SCIENTIFIC REPORTS

OPEN

Received: 24 August 2016
Accepted: 28 December 2016
Published: 03 February 2017

A seed-expanding method based on random walks for community detection in networks with ambiguous community structures

Yansen Su, Bangju Wang & Xingyi Zhang

Community detection has received a great deal of attention, since it could help to reveal the useful information hidden in complex networks. Although most previous modularity-based and local modularity-based community detection algorithms could detect strong communities, they may fail to exactly detect several weak communities. In this work, we define a network with clear or ambiguous community structures based on the types of its communities. A seed-expanding method based on random walks is proposed to detect communities for networks, especially for the networks with ambiguous community structures. We identify local maximum degree nodes, and detect seed communities in a network. Then, the probability of a node belonging to each community is calculated based on the total probability model and random walks, and each community is expanded by repeatedly adding the node which is most likely to belong to it. Finally, we use the community optimization method to ensure that each node is in a community. Experimental results on both computer-generated and real-world networks demonstrate that the quality of the communities detected by the proposed algorithm is superior to the- state-of-the-art algorithms in the networks with ambiguous community structures.

Extensive researches on real-world networks show that community structure is an important property[1,2]. The nodes of the same community may be the individuals with certain relationships in social networks, the genes or proteins with the similar function and the web pages dealing with the same topic[3,4]. Therefore, it is helpful to reveal community structures to understand the structures of networks and detect potentially useful information of networks.

Communities can be loosely defined as the subsets of nodes which are more densely linked than the rest of the network. In this sense, modularity and local modularity were proposed as indices of community structure[1,5]. There are also two community definitions (i.e., the strong and weak community definitions) based on the topology of networks, where the strong community is the community in which each node has more connections than in each rest community and the weak community is the community in which the sum of all degrees is larger than the sum of all degrees in each rest community[6,7]. In this article, following the strong and weak community definitions, we define that a network has a clear or ambiguous community structure as follows. If the communities in a network are all strong communities, then the network has a clear community structure; otherwise, if some communities in a network are weak communities, then the network has an ambiguous community structure.

Many efforts have focused on detecting community structures in complex networks[8–14]. Popular algorithms include modularity-based algorithms (e.g. Girvan-Newman algorithm (*GN*)[8], fast Newman algorithm (*FN*)[9] and Fast unfolding algorithm (*FUA*)[13]) and local modularity-based algorithms (e.g. local maximum degree algorithm (*LMDR*)[14]). Most of the modularity-based and local modularity-based community detection algorithms could exactly detect communities in several networks, especially for the networks with clear community structures[15–17]. However, these algorithms still suffer from some limitations, which may prevent them to achieve satisfactory performance on the network with ambiguous community structures. For instance, the modularity-based measurement may fail to identify the modules which are smaller than the scale which depends on the total number

Key Lab of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230039, China. Correspondence and requests for materials should be addressed to X.Z. (email: xyzhanghust@gmail.com)

of edges and on the degree of interconnectedness of the modules[7]. *LMDR* is a greedy maximum algorithm which starts from a local degree central node whose degree is greater than or equal to the degree of its neighbor nodes, and then iteratively adds the nodes yielding the largest increase of the local modularity until the community reached a predefined size. However, some weak communities may fail to be detected by *LMDR*. The modularity-based and local modularity-based algorithms mainly maximize the modularity and local modularity, which only compare the inner edges of a community with the edges between the community and the rest part of the network. Thus, it is hard to exactly detect some weak communities by the modularity-based and local modularity-based algorithms.

Besides the community detection methods mentioned above, researches explore several random walks-based methods for community detection (e.g., a seed set expansion algorithm[18] and an algorithm for finding and extracting a community (*FEC*)[19]), since the random walks-based techniques have a good ability to deal with uncertainty or fuzziness. Previous researches show that the communities identified by random walks-based algorithms are structurally close to real-world communities[20]. Specifically, the basic idea of *FEC* algorithm is that a random walker is more likely to reach the nodes in its own community, when compared to other communities[19]. Following the basic idea, the algorithm checks whether a node should be added into the community by comparing the probability of this node in the community with the one of the node in each of the rest communities. It is likely to identify the communities, in which each node has more connections than in each rest community. However, *FEC* is unstable, as the algorithm starts with an arbitrary destination node; the performance of *FEC* needs to be enhanced in networks, as it is hard to accurately detect weak communities.

Here, inspired by the basic idea of *FEC* algorithm, we propose a random walks-based algorithm named *RWA* to detect communities for complex networks, especially for the networks with ambiguous community structures. The overall framework of *RWA* is selecting the dense subgraphs which contain important nodes in a network, and expanding these dense subgraphs based on random walks. Specifically, (1) the seed communities are detected based on the nodes whose degree is greater than or equal to the degree of its neighbor nodes; (2) the seed communities are expanded using random walks; (3) the expanded communities are adjusted to ensure each node in a network is in a community. A difference between *FEC* and *RWA* is that, a seed in *FEC* is an arbitrary node which leads to the instability of detection results, while a seed in *RWA* is a dense subgraph which could avoid the instability of detection results. The performance of *RWA* is tested on both computer-generated and real-world networks. Experimental results demonstrate that the quality of the communities detected by *RWA* is superior to those detected by comparative algorithms, especially in the networks which have ambiguous community structures. *RWA* may be helpful to understand the real-world networks, most of which have ambiguous community structures.

Results

This section presents the comparative results of the proposed algorithm and the traditional algorithms in the experiments preformed on both computer-generated and real-world networks.

Computer-generated and real-world networks. The first kind of computer-generated networks employed in the experiments are the GN benchmark networks, proposed by Girvan and Newman[8]. This network is constructed as follows: 128 nodes are randomly and equally divided into four communities; edges are randomly placed between node pairs to make the average degree of the graph equal to 16. Each pair of nodes in the same community has an edge with probability P_{in}. Here, P_{in} is a parameter of networks generated. Generally speaking, when $P_{in} < 0.40$, it is unable to detect community structures. When the value of P_{in} becomes larger, the community can be more easily detected. In our experiments, $0.40 \leq P_{in} \leq 0.90$. For each $P_{in} \in \{0.40, 0.45, \cdots, 0.90\}$, 100 networks are generated. According to the parameter P_{in}, the computer-generated networks could be classified into two classes. When $0.80 \leq P_{in} \leq 0.90$, all of the communities in the networks are strong communities (p-value = 0.05). These networks have clear community structures. When $0.40 \leq P_{in} < 0.80$, some of the predefined communities are not strong communities, but they are weak communities. In this situation, the networks have ambiguous community structures.

