excessive overlapping parts, resulting in similar communities. Therefore, further merging is required. Community merging should satisfy topology and quality criterion. We define the community merge index $MI(C_i, C_j)$, which is defined as follows.

Definition 6 The merge index MI of two communities C_i and C_j is defined by

$$\begin{aligned} MI(C_i,C_j) = {} & \alpha \frac{|C_i \cap C_j|}{\min\{|C_i|, |C_j|\}} \\ & + \beta \frac{|E(C_i) \cap E(C_j)|}{\min\{|E(C_i)|, |E(C_j)|\}} \\ & + \gamma \frac{\psi(C_i) + \psi(C_j)}{2\psi(C_i \cup C_j)}, \end{aligned} \tag{7}$$

where α, β, γ are adjustment parameters subject to $\alpha + \beta + \gamma = 1$.

The first two terms measure the topological similarity of two communities. It can exclude subsets, and communities where nodes and edges are highly overlapping. We use the Hub-Promoted Index (HPI) [49] to calculate the similarity, which is beneficial to the absorption of small communities. The third term in Eq. (7) expects the average conductance to drop. After community optimization and merging have been proceeded in Karate network, communities are shown in Fig. 1d. The algorithm for this stage is shown in Algorithm 4.

3.3.4 Complexity

Assume that the number of nodes in a network is N, the number of edges is M, and the number of communities is l. The average degree of network is $\overline{deg}$, and $|\overline{C}|$ is the average community size.

In seed initialization stage, the complexity of computing DCL is about $O(N * \overline{deg})$. The computation of nodes and components in del_k_egonet is approximately $O(N)$ when $k = 1$ and 2. Because of $\overline{deg} \ll N$, the main overhead in this stage is sorting, which is equal to $O(NlogN)$.

The complexity of local expansion based on PageRank vectors is $O(\sum_{i=1}^{l} k_{out}^{C_i})$ [17]. We suppose that the average edge number of the community connected to outside is denoted as $\overline{k_{out}^{C}}$. It can be approximated as $O(l * \overline{k_{out}^{C}})$. By the fact of $l \ll N$ and $\overline{k_{out}^{C}} \ll M$, the complexity in the second stage is close to $O(N)$.

In community optimization stage, the complexity of removing nodes is $O(l * |\overline{C}| * \overline{k_{out}^{C}})$. Suppose that the unallocated number of nodes is N_u. The complexity of SI method is $O(N_u * l * \overline{k_{out}^{C}})$, and that of FA is $O(N_u * \overline{deg}^2 + NlogN)$. The complexity of community merging is $O(l^2 * (|\overline{C}| + \overline{k_{out}^{C}} + \overline{k_{in}^{C}})) \approx O(l^2 * |\overline{C}| * \overline{deg})$. Because of $N_u \ll N$, the main overhead at this stage is merging. After dividing communities, the value of l and $|\overline{C}|$ are reduced greatly. Therefore, the overall complexity is acceptable, which is also confirmed by subsequent experiments.

4 Experiment and analysis

The algorithms in this work are compiled in Python. The environment of experiments are Intel(R) Core(TM) i5-1135G7 4-core CPU, 4GB memory, and the operating system is Windows 7.

4.1 Evaluation metrics for community quality

First, we recall some evaluation metrics, such as EQ, $ONMI$, $FScore$ and AC and so on, for comparison to analyze the accuracy of the algorithm.

1. EQ (Extended Modularity) [40, 50]. Shen et al. [40] proposed an extension of the classical modularity for overlapping communities, where the value is in [0, 1]. Larger value indicates better community partition. Here,

$$EQ = \frac{1}{2M} \sum_{C \in Cover} \sum_{i,j \in C} \left[A_{ij} - \frac{deg(i)deg(j)}{2M} \right] \frac{1}{O_i O_j}, \tag{8}$$

where O_i represents the number of communities to which node i belong, A is the adjacency matrix.

2. *ONMI* (Overlapping Normalized Mutual Information) [50, 51]. *ONMI* is an extension of *NMI* for overlapping communities. It uses the normalized conditional entropy to quantify the similarity between the found community and the real community to evaluate the accuracy of algorithms, where the field is [0, 1]. The larger the value, the more accurate the community detected. Here,

$$ONMI(X,Y) = 1 - \frac{1}{2}(H(X|Y)_n + H(Y|X)_n), \tag{9}$$

where $H(X|Y)_n$ is the normalized conditional entropy of X over Y, and X and Y are two covers containing different communities.

3. *FScore* Measure [50]. *FScore* Measure is the weighted average of the harmonic averages of precision and recall, where the value is in [0, 1]. The larger value means better for community detection. Here,

$$FScore = \sum_{k=1}^{l} \frac{|C_k|}{N} \max_{1\le i\le k} \frac{2precision(C_k,C_i)recall(C_k,C_i)}{precision(C_k,C_i)+recall(C_k,C_i)}, \tag{10}$$

where C_k represents the k-th real community, C_i indicates the i-th community that the algorithm identifies. The $precision(C_k, C_i)$ and $recall(C_k, C_i)$ represent precision and recall between C_k and C_i.

4. *AC* (Average Conductance) [52]. *AC* is a weighted average of conductances for all communities,

$$AC = \frac{\sum_{i=1}^{l} |E(C_i)|\psi(C_i)}{\sum_{i=1}^{l} |E(C_i)|}. \tag{11}$$

A smaller value implies a better result of community detection.

4.2 Experiments on *LFR* synthetic dataset

In this subsection, we present effectiveness and stability with experimental results in *LFR* synthetic networks.

The synthetic networks are generated using the *LFR* benchmark proposed by Lancichinetti and Fortunato [53, 54]. The parameters in the *LFR* network are as follows: *max_deg* represents the maximum degree; *min_c* and *max_c* represent the minimum community size and maximum community size; μ represents the probability that a node in a community has edge connections with external nodes of the community. The larger the value, the more difficult it is to find communities in the network. O_n represents the number of overlapping nodes; O_m represents the maximum number of communities a node belongs to.

