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. 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. 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. 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. 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. Popular algorithms include modularity-based algorithms (e.g. Girvan-Newman algorithm (GN), fast Newman algorithm (FN) and Fast unfolding algorithm (FUA)) and local modularity-based algorithms (e.g. local maximum degree algorithm (LMDR)). 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. 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 of nodes.
of edges and on the degree of interconnectedness of the modules. 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 and an algorithm for finding and extracting a community (FEC)), 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. 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. 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. 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 \( P_{in} \in \{0.40, 0.45, \ldots, 0.90\} \), 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. 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 \( \mu \) is the probability of nodes connected with nodes of external community. The bigger \( \mu \) is, the more difficult the community detection is. When \( \mu \geq 0.3 \), 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 \( \mu \) 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), the Bottlenose Dolphins network (Dolphins network, for short), the Books about US politics network (Polbooks network, for short) and the American College Football network (Football network, for short), 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.
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 \( \binom{0,4}{0,45,0,9} \). 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 (F1) values averaged over 30 independent runs for RWA and other five comparative algorithms.

As can be 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 F1 value obtained through RWA is no less than those obtained by other five comparative algorithms. In details, the F1 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 F1 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 F1 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 show that RWA has the best performance. When two evaluation measures (NMI and F1) are considered together, although the F1 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 F1 values of LMDR are larger 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 F1 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.

### Table 1. The number of strong and weak communities in real-world networks.

|      | Karate | Dolphins | Polbooks | Football |
|------|--------|----------|----------|----------|
| strong | 0      | 1        | 1        | 8        |
| weak  | 2      | 1        | 2        | 4        |
(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 from 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 F1 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 F1. 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 F1 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 F1 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 2. The comparative results on the LFR benchmark networks.](image-url)
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-world networks. The results of NMI values and \( F_1 \) values for different settings of \( Z \in \{3, 4, \ldots, 9\} \) on computer-generated networks \( (P_{m} \in \{0.4, 0.5, \ldots, 0.9\}) \) are shown in Fig. 4(a,b), averaging over 30 independent runs. According to Fig. 4(a), if \( P_{m} \) is either 0.6 or 0.7, then the value of NMI is the largest when \( Z = 3 \), and it is little larger than those when \( Z \in \{4, 5, 6, 7, 8, 9\} \); if \( P_{m} \in \{0.4, 0.5, 0.8, 0.9\} \), the values of NMI are the same, regardless of what \( Z \) is. Similarly, if \( P_{m} \) is either 0.6 or 0.7, when \( Z = 3 \), 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 \( Z \in \{3, 4, \ldots, 9\} \) on four real-world 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 \( Z = 3 \).

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-world networks. The results of NMI values and \( F_1 \) values for different settings of \( Z \in \{3, 4, \ldots, 9\} \) on computer-generated networks \( (P_{m} \in \{0.4, 0.5, \ldots, 0.9\}) \) are shown in Fig. 4(a,b), averaging over 30 independent runs. According to Fig. 4(a), if \( P_{m} \) is either 0.6 or 0.7, then the value of NMI is the largest when \( Z = 3 \), and it is little larger than those when \( Z \in \{4, 5, 6, 7, 8, 9\} \); if \( P_{m} \in \{0.4, 0.5, 0.8, 0.9\} \), the values of NMI are the same, regardless of what \( Z \) is. Similarly, if \( P_{m} \) is either 0.6 or 0.7, when \( Z = 3 \), 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 \( Z \in \{3, 4, \ldots, 9\} \) on four real-world 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 \( Z = 3 \).

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 tested 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 \( (GN, FN, FUA, FEC \text{ and } LMDR) \) 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,
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\(^1\). (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\(^2\). From a topological point of view, a single seed may be a set of nodes which are not necessarily connected\(^3\), or a set of nodes which are closely connected\(^4\). For instance, the seed is proposed to be random nodes in a network\(^5\). 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.

![Figure 4. Sensitivity of \( Z \). Each point is the mean of NMI and F1 values averaged over 30 independent runs and error bars show the standard deviations estimated from 30 networks.](image-url)
nodes, which playing the role of leaders in the network (i.e., the nodes whose removal from the network implies community collapse)\(^2\,^8\,^{26}\). Besides, the local hubs, such as the nodes with local maximal degree in a network, are selected as seeds\(^2\,^{25}\,^{26}\). The seed is also proposed to be a core set, in which the nodes are densely connected based on structural similarity\(^2\).

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\(^1\,^{14}\). 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 previous work\(^1\,^{14}\).

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\(_t\)) in the local hub set, we detect its local maximal degree nodes. The dense subgraphs are detected.

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 \( Y = \{ Y_k | k = 1, \ldots, q \} \) 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, \( Y_1(k = 1, \ldots, q) \) 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. \( p(u \rightarrow Y_j | k = 1, \ldots, q) \)). A community is expanded by iteratively adding the nodes which have the largest probability to reach the community. There are \( q \) communities, so we perform \( q \) runs of random walks to calculate \( p(u \rightarrow Y_j) \).

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:

\[
G_k = (V'_k, E'_k),
\]

where \( V'_k = (\bigcup_{j=1}^{k-1} V_j) \cup \{ u \} \), \( E'_k = (\bigcup_{j=1}^{k-1} E_j) \cup \{ (u, v) | v \in V_k, 1 \le k \le q \} \).