Another set of computer-generated networks is the LFR benchmark networks[21]. Compared with the GN benchmark networks, the LFR benchmark networks have more adjustable parameters, which control the number of nodes generated, the average degree of nodes and the size of communities generated. The LFR benchmark networks mainly include the following parameters: N is the number of nodes in networks; d is the average degree of nodes in network; $Maxd$ is the biggest degree of node; $Minc$ is the number of nodes that the smallest community contains; $Maxc$ is the number of nodes that the biggest community contains; and μ is the probability of nodes connected with nodes of external community. The bigger μ is, the more difficult the community detection is. When $u \geq 0.3$, the networks have ambiguous community structures (p-value < 0.05). We produce two groups of the LFR benchmark networks. These two groups share these parameters $d = 10$, $Maxd = 50$, $Minc = 10$ and $Maxc = 20$. The numbers of nodes in these two groups of networks are set to $N = 200$ and $N = 300$, respectively. The value of μ in each group is set from 0.1 to 0.6, with the interval 0.1.

We also employ four real-world networks in the experiments. The four real-world networks are the Zachary's Karate Club network (Karate network, for short)[22], the Bottlenose Dolphins network (Dolphins network, for short)[23], the Books about US politics network (Polbooks network, for short)[24] and the American College Football network (Football network, for short)[8], respectively. Each real-world network employed in our work has at least one weak community (see Table 1). Thus, all of the four real-world networks have ambiguous community structures.

	Karate	Dolphins	Polbooks	Football
strong	0	1	1	8
weak	2	1	2	4

Table 1. The number of strong and weak communities in real-world networks. 'Karate', 'Dolphins', 'Polbooks' and 'Football' represent the Zachary's Karate Club network, the Bottlenose Dolphins network, the Books about US politics network and the American College Football network, respectively.

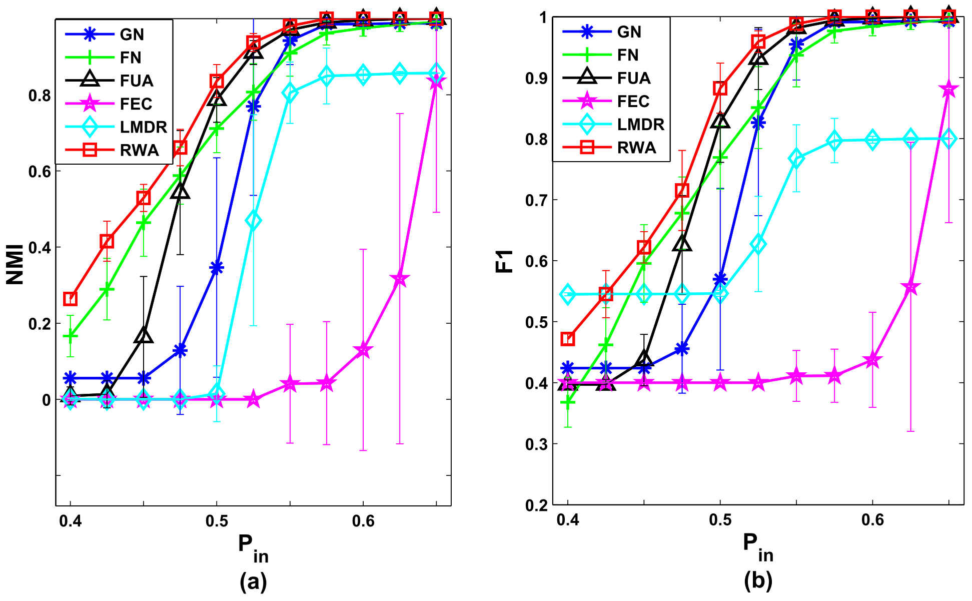

Figure 1. The comparative results on the GN benchmark networks. Each point is the mean of *NMI* and *F*1 values averaged over 30 independent runs. Error bars show the standard deviations estimated from 30 networks.

Comparison and evaluation. *Comparison with other algorithms.* We verify the performance of the proposed algorithm (*RWA*) by comparing it with five representative algorithms (*GN*, *FN*, *FUA*, *FEC* and *LMDR*) on both computer-generated networks and four real-world networks.

(1) We apply our algorithm and other five algorithms (*GN*, *FN*, *FUA*, *FEC* and *LMDR*) to the GN benchmark networks with 128 nodes and four predetermined communities ($Pin \in \{0.4, 0.45, \cdots, 0.9\}$). The comparative results on computer-generated networks are given in Fig. 1, with both the mean of the normalized mutual information (*NMI*) values and the mean of the F-measure (*F*1) values averaged over 30 independent runs for *RWA* and other five representative algorithms.
As can been seen from Fig. 1(a), when the networks have clear community structures (i.e., $P_{in} \geq 0.80$), all algorithms except *FEC* and *LMDR* can get the nearly true partition results (*NMI* value is nearly 1.0). In this situation, *RWA* performs very similar to the comparative algorithms (*GN*, *FN* and *FUA*). However, *RWA* generates the best detection results when the networks have ambiguous community structures ($0.40 \leq P_{in} < 0.80$), and the results obtained through our algorithm remain relatively stable. When the networks have clear community structures (i.e., $P_{in} \geq 0.80$), the *F*1 value obtained through *RWA* is no less than those obtained by other five comparative algorithms. In details, the *F*1 values of *RWA* and four comparative algorithms (*GN*, *FN* and *FUA*) are almost 1.0 when $P_{in} \geq 0.80$. That is, the proposed algorithm and four of the five comparative algorithms could get the nearly true partition results. In contrast, both *LMDR* and *FEC* produce the *F*1 values which are less than 0.90. When the networks have ambiguous community structures (i.e., $0.40 \leq P_{in} < 0.80$), the values of *F*1 obtained by *RWA* are not the largest, and *RWA* performs slightly less well than some of the comparative algorithms (e.g. *LMDR*) for few detection problems (i.e., $P_{in} = 0.40$). However, the detection results shows that *RWA* has the best performance. When two evaluation measures (*NMI* and *F*1) are considered together, although the *F*1 value of *RWA* is smaller than that of *LMDR* for few detection problems, the performance of *RWA* is still better than *LMDR*. Actually, when $0.40 \leq P_{in} \leq 0.55$, it can be seen that the *F*1 values of *LMDR* are lager than some comparative algorithms, and the *NMI* values of *LMDR* are equal to zero in networks. That is because in these situations, all nodes in the network fall into a community, which is far from the true partition. Besides, the *F*1 values obtained through *RWA* decline relatively stable, and our algorithm obtains the best results when $0.4 \leq P_{in} < 0.80$. We can conclude from Fig. 1 that *RWA* performs the best among the comparative algorithms on the GN benchmark networks, especially when the networks have ambiguous community structures.