Table 2 The parameter setting of *LFR* synthetic networks

Networks	μ	*max_c*	O_m	O_n
LFR_1	0.1	50	2	{0, 500, 1000, 1500, 2000}
LFR_2	0.3	50	2	{0, 500, 1000, 1500, 2000}
LFR_3	0.1	500	4	{0, 500, 1000, 1500, 2000}
LFR_4	0.3	500	4	{0, 500, 1000, 1500, 2000}

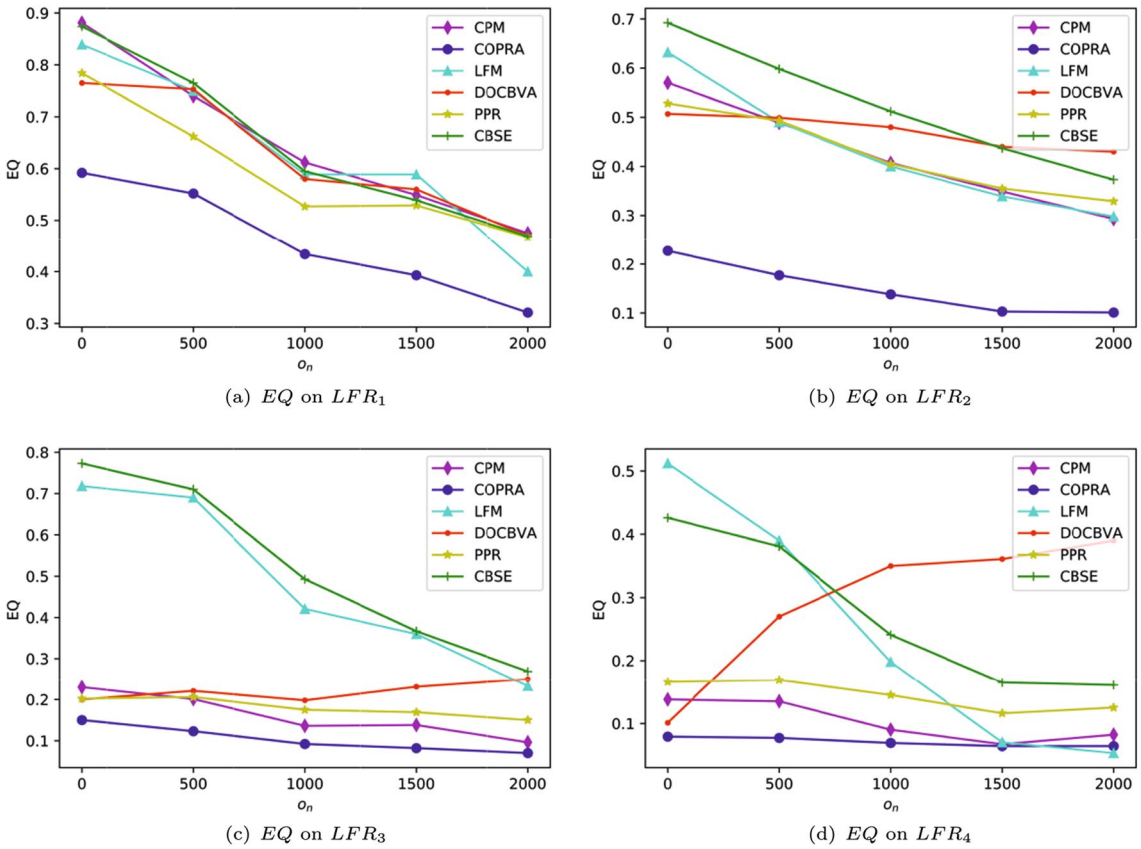
Four groups of *LFR* synthetic datasets are generated, each group contains five networks. They have some unified parameters: $N = 5000$, $\overline{deg} = 15$, $max\_deg = 50$, $min\_c = 10$ (different parameter settings are shown in Table 2).

For *LFR* networks, the compared algorithms are *CPM* [11], *COPRA* [12], *LFM* [14], *DOCBVA* [39], *PPR* [37], respectively. We use *EQ*, *ONMI*, *FScore*, and *AC* as metrics for comparison on synthetic datasets.

The parameters of experiments are as follows: In *CPM*, the size of clique is set to 3. In *LFM*, the parameter in the fitness function is 0.9. In *COPRA*, the length of information stored for a node is set to be 3. In *PPR*, the parameter $\alpha = 0.95$ in local expansion, and the accuracy $\epsilon = 10^4$, and "spread hubs" is selected as the seed selection method. The number of communities needs to be fixed in *PPR*. In experiments, it is set as the average of community number, which is obtained by other methods in each round. The proposed method *CBSE* adopts the same settings as *PPR* in local expansion stage. For the merge index, $\alpha = 0.6$, $\beta = 0.2$ and $\gamma = 0.2$. Based on possible randomness, the algorithms in each network are run 10 times and averaged. Experiments are performed on the synthetic network generated according to settings in Table 2, and the results are shown in Figs. 2, 3, 4 and 5. The x-axis represents the number of overlapping nodes, from 0 to 2000. The y-axis is the values of *EQ*, *ONMI*, *FScore*, and *AC*, respectively.

From the comparison of metrics in Figs. 2, 3, 4 and 5, *CBSE* is more effective in that the found communities has higher accuracy. Especially for *AC*, it is an evaluation based on conductance. Since conductance is optimized several times in the whole process, in most networks, the values of *AC* in *CBSE* is small, which means a better partition. As the number of overlapping nodes increase, the community structure becomes more ambiguous and the difficulty of community detection increases. *CBSE* is also better than most methods, and is stable to obtain better results. The value of *max_c* in LFR_3 and LFR_4 is much larger than that of LFR_1 and LFR_2. The results show that in the presence of a large community, *CBSE* benefits from multiple rounds of community optimization, and the result is more prominent. Therefore, *CBSE* shows better effectiveness.