First, we calculate the probability of the walker teleporting from \( u \) to the node \( v \), \( \forall v \in \bigcup_{j=1}^{k-1} V_j \) in the graph \( G_k \), which is denoted as \( p(u \rightarrow v \mid u \in G_k) \). From the time \( t \) to the time \( t + 1 \), the walker has a teleporting probability \( \alpha \) to jump, as well as a probability \( 1 - \alpha \) to stay. Usually, the teleporting probability \( \alpha \) is 0.15\(^3\). 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 \( \bigcup_{j=1}^{k-1} V_j \) is the same, then the transition probability vector is \( d = \left( \frac{1}{m}, \frac{1}{m}, \ldots, \frac{1}{m} \right)^T \), where \( m \) is the number of nodes in the node-set \( \bigcup_{j=1}^{k-1} V_j \), 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:

\[
s_t = (1 - \alpha) \cdot M^T \cdot s_{t-1} + \alpha \cdot d,
\]

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, \( s_t(i) \) 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 \( \pi \) satisfies \( \pi = (1 - \alpha) \cdot M^T \cdot \pi + \alpha \cdot d \). In this situation, \( \pi \) is the stationary distribution, where the \( k \)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 \( p(u \rightarrow Y_j \mid u \in G_k) \) to teleport from the node \( u \) to a community \( Y_j \) when \( u \) belongs to the \( k \)th community. Specifically, \( p(u \rightarrow Y_j \mid u \in G_k) \) is the average value of the conditional probabilities:

\[
p(u \rightarrow Y_j \mid u \in G_k) = \text{avg} \{ p(u \rightarrow v \mid u \in G_k) \mid v \in V_j \},
\]

where \( p(u \rightarrow v \mid u \in G_k) = \pi \), and \( \text{avg}(x) \) 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:

\[
p(u \in G_k) = \text{avg} \{ \text{Similar}(u, v) \mid v \in V'_k \},
\]

where \( \text{Similar}(u, v) \) is the similarity between nodes \( u \) and \( v \) in \( V_c \) (See the ‘Calculation of similarity’ subsection for the calculation of \( \text{Similar}(u, v) \)) and \( \text{avg}(x) \) 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 \( p(u \rightarrow Y_j) \) is calculated as Eq. (5).
Input: Node-set $V = \{v_1, \ldots, v_N\}$, 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 $P(u \rightarrow Y) = (p(u \rightarrow Y_1), \ldots, p(u \rightarrow Y_q))$.

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 $k^\text{th}$ 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 $v_0$.

Step 5: Iterate the Eq. (2) until $s$ is convergent, and the probability vector $p = \{p_1, \ldots, p_q\}$ is $s$.

Step 6: Calculate $p(u \rightarrow Y_i|u \in G_k)$ ($i = 1, \ldots, q$, and then $PC(k, i) = p(u \rightarrow Y_i|u \in G_k)$).

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 $p(u \rightarrow Y_1)$; $p(u \rightarrow Y_i) = PC \times PP$.

Step 10: Return $P(u \rightarrow Y) = (p(u \rightarrow Y_1), \ldots, p(u \rightarrow Y_q))$.

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

$$p(u \rightarrow Y_i) = \sum_{k=1}^{q} [p(u \rightarrow Y_i|u \in G_k) \times p(u \in G_k)].$$

(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 in 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 $Y_i$, denoting by $T(u, Y_i)$ that the tightness between the node $u$ and the community $Y_i (1 \leq k \leq q)$, we have

$$T(u, Y_i) = \frac{\text{num}_1}{\text{num}_2},$$

(6)

where $\text{num}_1$ denotes the number of nodes which have connections with the node $u$ in the community $Y_i$, and $\text{num}_2$ is the number of nodes in the community $Y_i$. 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 $C$.

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

(7)

where $\xi$ 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 $O(\log m)$ in each iteration as stated in ref. 32, where $m$ is the number of nodes in the communities. The worst-case complexity is $O(\log N)$. The time complexity of the stage after $p$ iterations is $O(p\log N)$. 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 $O(p\log N) < O(pN)$.

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

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

(8)
where $\Gamma_{v_i}(\Gamma_v)$ is the neighborhood of $v_i \in \Gamma_v$ 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., $M(i, j) = \frac{\text{Similar}(v_i, v_j)}{\sum_{k} \text{Similar}(v_k, v_j)}$. Let $v_i = u$ in Eq. (8). The similarity between nodes $u$ and $v_i \in V \ (1 \leq i \leq m)$ is calculated, and it is denoted as $\text{Similar}(v_i, u)$ (Similar for short). The initial probability vector $s_0$ is the normalization of vector $\text{Similar}(v_i)$ (i.e., $s_0(i) = \frac{\text{Similar}(v_i)}{\sum_i \text{Similar}(v_i)}$).

**Evaluation measures.** For networks whose true partitions are known, Normalized Mutual Information (NMI) and the F-measure ($F_1$) are widely used indexes for measuring the performance of community detection algorithms. Both of them reflect the detection results from different points of view. Thus, both NMI and $F_1$ are employed here as indexes to test the detection results.

NMI is defined as follows:

$$\text{NMI}(P_B, P_F) = \frac{-2 \sum \sum X_{ij} \log \left( \frac{X_{ij} N}{N_i N_j} \right)}{\sum_i X_i \log \left( \frac{X_i N}{N} \right) + \sum_j X_j \log \left( \frac{X_j N}{N} \right)},$$

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

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. The recall is the fraction of identified nodes which belong to the true community in the discovered community. $F_1$ is the combination of the precision and the recall, and it is calculated as follows:

$$F_1 = \frac{\text{precision} \times \text{recall}}{\frac{1}{2} \times (\text{precision} + \text{recall})}.$$
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. Khosravani, 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 *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. & Lakshminarahan, 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).

**Publisher’s note:** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This work is licensed under a Creative Commons Attribution 4.0 International License. The images or other third party material in this article are included in the article’s Creative Commons license, unless indicated otherwise in the credit line; if the material is not included under the Creative Commons license, users will need to obtain permission from the license holder to reproduce the material. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/

© The Author(s) 2017