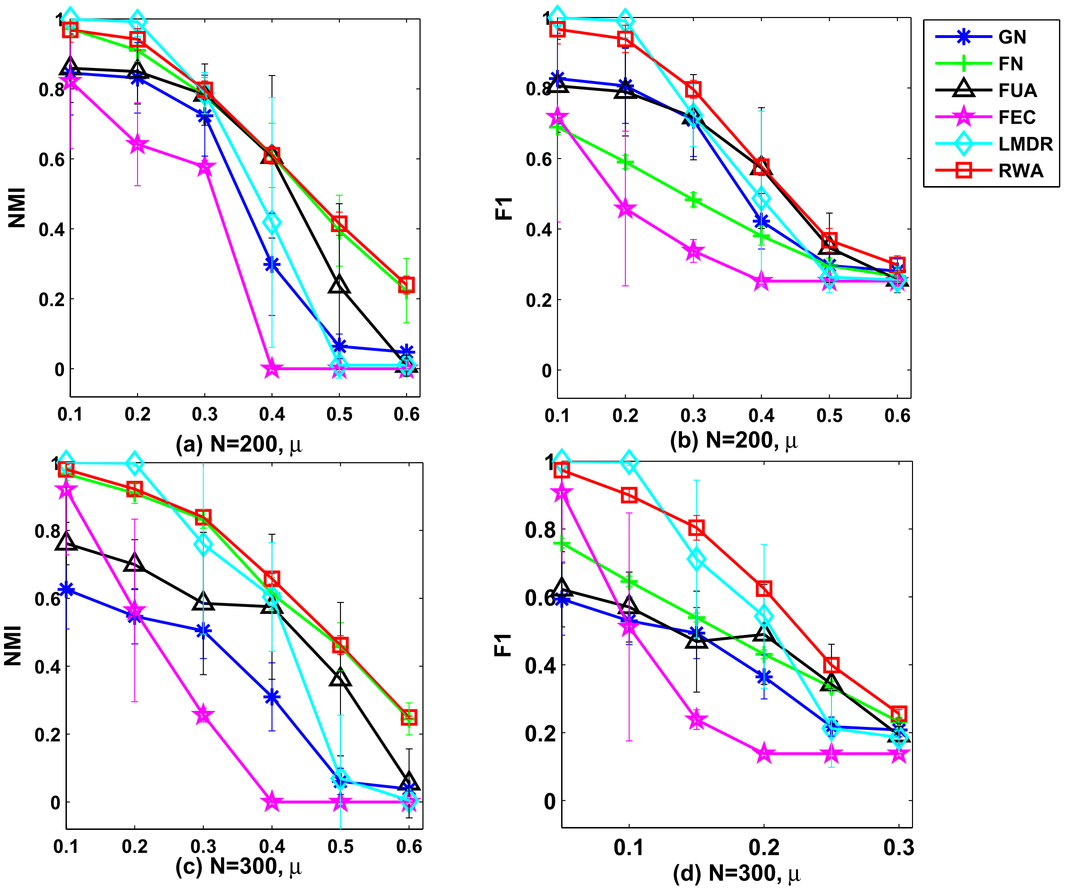

Figure 2. The comparative results on the LFR benchmark networks. Each point is the mean of *NMI* and *F*1 values averaged over 30 independent runs. Error bars show the standard deviations estimated from 30 networks.

(2) In our work, the performance of *RWA* is also compared with other five algorithms on two groups of the LFR benchmark networks. Figure 2 shows the average results over 30 runs on LFR benchmark networks. It can be seen form Fig. 2(a,c) that, when the value of μ is smaller than or equal to 0.2, the *NMI* obtained by *RWA* is larger than 0.9, but it is less well than *LMDR*. It suggests that although *RWA* gets the nearly true partition results, its performance is not the best. As μ increases, the *NMI* obtained by *RWA* remains relatively stable and *RWA* obtains the best results when μ is greater than or equal to 0.3. Similarly, *RWA* generates the largest and stablest value of *F*1 when μ is larger than or equal to 0.3 (see Fig. 2(b,d)). The value of *NMI* obtained by *RWA* is a slightly larger than that obtained by *FN*. However, compared with *FN*, *RWA* generates much larger value of *F*1. It is concluded that the performance of *RWA* is superior to the comparative algorithm on the LFR benchmark networks.

(3) All algorithms run 30 times on the four real-world networks, and the average *NMI* values and the average *F*1 values are shown in Fig. 3. As can be seen from Fig. 3(a), *RWA* generates significantly better results than the comparative algorithms. Specifically, *RWA* can achieve the largest *NMI* values on the four real-world networks ($p\_value < 0.05$). Similarly, as can be seen from Fig. 3(b), the average *F*1 values obtained by *RWA* are also larger than the comparative algorithms on the four real-world networks ($p\_value < 0.05$). Therefore, the proposed algorithm achieves the best detection results when tested on the four benchmark networks.