Then, we illustrate the stability of the proposed algorithm, i.e., the accuracy of the results remains at a good level regardless of whether the community structure is clear or not. Taking LFR_2 and LFR_4 as examples, we calculate the mean and variance of the metric values for each



(a) EQ on LFR_1 (b) EQ on LFR_2

(c) EQ on LFR_3 (d) EQ on LFR_4

Fig. 2 Experimental results of *EQ* on *LFR* networks

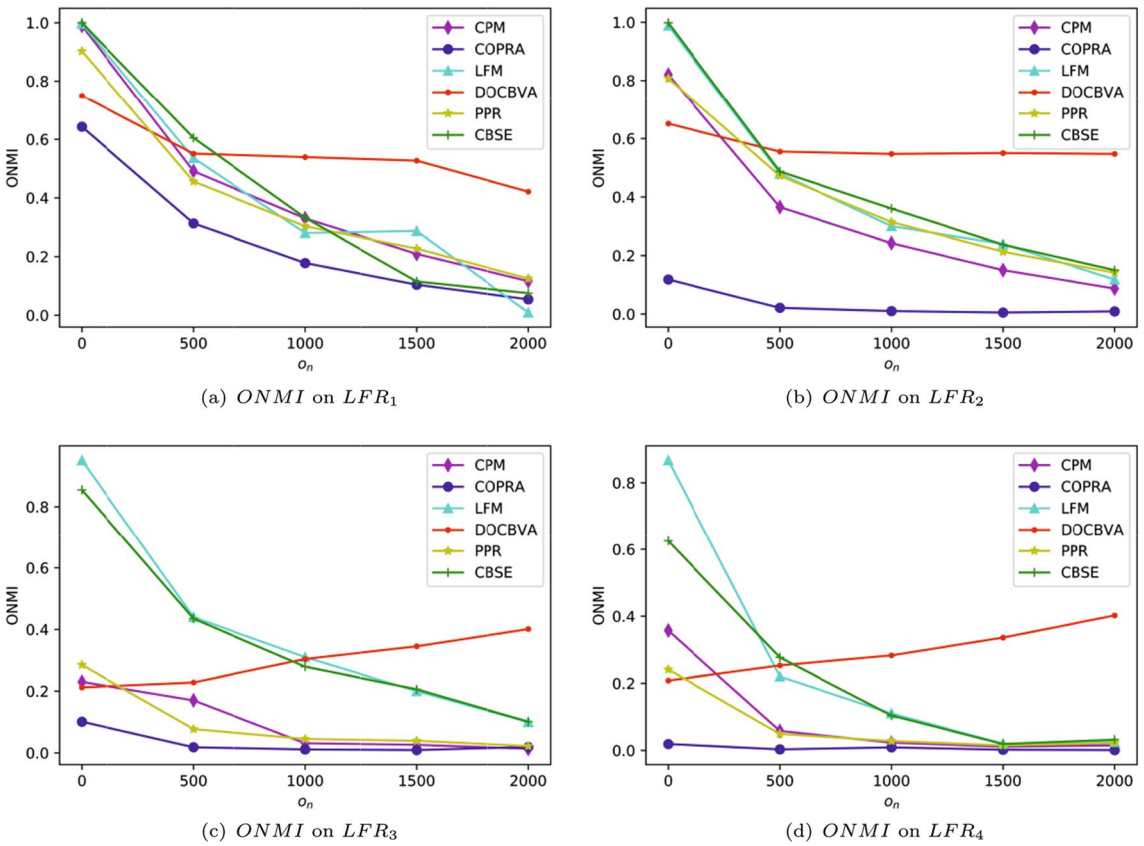
algorithm in a group of *LFR* networks, shown in Tables 3 and 4, respectively.

As can be seen from Tables 3 and 4, the mean of the metrics for *CBSE* is almost optimal or close to the best value, both in LFR_2 and LFR_4. The mean value already includes a variety of scenarios where the community is explicit and unambiguous (with varied O_n). In addition, the variance shows how much the result of each algorithm deviates from the mean. Combining Figs. 2, 3, 4 and 5, it appears that both too large or too small variances are inappropriate. The variances for *CBSE* are within a reasonable range. Thus, it shows that the community results obtained by *CBSE* are stable in all cases.

Furthermore, compared to other methods, beside the excellent performance on the mean, *CBSE* also performs stably during the algorithm execution process. For example, in the implementation of the experiments, *LFM* often falls into an infinite-loop because a node is added and deleted from one community repeatedly. Although it has high accuracy, its effect strongly depends on the network structure. And with the increase of network size, *CPM* will cause the memory to be exhausted. It is noteworthy that *DOCBVA* has low accuracy when the community structure is clear, but improves when the community structure is vague. The reason is that *DOCBVA* has dilemma in seed selection. When the community structure is crisp, the chosen seeds are inaccurate, which leads to the final poor accuracy. When the community structure is fuzzy, there are a large number of single nodes in communities generated by the *DOCBVA*, which coincides with the existence of a large number of small-scale communities in this case.

4.3 Experimental analysis in real networks

We compare seven algorithms to work on six real networks by the evaluation of *EQ*, which are shown in Table 5. Here, in addition to the previous six algorithms, another algorithm adopted for comparison is *CPPR* (Constrained Personalized PageRank) [30]. The settings of algorithms are the same as those of *LFR* networks. For *CPPR*, the value of θ in seeding is labelled in parentheses after *EQ* (Table 6), which varies according to the networks when $\lambda_1 = \lambda_2 = \lambda_3 = 0.1$. The values of *EQ* calculated by all algorithms are listed in Table 6. As can be seen from Table 6, compared with other algorithms, *CBSE* achieves better results on half of



(a) $ONMI$ on LFR_1 (b) $ONMI$ on LFR_2

(c) $ONMI$ on LFR_3 (d) $ONMI$ on LFR_4

Fig. 3 Experimental results of *ONMI* on *LFR* networks

real networks, which also shows the effectiveness of our algorithm.