The communities in a real-world network could be divided into two classes: strong and weak communities. For each class, we count the number of times that each algorithm shows the best performance. As we can see from Table 2, one of the comparative algorithms shows better performance than the proposed algorithm in ≤60% of all strong communities and ≤22.22% of all weak communities. However, in 60% of all strong communities and 77.78% of all weak communities, *RWA* shows the best performance. That is, *RWA* surpasses previously proposed algorithms in most cases. We can conclude that *RWA* performs better than the comparative algorithms in both strong and weak communities, particularly in weak communities.

Selection of the parameter Z. In the proposed algorithm, a seed community is used as a seed by extending it to a larger community. The nodes of the seed community should be connected as densely as possible. To this end, we choose a complete subgraph as a seed community. Due to the fact that a complete subgraph with one node or two nodes is meaningless, we only consider complete subgraphs consisting of three or more nodes in

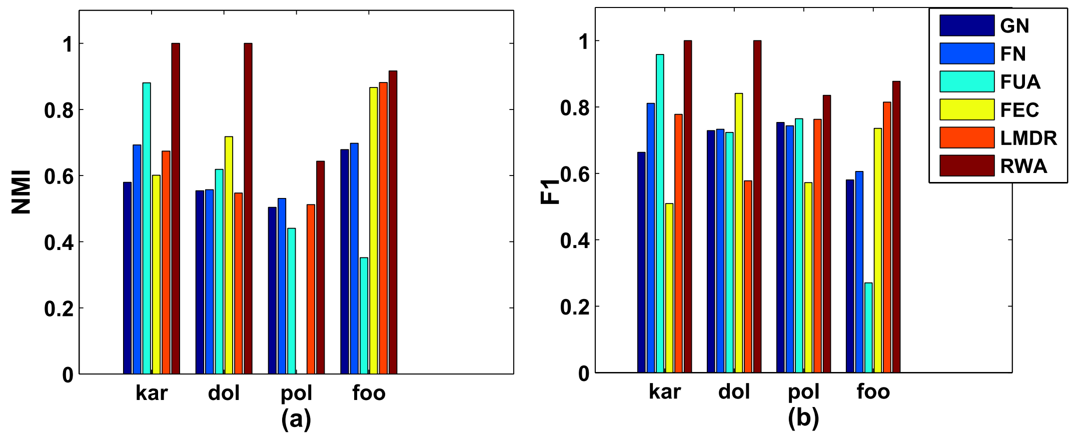

Figure 3. The comparative results on real-world networks. 'kar', 'dol', 'pol' and 'foo' represent the Zachary's Karate Club network, the Bottlenose Dolphins network, the Books about US politics network and the American College Football network, respectively.

	Karate		Dolphins		Polbooks		Football		Total	
	s(0)	w(2)	s(1)	w(1)	s(1)	w(2)	s(8)	w(4)	s(10)	w(9)
GN	0	0	0	1	0	0	1	0	10%	11.11%
FN	0	0	0	0	0	0	1	0	10%	0
FUA	0	2	0	0	0	0	0	0	0	22.22%
FEC	0	0	0	0	0	0	3	2	30%	22.22%
LMDR	0	0	1	0	0	0	5	1	60%	11.11%
RWA	0	2	1	1	1	1	5	3	60%	77.78%

Table 2. The performance in strong and weak communities of real-world networks. 'Karate', 'Dolphins', 'Polbooks' and 'Football' represent the Zachary's Karate Club network, the Bottlenose Dolphins network, the Books about US politics network and the American College Football network, respectively. 's(*)' represents the number of strong communities is '*' in a specific network. 'w(**)' represents the number of strong communities is '**'. 'Total s(10)' means the total number of strong communities in four real-world networks is 10, and 'Total w(9)' means the total number of weak communities in four real-world networks is 9.

this work. Let Z be the number of nodes in a seed community. In the following, we investigate the influence of $Z \geq 3$ on the performance of *RWA* in our work.

We do experiments on computer-generated and real-word networks. The results of *NMI* values and *F*1 values for different settings of Z ($Z \in \{3, 4, \cdots, 9\}$) on computer-generated networks ($P_{in} \in \{0.4, 0.5, \cdots, 0.9\}$) are shown in Fig. 4(a,b), averaging over 30 independent runs. According to Fig. 4(a), if P_{in} is either 0.6 or 0.7, then the value of *NMI* is the largest when Z is 3, and it is little larger than those when $Z \in \{4, 5, 6, 7, 8, 9\}$; if $P_{in} \in \{0.4, 0.5, 0.8, 0.9\}$, the values of *NMI* are the same, regardless of what Z is. Similarly, if P_{in} is either 0.6 or 0.7, when Z is 3, our algorithm produces the largest *F*1; otherwise, the value of *F*1 is unrelated with Z. Thus, the values of *NMI* and *F*1 have low sensitivity of Z when the experiments are tested on computer-generated networks. Figure 4(c,d) show the results of *NMI* values and *F*1 values for different settings of Z ($Z \in \{3, 4, \cdots, 9\}$) on four real-word networks. We can see that, on Dolphin network, the values of *NMI* (*F*1) are the same, regardless of what Z is; and on Karate, Polbooks, and Football networks, the best performance has been achieved when Z is 3.

Sensitivity analysis of this parameter shown in Fig. 4 has indicated that complete subgraphs with three nodes can achieve the best performance of the proposed algorithm. To obtain the best performance of *RWA*, Z can be set to 3.

Discussion

In this paper, we have proposed the algorithm *RWA* to detect community structure in a network, especially for the network with ambiguous community structure. In order to avoid the instability of detection results, seed communities were detected based on local maximal degree nodes, which have relatively high degree compared with their neighbors. In addition, the seed communities were expanded through random walks by adding nodes step by step.