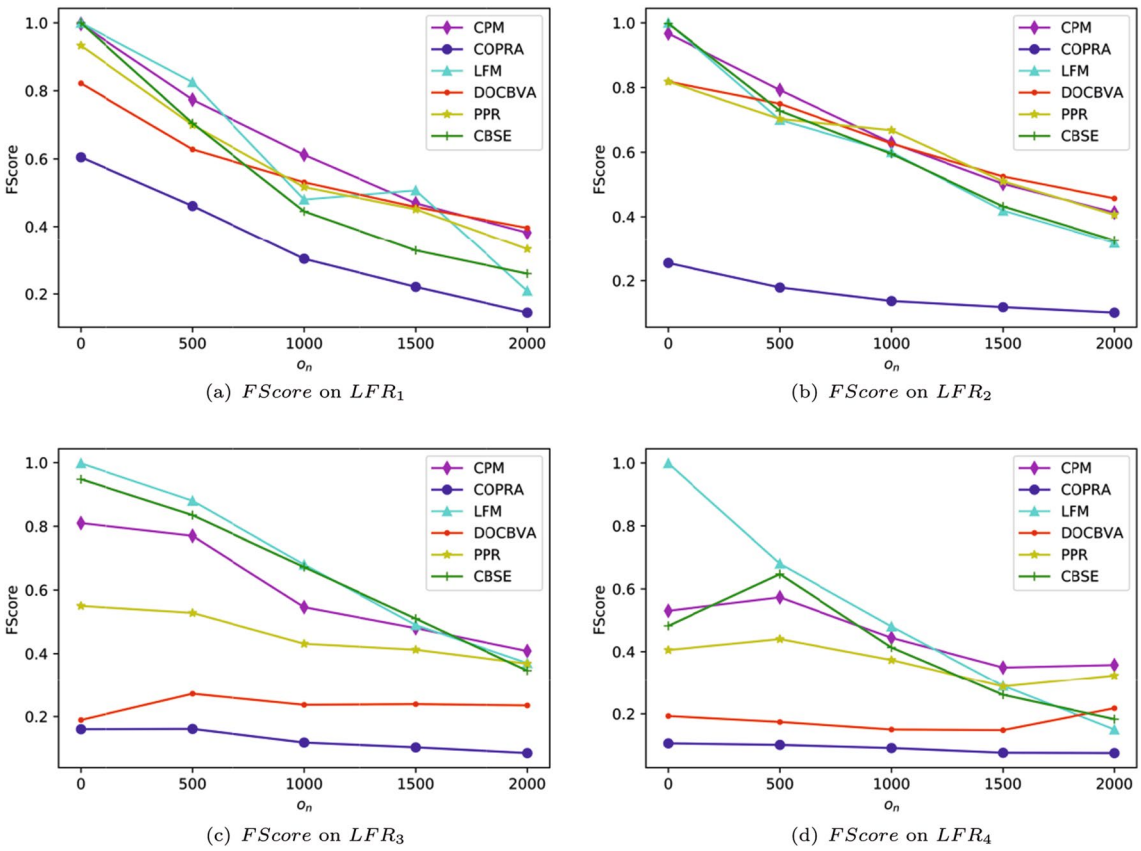
We take the node's number of all real networks as x-axis and running times of different algorithms as y-axis (see Fig. 6). *CPM* is a clique-based algorithm, which takes up a lot of memory to run. The larger the network size, the more likely problems will arise. *COPRA* is a variant of label propagation algorithm, so it is relatively fast but lacks accuracy. In all local expansion algorithms, *LFM* is the fastest, but it is easy to fall into an infinite loop. *DOCBVA* also run fast, but the accuracy is not guaranteed to be stable (Detailed reasons are explained in Sect. 5). *CBSE* is slightly more time-consuming than *PPR* and *CPPR*, mainly due to the cost of community optimization and merging. But it achieves a balance between accuracy and running time on the premise of improving accuracy.

4.4 The setting of *k* in *k_egonet*

In *k_egonet*, the increase of k expands the range of neighborhood, but this expansion will increase the complexity. We take $k = 1$ and $k = 2$ to check the effect of the range of neighborhood on results, respectively. Taking the LFR_2 network as an example (see Table 7), we list *EQ* values after running *CBSE* when k takes different values. It is easy to check that *EQ* values by 2*_egonet* are better than that of 1*_egonet*. This shows that applying two-hop neighbors is better than that using one-hop neighbors. In fact, the larger the neighborhood size is, the more information is comprehensively used, and the higher the accuracy is. For Karate network, the process of community detection (see Fig. 1) is based on $k = 1$. In Fig. 7, we present the process of community detection when $k = 2$. It is easy to see that the effect of $k = 2$ ($EQ = 0.3624$) is better than that of $k = 1$ ($EQ = 0.3521$).

5 Discussion

This section will explore the relationship between community quality and community quantity. Taking LFR_2 as an example, we check the relationship between the benchmark number of communities and the number of communities obtained by compared algorithms. The values of *ONMI* and the number of communities are listed in Table 8. The number of communities in bold indicates the number of communities obtained by the algorithm that is closest to the benchmark number. The *ONMI* in bold represents the largest *ONMI* value obtained by the algorithm.