We have test the performance of the proposed algorithm, and compared it with other representative algorithms on both computer-generated and real-world networks. (1) The experimental results have demonstrated the superior performance of *RWA* over the comparative algorithms (*GN*, *FN*, *FUA*, *FEC* and *LMDR*) in terms of *NMI* and *F*1 for detecting communities. An interesting observation was that the proposed algorithm surpassed five previously proposed algorithms in detecting weak communities in real-world networks. It is concluded that the performance of *RWA* showed more advantages in the networks which have ambiguous community structures,

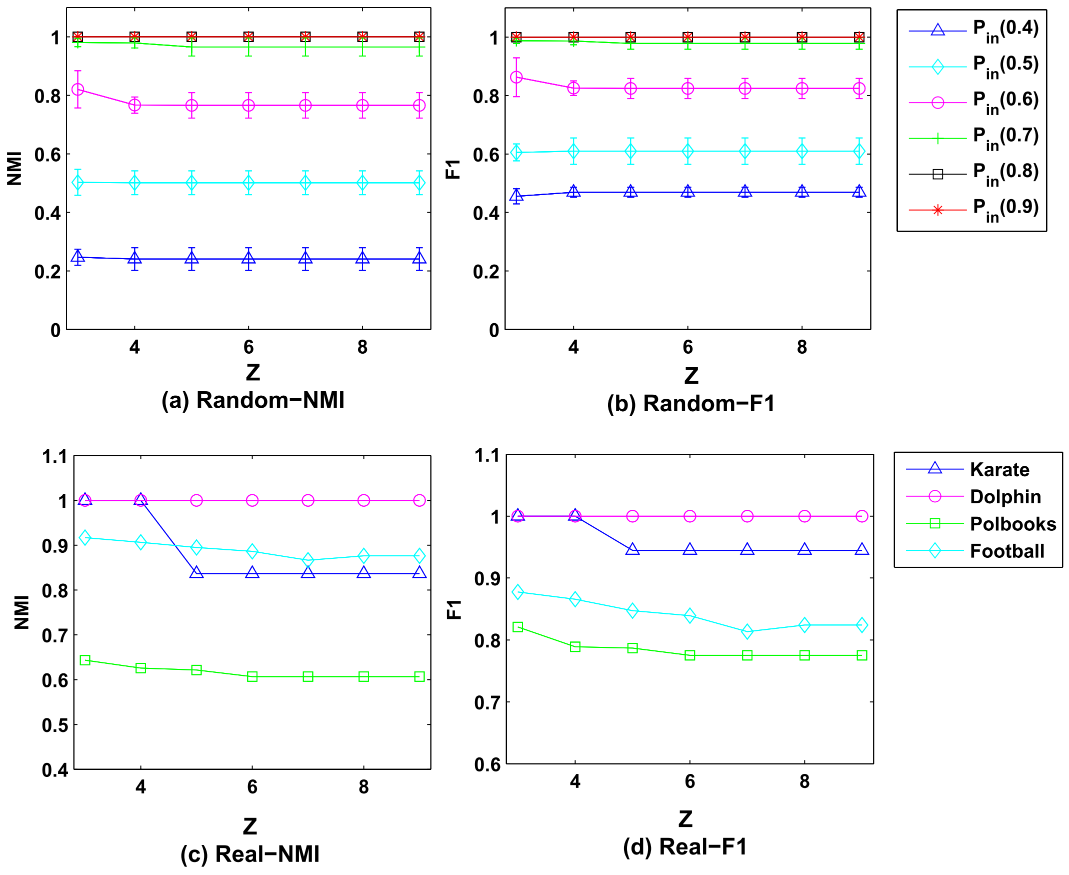

Figure 4. Sensitivity of *Z*. Each point is the mean of *NMI* and *F*1 values averaged over 30 independent runs and error bars show the standard deviations estimated from 30 networks.

when compared with the comparative algorithms. (2) An initial community is a dense subgraph with *Z* nodes. The experimental results have demonstrated that the proposed algorithm showed good performance with low sensitivity of *Z*. Furthermore, if *Z* is equal to three, then the proposed algorithm gained the best results. Therefore, we adopt $Z = 3$ in our work. In total, the experimental results have showed the effectiveness and robustness of the proposed algorithm. These experimental results confirmed that the proposed algorithm might be more suitable for the community detection of the complex networks with ambiguous community structures.

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. We will extend the algorithm to detect overlap communities. In addition, we will improve the detection accuracy, so that the algorithm can detect community structures efficiently.

Methods

The proposed algorithm (*RWA*) aims to select the dense subgraphs which contain important nodes in the network, and expand these dense subgraphs based on random walks. The overall framework of the proposed algorithm (*RWA*) contains the following three steps: (1) A procedure is proposed to detect seed communities based on local maximal degree nodes. These local maximal degree nodes have relatively high degree compared with their neighbors and locate dispersedly in the network, which could be considered as a local hub of a community[14]. (2) A strategy is applied to expand seed communities using random walks. In the expansion process, we calculate the probability of a node in a community based on random walks, and then add the node to the community which it most likely belongs to. A community may have more than one seed community, so that the expanded communities which have a large number of common nodes are deserved to be merged. (3) The expanded communities are adjusted to ensure each node in a network is in a community. In what follows, we introduce the details about *RWA*.

Detecting seed communities.

The basic idea of seed-based community detection algorithms includes the identification of the seeds, which are special nodes in networks[25]. From a topological point of view, a single seed may be a set of nodes which are not necessarily connected[18,26], or a set of nodes which are closely connected[27]. For instance, the seed is proposed to be random nodes in a network[28]. However, it does not use the topological information of the real-world networks. Generally speaking, the nodes which suit to constitute a seed are always the important nodes in a network. The seed has been proposed to be composed of the top *k* highest degree nodes, which playing the role of leaders in the network (i.e., the nodes whose removal from the network implies community collapse)[18,26]. Besides, the local hubs, such as the nodes with local maximal degree in a network, are selected as seeds[29,30]. The seed is also proposed to be a core set, in which the nodes are densely connected based on structural similarity[27].

Here, a seed community includes the important node which is most likely in a community, as well as the nodes and edges which are closely connected with the important node. Thus, a single seed is no longer a set of nodes, and it is a dense subgraph in a network. In what follows, the important nodes in a network are identified first, and then the dense subgraphs are detected.

A local maximal degree node is defined as a node which has a larger number of edges compared with its neighbors in a network[14]. Here, we identify local maximal degree nodes from all nodes in a complex network. The way to discover local maximal degree nodes from a given starting node was referred in a pervious work[14].

We detect the dense subgraphs based on the local hub set, which is a union of all local maximal degree nodes in a complex network. For the node ($node_1$) in the local hub set, we detect its local maximal degree nodes. The node ($node_1$) and one of its local maximal degree node ($node_2$) may have a common neighbor node ($node_3$). A dense subgraph with three nodes is comprised by the nodes $node_1$, $node_2$ and $node_3$, together with the edges among them. In this way, a dense subgraph with more than three nodes may also be detected. We analyze the influence of the number of nodes in a seed community on the performance of the proposed algorithm. Here we choose the dense subgraph with three nodes to be a seed community.

Expanding seed communities. Let $Y = \{Y_k | k = 1, \cdots, q\}$ be the set of all communities, where $Y_k = (V_k, E_k)$ is the k^{th} community, V_k is the set of nodes in the k^{th} community, E_k is the set of edges in the k^{th} community and q is the number of communities. Particularly, in the initial situation, $Y_k (k = 1, \cdots, q)$ is a seed community.

Let the walker start from a node u which does not belong to any communities. The total probability theorem and conditional probability model are used to calculate the probabilities of the walker teleporting from the node u to each community (i.e. $p(u \to Y_k)(k = 1, \cdots, q)$). A community is expanded by iteratively adding the nodes which has the largest probability to reach the community. There are q communities, so we perform q runs of random walks to calculate $p(u \to Y_k)$.

At the k^{th} run of random walks, it is supposed that u belongs to the k^{th} community. The graph of the k^{th} random walk process is:

$$G_k = (V_k', E_k'), \tag{1}$$

where $V_k' = (\cup_{t=1}^{q} V_t) \cup \{u\}$, $E_k' = (\cup_{t=1}^{q} E_t) \cup \{(u, v_i) | v_i \in V_k, 1 \le k \le q\}$.

First, we calculate the probability of the walker teleporting from u to the node $v_i \in \cup_{t=1}^{q} V_t$ in the graph G_k, which is denoted as $p(u \to v_i | u \in G_k)$. From the time t to the time $t+1$, the walker has a teleporting probability α to jump, as well as a probability $1-\alpha$ to stay. Usually, the teleporting probability α is 0.15[31]. When the walker jumps, it may jump to a node with a transition probability. Suppose that the transition probability for the walker jumping from u to each node in $\cup_{t=1}^{q} V_t$ is the same, then the transition probability vector is $d = \left(\frac{1}{m}, \frac{1}{m}, \cdots, \frac{1}{m}\right)^T$, where m is the number of nodes in the node-set $\cup_{t=1}^{q} V_t$ and d is a $m \times 1$ vector. When the walker stays, it may reach a node on the basis of the similarity between nodes (See the 'Calculation of similarity' subsection for the way to calculate the similarity between nodes). Let the matrix M with dimension of $m \times m$ denote the normalization of similarity between nodes in V. Suppose the probability of the walker teleporting from u to v_i is $s_t(i)$ at the time t. At the time $t+1$, the probability vector s_{t+1} is calculated as follows.

$$s_t = (1-\alpha) \cdot M^T \cdot s_{t-1} + \alpha \cdot d, \tag{2}$$

where M^T is the transpose of the matrix M, and $t \ge 1$. Particularly, in the initial situation, the probability of u teleporting to v_i is proportional to the similarity between u and v_i (See the 'Calculation of similarity' subsection). Here, $s_0(i)$ is the normalization of the similarity between u and v_i.

Iterate the Eq. (2) until s is convergent. Suppose the distribution vector is $\pi = (\pi_1, \ldots, \pi_m)$, then π satisfies $\pi = (1-\alpha) \cdot M^T \cdot \pi + \alpha \cdot d$. In this situation, π is the stationary distribution, where the i^{th} entry captures the conditional probability that the walker teleports from the node u to the node v_i when u belongs to the k^{th} community.

Next, the walker has an average conditional probability $p(u \to Y_j | u \in G_k)$ to teleport from the node u to a community Y_j when u belongs to the k^{th} community. Specifically, $p(u \to Y_j | u \in G_k)$ is the average value of the conditional probabilities.

$$p(u \to Y_j | u \in G_k) = avg(\{p(u \to v_i | u \in G_k) | v_i \in V_j\}), \tag{3}$$

where $p(u \to v_i | u \in G_k) = \pi_i$ and $avg(x)$ means the average value of the elements in the set x.

Finally, the average probability that the node u belongs to the k^{th} community is calculated as:

$$p(u \in G_k) = avg(\{Similar(u, v_i) | \forall v_i \in V_k'\}), \tag{4}$$

where $Similar(u, v_i)$ is the similarity between nodes u and $v_i \in V_{k'}$ (See the 'Calculation of similarity' subsection for the calculation of $Similar(u, v_i)$) and $avg(x)$ means the average value of the elements in set x.

According to Eq. (3) and Eq. (4), the probability of the walker teleporting from u to Y_j, denoted as $p(u \to Y_j)$ is calculated as Eq. (5).

Input	**Node-set $V = \{v_1, \ldots, v_m\}$, a node u and the set of communities $Y = \{Y_1, \ldots, Y_q\}$, where v_i represent the node included in a community, and q is the number of communities.**
Output	**The probability vector for the node u in each community $P(u \to Y) = (p(u \to Y_1), \ldots, p(u \to Y_q))$**
Step 1	Initialize an array *PC* with dimension of $m \times q$ (Save the conditional probability that the walker teleports from the node u to the node v_i when the node u belongs to the k^{th} community); Initialize an array *PP* with dimension of $q \times 1$ (Save the probability for the node u in the community G_k).
Step 2	For $k = 1$ to q do
Step 3	Construct the graph G_k;
Step 4	Calculate the matrix M and the initial vector s_0;
Step 5	Iterate the Eq. (2) until s is convergent, and the probability vector $\pi = (\pi_1, \ldots, \pi_m)$ is s;
Step 6	Calculate $p(u \to Y_i \mid u \in G_k)$ $(i = 1, \cdots, q)$, and then $PC(k, i) = p(u \to Y_i \mid u \in G_k)$;
Step 7	Calculate $p(u \in G_k)$, and then $PP(k) = p(u \in G_k)$;
Step 8	End For
Step 9	Normalize *PC* and *PP*; Calculate $p(u \to Y_i)$: $p(u \to Y_i) = PC \times PP$;
Step 10	Return $P(u \to Y) = (p(u \to Y_1), \ldots, p(u \to Y_q))$.