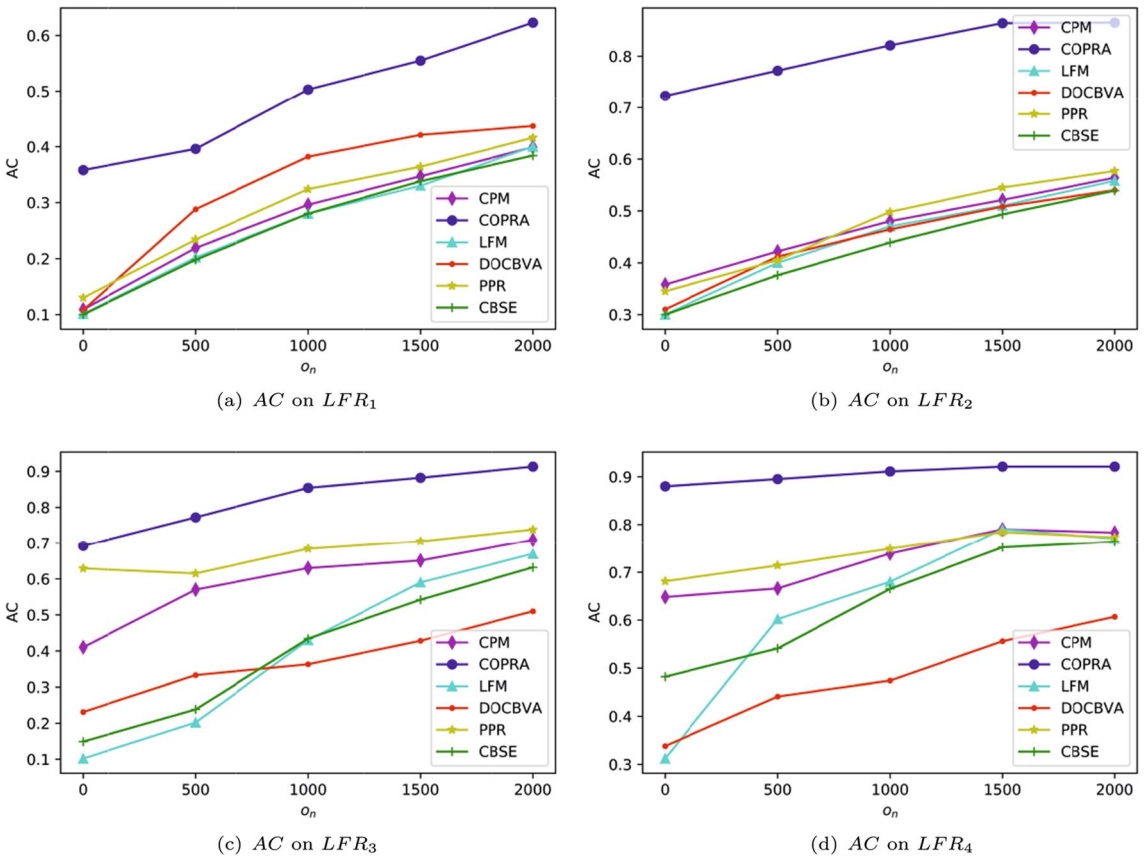
(a) $FScore$ on LFR_1 (b) $FScore$ on LFR_2

(c) $FScore$ on LFR_3 (d) $FScore$ on LFR_4

Fig. 4 Experimental results of *FScore* on *LFR* networks

It is easy to see from Table 8 that the number of overlapping nodes increases, the number of benchmark communities increases sharply. For example, for a network with 5000 nodes, when the number of overlapping nodes reaches 2000, the benchmark number of communities is 2207. This means that the community structure is very unclear, and the average number of nodes in each community is less than 3. Many algorithms directly exclude communities with less than 3 nodes. However, if the number of communities determined by an algorithm is larger, that is, closer to the benchmark number of communities, then *ONMI* will be larger. Just like *DOCBVA*, no matter what the situation is, a large number of single-node communities will be generated. When the community structure is clear, *ONMI* is much worse than other methods. On the contrary, when the number of overlapping nodes is large, it is consistent with the existence of a large number of ultra-small communities. However, this phenomenon benefits from the large number of initial communities obtained by *DOCBVA*, which does not imply the goodness of *DOCBVA*. In general, if the number of communities generated by the algorithm is closer to baseline number, the probability of detecting high-quality communities will be greater. Unfortunately, the final number of communities obtained from most algorithms is small when the community is fuzzy, which may be one of the reasons for the poor result of the community detection algorithm.

We focus on how to improve the algorithm. Due to the large discrepancy between the number of communities and the benchmark, we suspect that, during the community merging phase, is it over-merging? In Equation (7), using Hub Depressed Index (*HDI*) [49] or Jaccard Index [49] to replace Hub-Promoted Index (*HPI*) to calculate the topological similarity may reduce the merge. In addition, increasing the merge threshold in Algorithm 4 may also reduce merging. However, it's hard to decide the time to reduce merging. Alternatively, we can use some prior knowledge to reduce community merging and improve the quality of communities. In terms of running times, the number of communities obtained by the fastest *LFM* and *COPRA* is far from the benchmark. There is a smaller gap between the number of communities obtained by *DOCBVA* (see Table 8) and the benchmark than that of other algorithms. Therefore, under the premise of running faster, *DOCBVA* can be applied to improve the merge. The number of communities from *DOCBVA* is set as threshold η, which is used to improve the problem of excessive community merging. The procedure is presented in Algorithm 5, when the number of communities reaches



(a) AC on LFR_1 (b) AC on LFR_2

(c) AC on LFR_3 (d) AC on LFR_4

Fig. 5 Experimental results of *AC* on *LFR* networks

Table 3 The mean and variance of the metric value in LFR_2 networks

Metric	*EQ*		*ONMI*		*FScore*		*AC*	
	Mean	*Variance*	*Mean*	*Variance*	*Mean*	*Variance*	*Mean*	*Variance*
CPM	0.4216	0.009717	0.3328	0.068096	**0.6606**	0.039721	0.4690	0.005276
CORPA	0.1492	0.002282	0.0326	0.001851	0.1580	0.003028	0.8090	0.003014
LFM	0.4318	0.014199	0.4250	0.092893	0.6074	0.056402	0.4476	0.008127
DOCBVA	0.4712	0.000961	**0.5700**	0.001649	0.6354	0.018112	0.4468	0.006531
PPR	0.4220	0.005941	0.3894	0.055943	0.6210	0.021258	0.4740	0.007526
CBSE	**0.5224**	0.012840	0.4464	0.089015	0.6158	0.055373	**0.4294**	0.007149

The mean is denoted as *Mean* and the variance is denoted as *Variance*. The best results of the means for each metric are shown in bold