Table 3. The algorithm to calculate the probability of a node belonging to each community.

$$p(u \to Y_j) = \sum_{k=1}^{q} [p(u \to Y_j \mid u \in G_k) \times p(u \in G_k)]. \tag{5}$$

The algorithm to calculate the probability of a node belonging to each community is described in Table 3. A community is expanded by iteratively adding the node which is the most likely to belong to the community.

Community optimization. Each node in a connected network should be involved into a community, but several nodes with very low degree may still be not included in any communities. In other words, the node which is not added into a communities always has small number of neighbors. Given the node u which is not added into any communities and the community Y_k, denoting by $T(u, Y_k)$ that the tightness between the node u and the community $Y_k (1 \le k \le q)$, we have

$$T(u, Y_k) = \frac{num_1}{num_2}, \tag{6}$$

where num_1 denotes the number of nodes which have connections with the node u in the community Y_k, and num_2 is the number of nodes in the community Y_k. The node is added to the community which has the largest tightness with it.

Two or more of the expanded communities may have a large number of common nodes. The communities which are expanded from different communities may be identical or similar, in which case the expanded communities should be merged into one community. If two communities C_i and C_j satisfy the following formula, then they can be merged into a larger community C.

$$\frac{|C_i \cap C_j|}{min(|C_i|, |C_j|)} > \xi, \tag{7}$$

where ξ is a threshold. Let $\xi = 0.5$, meaning that most members of the small community are in the large community, the two communities can be merged into one.

Time complexity. In this section, we analyze the time complexity of the proposed algorithm. The time complexity is $O(dN)$ to find local maximum degree nodes in a network, where N is the number of nodes in the network and d is the average degree of nodes. At the stage of detecting seed communities, the time used to detect seed communities based on local maximum degree nodes is $O(dp)$, where p is the number of local maximum degree nodes. At most, there are p seed communities. At the stage of expanding communities, it needs to calculate the probability that a node teleports to each node in communities based on an iterative formula. It takes a time complexity of $O(logm)$ in each iteration as stated in ref. 32, where m is the number of nodes in the communities. The worst-case complexity is $O(logN)$. The time complexity of the stage after p iterations is $O(plogN)$. At the stage of community optimization, a small number of nodes which are not in any communities needs to be added to a community based on the tightness. The time complexity is $O(ph)$ to calculate the tightness between a node and a community, where h represents the number of nodes which are not in any communities. Therefore, the time complexity of the entire algorithm is $O((d + p)N)$, since $O(dp) = O(dN)$, $O(ph) = O(pN)$ and $O(plogN) < O(pN)$.

Calculation of similarity. We calculate the similarity between the nodes $v_i \in V$ and $v_j \in V$ $(1 \le i, j \le m)$ as follows[33].

$$Similar(v_i, v_j) = \frac{|\Gamma_{v_i} \cap \Gamma_{v_j}|}{|\Gamma_{v_i}| \cup |\Gamma_{v_j}|}, \tag{8}$$

where Γ_{v_i} (Γ_{v_j}) is the neighborhood of v_i (v_j) in a network, and $|x|$ indicates the cardinality (i.e., number of elements in) the set x.

In our work, the similarity between nodes is used to calculate the matrix M and the initial probability vector s_0. The similarity between nodes is normalized to obtain the matrix M, i.e., $M(i, j) = \frac{Similar\,(v_i, v_j)}{\sum_{v_j} Similar\,(v_i, v_j)}$. Let $v_j = u$ in Eq. (8). The similarity between nodes u and $v_i \in V$ ($1 \le i \le m$) is calculated, and it is denoted as $Similar(v_i, u)$ ($Similar(v_i)$ for short). The initial probability vector s_0 is the normalization of vector $Similar(v_i)$ (i.e., $s_0(i) = \frac{Similar\,(v_i)}{\sum_{v_i} Similar\,(v_i)}$).

Evaluation measures. For networks whose true partitions are known, Normalized Mutual Information (NMI)[34] and the F-measure ($F1$)[14] are widely used indexes for measuring the performance of community detection algorithms[1,35,36]. Both of them reflect the detection results from different points of view. Thus, both NMI and $F1$ are employed here as indexes to test the detection results.

NMI is defined as follows:

$$NMI\,(P_R, P_F) = \frac{-2\sum_i\sum_j X_{ij}\log\left(\frac{X_{ij}N}{X_{i.}X_{.j}}\right)}{\sum_i X_{i.}\log\left(\frac{X_{i.}}{N}\right) + \sum_j X_{.j}\log\left(\frac{X_{.j}}{N}\right)}, \tag{9}$$

where N is the number of nodes, X is a 2×2 matrix with X_{ij} being the number of nodes from the real community i that also belong to the found community j, $X_{.j} = X_{1j} + X_{2j}$, and $X_{i.} = X_{i1} + X_{i2}$. If the partitioning result P_F is the same as P_R, then $NMI(P_R, P_F) = 1$; if they are completely opposite, then $NMI(P_R, P_F) = 0$.

The *precision* is the ratio of the number of identified nodes which belong to the true community and the number of nodes in a discovered community[14]. The *recall* is the fraction of identified nodes which belong to the true community in the true community[14]. $F1$ is the combination of the *precision* and the *recall*, and it is calculated as follows:

$$F1 = \frac{prescision \times recall}{\frac{1}{2} \times (precision + recall)}. \tag{10}$$

The *precision* and the *recall* only reflect one aspect of the performance of an algorithm. However, $F1$ is the combination of *precision* and *recall*, and it takes the performance of an algorithm into a comprehensive consideration. Therefore, $F1$ is of more comparative significance, compared with *precision* and *recall*.

References

1. Shang, R., Luo, S., Li, Y., Jiao, L. & Stolkin, R. Large-scale community detection based on node membership grade and sub-communities integration. *Physica A: Statistical Mechanics and Its Applications* **428,** 279–294 (2015).
2. Fortunato, S. Community detection in graphs. *Physics Reports* **486,** 75–174 (2010).
3. Conaco, C. *et al.* Functionalization of a protosynaptic gene expression network. *Proceedings of the National Academy of Sciences* **109,** 10612–10618 (2012).
4. Dourisboure, Y., Geraci, F. & Pellegrini, M. Extraction and classification of dense communities in the web. In *Proceedings of the 16th international conference on World Wide Web*. 461–470 (ACM, New York, 2007).
5. Newman, M. E. J. & Girvan, M. Finding and evaluating community structure in networks. *Physical Review E* **69,** 026113 (2004).
6. Castellano, C., Cecconi, F., Loreto, V., Parisi, D. & Radicchi, F. Self-contained algorithms to detect communities in networks. *The European Physical Journal B-Condensed Matter and Complex Systems* **38,** 311–319 (2004).
7. Hu, Y. *et al.* Comparative definition of community and corresponding identifying algorithm. *Physical Review E* **78,** 026121 (2008).
8. Girvan, M. & Newman, M. E. J. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences* **99,** 7821–7826 (2002).
9. Newman, M. E. J. Fast algorithm for detecting community structure in networks. *Physical Review E* **69,** 066133 (2004).
10. Rosvall, M. & Bergstrom, C. T. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* **105,** 1118–1123 (2008).
11. Raghavan, U. N., Albert, R. & Kumara, S. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E* **76,** 036106 (2007).
12. Li, S., Chen, Y., Du, H. & Feldman, M. W. A genetic algorithm with local search strategy for improved detection of community structure. *Complexity* **15,** 53–60 (2010).
13. Blondel, V. D., Guillaume, J.-L., Lambiotte, R. & Lefebvre, E. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment* **2008,** P10008 (2008).
14. Chen, Q., Wu, T. & Fang, M. Detecting local community structures in complex networks based on local degree central nodes. *Physica A: Statistical Mechanics* and *Its Applications* **392,** 529–537 (2013).
15. Kernighan, B. W. & Lin, S. An efficient heuristic procedure for partitioning graphs. *Bell System Technical Journal* **49,** 291–307 (1970).
16. Luo, F., Wang, J. Z. & Promislow, E. Exploring local community structures in large networks. *Web Intelligence and Agent Systems* **6,** 387–400 (2008).
17. Lancichinetti, A., Fortunato, S. & Kertész, J. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics* **11,** 033015 (2009).
18. Whang, J. J., Gleich, D. F. & Dhillon, I. S. Overlapping community detection using seed set expansion. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, 2099–2108 (ACM, 2013).
19. Yang, B., Cheung, W. & Liu, J. Community mining from signed social networks. *IEEE Transactions on Knowledge and Data Engineering* **19,** 1333–1348 (2007).
20. Abrahao, B., Soundarajan, S., Hopcroft, J. & Kleinberg, R. On the separability of structural classes of communities. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 624–632 (ACM, 2012).
21. Lancichinetti, A., Fortunato, S. & Radicchi, F. Benchmark graphs for testing community detection algorithms. *Physical Review E* **78,** 561–570 (2008).
22. Zachary, W. W. An information flow model for conflict and fission in small groups. *Journal of Anthropological Research* **33,** 473 (1977).

23. Lusseau, D. The emergent properties of a dolphin social network. *Proceedings of the Royal Society B Biological Sciences* **270** suppl 2, 186–188 (2003).
24. Newman, M. E. J. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences* **103,** 8577–82 (2006).
25. Symeon, P., Yiannis, K., Athena, V. & Ploutarchos, S. Community detection in social media, performance and application considerations. *Journal of Data Mining Knowledge Discovery* **24,** 515–554 (2012).
26. Khorasgani, R. R., Chen, J. & Zaane, O. R. Top leaders community detection approach in information networks. In *4th SNA-KDD Workshop on Social Network Mining and Analysis* (Citeseer, 2010).
27. Papadopoulos, S., Kompatsiaris, Y. & Vakali, A. A graph-based clustering scheme for identifying related tags in folksonomies. In *Data Warehousing and Knowledge Discovery* 65–76 (Springer, 2010).
28. Andersen, R. & Lang, K. J. Communities from seed sets. In *Proceedings of the 15th international conference on World Wide Web*, 223–232 (ACM, 2006).
29. Chen, Q. & Fang, M. An efficient algorithm for community detection in complex networks. In *the 6th Workshop on Social Network Mining and Analysis* (2012).
30. Gleich, D. F. & Seshadhri, C. Vertex neighborhoods, low conductance cuts, and good seeds for local community methods. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, 597–605 (ACM, 2012).
31. Zhang, L., Wu, J., Zhuang, Y., Zhang, Y. & Yang, C. Review-oriented metadata enrichment: A case study. In *Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries*, 173–182 (ACM, New York, 2009).
32. Krishnan, V. & Lakshmivarahan, S. Probability and random processes. *Journal of the Royal Statistical Society* **40,** 164–165 (2001).
33. Jaccard, P. Etude comparative de la distribution dans une portion des alpes et du jura. *Bulletin de la Societe Vaudoise des Sciences Naturelle* **4** (1901).
34. Bagrow, J. P. Evaluating local community methods in networks. *Journal of Statistical Mechanics: Theory and Experiment* **5,** P05001 (2008).
35. Larremore, D. B., Clauset, A. & Jacobs, A. Z. Efficiently inferring community structure in bipartite networks. *Physical Review E* **90,** 012805 (2014).
36. Sah, P., Singh, L. O., Clauset, A. & Bansal, S. Exploring community structure in biological networks with random graphs. *BMC Bioinformatics* **15,** 220 (2014).

Acknowledgements
This work was supported by National Natural Science Foundation of China (61272152, 61320106005, 61472002, 61502004, 61672033), Natural Science Foundation of Anhui Higher Education Institutions of China (KJ2012A010).

Author Contributions
X.Z., B.W. and Y.S. designed the study; X.Z., B.W. and Y.S. performed the experiments, analyzed the data and prepared the figures; B.W. and Y.S. wrote the paper. All authors reviewed the manuscript.

Additional Information
Competing financial interests: The authors declare no competing financial interests.

How to cite this article: Su, Y. *et al.* A seed-expanding method based on random walks for community detection in networks with ambiguous community structures. *Sci. Rep.* **7**, 41830; doi: 10.1038/srep41830 (2017).