Table 4 The mean and variance of the metric value in LFR_4 networks

Metric	*EQ*		*ONMI*		*FScore*		*AC*	
	Mean	*Variance*	*Mean*	*Variance*	*Mean*	*Variance*	*Mean*	*Variance*
CPM	0.1024	0.000831	0.0928	0.017726	0.4506	0.008083	0.7252	0.003439
CORPA	0.0706	4.02E-05	0.0068	4.5E-05	0.0898	0.00016	0.9056	0.000254
LFM	0.2446	0.032402	0.2484	0.10094	**0.5198**	0.089288	0.6308	0.029921
DOCBVA	0.2744	0.010933	**0.2964**	0.004519	0.1766	0.000703	**0.4832**	0.008715
PPR	0.1442	0.000451	0.0710	0.007354	0.3658	0.002996	0.7402	0.001453
CBSE	**0.2948**	0.012058	0.2114	0.051221	0.3972	0.026855	0.6408	0.012661

The mean is denoted as *Mean* and the variance is denoted as *Variance*. The best results of the means for each metric are shown in bold

Table 5 The basic information of real networks

No.	*Name*	$\|N\|$	$\|E(N)\|$	*Reference*
N_1	Karate	34	78	[55]
N_2	Dolphin	62	159	[56]
N_3	Polbooks	105	441	[57]
N_4	Football	115	613	[2]
N_5	Email	1133	5451	[58]
N_6	PGP	10,680	24,316	[59]

the threshold, the merging stops. Note that Algorithm 5 is only an improvement for Algorithm 4.

Although Algorithm 5 improves the merging step, it brings a larger computational cost, which is not suitable for large-scale networks. In the future, we will pursue more efficient methods of improvement.

6 Conclusion

In this work, we propose a novel seed selection method based on core and bridge nodes, and design an overlapping community detection algorithm *CBSE* by virtue of local expansion and community optimization. First, we select seeds from nodes that act as bridges and cores, to choose initial community centers and construct community boundaries through seeds. Then, the strong community condition and the approximate PageRank vector is used for local expansion, which is proved to be efficient. Finally, the community optimization and merging of the initial cover are gradually refined. We remove incorrectly partitioned nodes, and then assign unclassified nodes by *SI* and *FA* strategies. At the

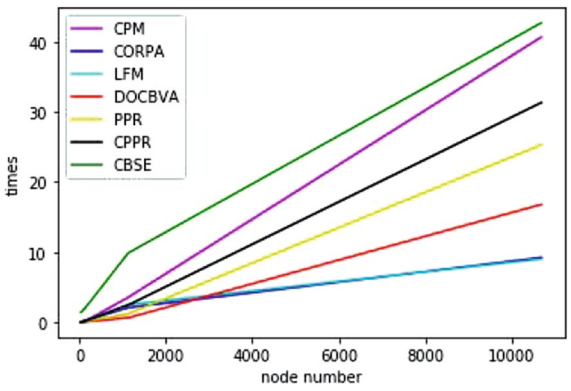


Fig. 6 The relationship between the number of nodes and the running time in real networks

Table 7 The comparison of *EQ* for LFR_2

EQ	$O_n = 0$	$O_n = 500$	$O_n = 1000$	$O_n = 1500$	$O_n = 2000$
1_*egonet*	0.6837	0.5974	0.4883	0.4171	0.3547
2_*egonet*	0.6915	0.5983	0.5115	0.4371	0.3731

Table 6 The comparison of *EQ* results on real networks

No.	*EQ*						
	CBSE	*CPM*	*COPRA*	*LFM*	*DOCBVA*	*PPR*	*CPPR*
N_1	**0.3624**	0.1858	0.3448	0.3299	0.1531	0.3594	0.3577 (0.20)
N_2	0.4539	0.3229	0.3794	**0.5058**	0.2875	0.3165	0.4291 (0.20)
N_3	**0.4933**	0.4372	0.4865	0.4776	0.4052	0.2777	0.4738 (0.15)
N_4	**0.5928**	0.3950	0.5242	0.4335	0.3880	0.4124	0.4866 (0.09)
N_5	0.3382	0.1514	0.3399	0.3090	0.2679	0.2204	**0.3423** (0.22)
N_6	**0.6358**	0.3686	0.6269	0.6220	0.4840	0.5117	0.5782 (0.20)

Bold text indicates the highest *EQ* value in the same network. The number shown in parentheses of the last column is the value of the algorithm's parameter θ

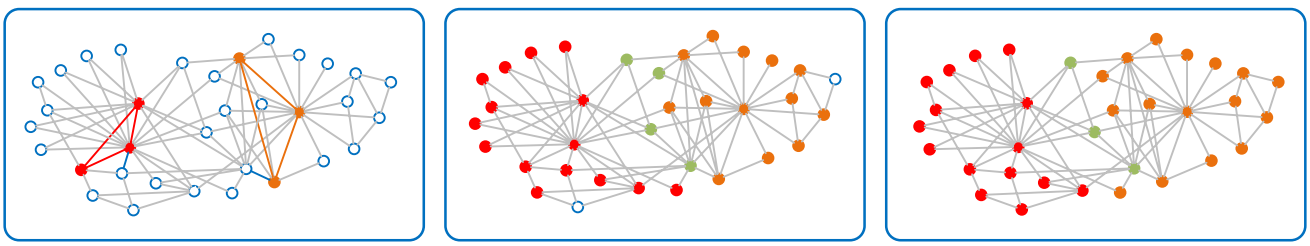


Fig. 7 When $k = 2$, the process of community detection in Karate network. Blue hollow dots represent unassigned nodes, red and orange solid dots are different community nodes. The green solid dots are overlapping nodes (color figure online)

Table 8 A list of the benchmark number of communities, the number of communities and *ONMI* obtained by each algorithm in LFR_2 networks

P_{on}	b_n	*CPM*		*COPRA*		*LFM*	
		c_n	*ONMI*	c_n	*ONMI*	c_n	*ONMI*
0	218	317	0.8205	131	0.1182	305	0.9885
0.1	706	356	0.3656	122	0.0213	201	0.4801
0.2	1213	400	0.2421	133	0.0095	238	0.3005
0.3	1714	481	0.1507	139	0.0054	212	0.2394
0.4	2207	563	0.0879	141	0.0090	221	0.1187
P_{on}	b_n	*DOCBVA*		*PPR*		*CBSE*	
		c_n	*ONMI*	c_n	*ONMI*	c_n	*ONMI*
0	218	212	0.6518	182	0.8070	**218**	**0.9975**
0.1	706	**430**	**0.5558**	182	0.4721	181	0.4868
0.2	1213	**661**	**0.5475**	348	0.3149	198	0.3600
0.3	1714	**876**	**0.5508**	348	0.2131	195	0.2377
0.4	2207	**1077**	**0.5479**	348	0.1418	214	0.1499

The first column P_{on} is the proportion of overlapping nodes to all nodes. b_n represents the number of benchmark communities, and c_n is the number of communities obtained by the algorithm

Algorithm 5 The community merge improvement

Input: The network $G = (V, E)$, cover $Cover$, community number threshold η, merge threshold ρ.
Output: The cover after processing $PCover$.

$PCover \leftarrow Cover$;
while $len(PCover) > \eta$ **do**
  for $C \in PCover$ **do**
    compute k_{in}^{C}/k_{out}^{C}
  end for
  sort the above ratios in ascending order to get the community number sequence D;
  take the first community C_1 in D;
  remove C_1 from D;
  for $C_2 \in D$ **do**
    calculate the merge index $Merge(C_1, C_2)$ according to Eq.(7);
    if $Merge(C_1, C_2) > \rho$ **then**
      merge C_1 and C_2, join to $PCover$, delete C_1, C_2 from $PCover$;
      delete C_2 from D;
      break;
    end if
  end for
end while
return $PCover$;

same time, both topology and quality criteria are taken into account in the merging stage. The high-quality communities are detected by the proposed algorithm effectively and stably. Experimental results on synthetic and real networks show that our algorithm outperforms some overlapping community detection algorithms. We also explore the relationship between the number of communities and the quality of communities, and propose an improved method in terms of prior knowledge of the known algorithm. Our work improves the identification accuracy in overlapping communities, with a balance of effectiveness and efficiency. In the future, we will work on detecting communities effectively through the semi-supervised method in attributed networks.

Acknowledgements The authors would like to express their sincere gratitude to all reviewers for valuable suggestions, which are helpful in improving and clarifying the original manuscript. We thank the National Institute of Education, Nanyang Technological University, where part of this research was performed. This work was partly supported by the National Natural Science Foundation of China (Nos. 61977016 and 61572010), Natural Science Foundation of Fujian Province (Nos. 2020J01164, 2017J01738). This work was also partly supported by Fujian Alliance of Mathematics (No. 2023SXLMMS04) and China Scholarship Council (CSC No. 202108350054).

Data availability No datasets were generated or analysed during the current study.

References

1. Chen J, Yuan B (2006) Detecting functional modules in the yeast protein–protein interaction network. Bioinformatics 22(18):2283–2290
2. Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99(12):7821–7826
3. Dourisboure Y, Geraci F, Pellegrini M (2007) Extraction and classification of dense communities in the web. In: Proceedings of the 16th international conference on World Wide Web, pp 461–470
4. Li X, Wang Z, Hu R, Zhu Q, Wang L (2019) Recommendation algorithm based on improved spectral clustering and transfer learning. Pattern Anal Appl 22(2):633–647
5. Coscia M, Giannotti F, Pedreschi D (2011) A classification for community discovery methods in complex networks. Stat Anal Data Min ASA Data Sci J 4(5):512–546
6. Lyzinski V, Tang M, Athreya A, Park Y, Priebe CE (2016) Community detection and classification in hierarchical stochastic blockmodels. IEEE Trans Netw Sci Eng 4(1):13–26
7. Hou Chin J, Ratnavelu K (2017) A semi-synchronous label propagation algorithm with constraints for community detection in complex networks. Sci Rep 7(1):45836
8. Fortunato S (2010) Community detection in graphs. Phys Rep 486(3–5):75–174
9. Chen Y, Mo D (2022) Community detection for multilayer weighted networks. Inf Sci 595:119–141
10. Tang Z, Tang Y, Li C, Cao J, Chen G, Lin R (2021) A fast local community detection algorithm in complex networks. World Wide Web 24(6):1929–1955
11. Palla G, Derényi I, Farkas I, Vicsek T (2005) Uncovering the overlapping community structure of complex networks in nature and society. Nature 435(7043):814–818
12. Gregory S (2010) Finding overlapping communities in networks by label propagation. New J Phys 12(10):103018
13. Xie J, Szymanski BK, Liu X (2011) Slpa: uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process. In: IEEE 11th international conference on data mining workshops, pp 344–349
14. Lancichinetti A, Fortunato S, Kertész J (2009) Detecting the overlapping and hierarchical community structure in complex networks. New J Phys 11(3):033015
15. Wu Z, Lin Y, Gregory S, Wan H, Tian S (2012) Balanced multilabel propagation for overlapping community detection in social networks. J Comput Sci Technol 27(3):468–479
16. Ahn YY, Bagrow JP, Lehmann S (2010) Link communities reveal multiscale complexity in networks. Nature 466(7307):761–764
17. Whang JJ, Gleich DF, Dhillon IS (2016) Overlapping community detection using neighborhood-inflated seed expansion. IEEE Trans Knowl Data Eng 28(5):1272–1284
18. Jokar E, Mosleh M, Kheyrandish M (2022) Overlapping community detection in complex networks using fuzzy theory, balanced link density, and label propagation. Expert Syst 39(5):e12921
19. Shen H, Cheng X, Guo J (2009) Quantifying and identifying the overlapping community structure in networks. J Stat Mech Theory Exp 07:P07042
20. Evans TS (2010) Clique graphs and overlapping communities. J Stat Mech Theory Exp 12:P12037
21. Luo M, Xu Y (2022) Community detection via network node vector label propagation. Phys A 593:126931
22. Yang J, Leskovec J (2013), Overlapping community detection at scale: a nonnegative matrix factorization approach. In: Proceedings of the 6th ACM international conference on Web search and data mining, pp 587–596
23. Coscia M, Rossetti G, Giannotti F, Pedreschi D (2012) Demon: a local-first discovery method for overlapping communities. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining, pp 615–623
24. Lancichinetti A, Radicchi F, Ramasco JJ, Fortunato S (2011) Finding statistically significant communities in networks. PLoS ONE 6(4):e18961
25. Baumes J, Goldberg MK, Krishnamoorthy MS, Magdon Ismail M, Preston N (2005) Finding communities by clustering a graph into overlapping subgraphs. In: Proceedings of the IADIS international conference on applied computing, pp 97–104
26. Yang J, Zhang X (2017) Finding overlapping communities using seed set. Phys A 467:96–106
27. Yin H, Benson AR, Leskovec J, Gleich DF (2017) Local higher-order graph clustering. In: Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining, pp 555–564
28. Wang X, Liu G, Li J, Nees JP (2017) Locating structural centers: a density-based clustering method for community detection. PLoS ONE 12(1):e0169355
29. Gao Y, Zhang H, Zhang Y (2019) Overlapping community detection based on conductance optimization in large-scale networks. Phys A 522:69–79
30. Gao Y, Yu X, Zhang H (2021) Overlapping community detection by constrained personalized PageRank. Expert Syst Appl 173:114682
31. Basuchowdhuri P, Sikdar S, Nagarajan V, Mishra K, Gupta S, Majumder S (2019) Fast detection of community structures using graph traversal in social networks. Knowl Inf Syst 59(1):1–31
32. Ding X, Zhang J, Yang J (2020) Node-community membership diversifies community structures: an overlapping community detection algorithm based on local expansion and boundary rechecking. Knowl Based Syst 198:105935
33. Ding X, Yang H, Zhang J, Yang J, Xiang X (2022) Ceo: identifying overlapping communities via construction, expansion and optimization. Inf Sci 596:93–118
34. Cheng F, Wang C, Zhang X, Yang Y (2020) A local-neighborhood information based overlapping community detection algorithm for large-scale complex networks. IEEE ACM Trans Netw 29(2):543–556
35. Jiang H, Liu Z, Liu C, Su Y, Zhang X (2020) Community detection in complex networks with an ambiguous structure using central node based link prediction. Knowl Based Syst 195:105626
36. Padrol Sureda A, Perarnau Llobet G, Pfeifle J, Muntés Mulero V (2010) Overlapping community search for social networks. In: IEEE 26th international conference on data engineering (ICDE 2010), pp 992–995
37. Whang JJ, Gleich DF, Dhillon IS (2013) Overlapping community detection using seed set expansion. In: Proceedings of the 22nd

ACM international conference on information & knowledge management, pp 2099–2108
38. Andersen R, Chung F, Lang K (2006) Local graph partitioning using pagerank vectors. In: 47th annual IEEE symposium on foundations of computer science (FOCS'06), pp 475–486
39. Wang X, Wang Y, Qin X, Li R, Eustace J (2018) Detecting overlapping communities based on vital nodes in complex networks. Chin Phys B 27(10):100504
40. Shen H, Cheng X, Cai K, Hu MB (2009) Detect overlapping and hierarchical community structure in networks. Phys A 388(8):1706–1712
41. Su Y, Wang B, Zhang X (2017) A seed-expanding method based on random walks for community detection in networks with ambiguous community structures. Sci Rep 7(1):1–10
42. Berahmand K, Bouyer A, Samadi N (2019) A new local and multidimensional ranking measure to detect spreaders in social networks. Computing 101(11):1711–1733
43. Meghanathan N (2017) A computationally lightweight and localized centrality metric in lieu of betweenness centrality for complex network analysis. Vietnam J Comput Sci 4(1):23–38
44. Meghanathan M (2021) Neighborhood-based bridge node centrality tuple for complex network analysis. Appl Netw Sci 6(1):1–36
45. Şimşek A (2021) Lexical sorting centrality to distinguish spreading abilities of nodes in complex networks under the susceptible-infectious-recovered (sir) model. J King Saud Univ Comput Inf Sci 34(8):4810–4820
46. Kloumann IM, Kleinberg JM (2014) Community membership identification from small seed sets. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1366–1375
47. Fortunato S, Hric D (2016) Community detection in networks: a user guide. Phys Rep 659:1–44
48. Radicchi F, Castellano C, Cecconi F, Loreto V, Parisi D (2004) Defining and identifying communities in networks. Proc Natl Acad Sci 101(9):2658–2663
49. Zhou T, Lü L, Zhang YC (2009) Predicting missing links via local information. Eur Phys J B 71(4):623–630
50. Chakraborty T, Dalmia A, Mukherjee A, Ganguly N (2017) Metrics for community analysis: a survey. ACM Comput Surv: CSUR 50(4):1–37
51. McDaid AF, Greene D, Hurley N (2011) Normalized mutual information to evaluate overlapping community finding algorithms. arXiv:1110.2515
52. Leskovec J, Lang KJ, Mahoney M (2010) Empirical comparison of algorithms for network community detection. In: Proceedings of the 19th international conference on World Wide Web, pp 631–640
53. Lancichinetti A, Fortunato S, Radicchi F (2008) Benchmark graphs for testing community detection algorithms. Phys Rev E 78(4):046110
54. Lancichinetti A, Fortunato S (2009) Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. Phys Rev E 80(1):016118
55. Zachary WW (1977) An information flow model for conflict and fission in small groups. J Anthropol Res 33(4):452–473
56. Lusseau D, Schneider K, Boisseau OJ, Haase P, Slooten E, Dawson SM (2003) The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. Behav Ecol Sociobiol 54(4):396–405
57. Kunegis J (2013) KONECT—the Koblenz network collection. In: Proceedings of the international conference on World Wide Web companion, pp 1343–1350
58. Guimera R, Danon L, Diaz Guilera A, Giralt F, Arenas A (2003) Self-similar community structure in a network of human interactions. Phys Rev E 68(6):065103
59. Boguná M, Pastor Satorras R, Díaz Guilera A, Arenas A (2004) Models of social networks based on social distance attachment. Phys Rev E 70(5):056122