TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 04/10 pp35–46
DOI: 10.26599/TST.2018.9010053
Volume 23, Number 1, February 2018

Hierarchical Community Detection Based on Partial Matrix Convergence Using Random Walks

Wei Zhang, Feng Kong, Liming Yang, Yunfang Chen*, and Mengyuan Zhang

Abstract: Random walks are a standard tool for modeling the spreading process in social and biological systems. But in the face of large-scale networks, to achieve convergence, iterative calculation of the transition matrix in random walk methods consumes a lot of time. In this paper, we propose a three-stage hierarchical community detection algorithm based on Partial Matrix Approximation Convergence (PMAC) using random walks. First, this algorithm identifies the initial core nodes in a network by classical measurement and then utilizes the error function of the partial transition matrix convergence of the core nodes to determine the number of random walks steps. As such, the PMAC of the core nodes replaces the final convergence of all the nodes in the whole matrix. Finally, based on the approximation convergence transition matrix, we cluster the communities around core nodes and use a closeness index to merge two communities. By recursively repeating the process, a dendrogram of the communities is eventually constructed. We validated the performance of the PMAC by comparing its results with those of two representative methods for three real-world networks with different scales.

Key words: community detection; random walks; transition matrix; clustering

1 Introduction

Community is a common phenomenon in social networks, in which various members tend to form closely linked groups. In different contexts, these groups may be referred to as communities, clusters, cohesive subgroups, or modules. Community detection is the identification of a set of nodes in a network based on cluster structural characteristics by the analysis of the network topology and network node properties, in which nodes are tightly connected between internal, but not external nodes. Obviously, internal clusters communicate between individuals more frequently than do nodes in different external clusters.

How to locate community structures in complex networks has become a hot topic in many fields, including sociology, bio-informatics, and physics. Belonging to the same closely linked community, these nodes have a greater probability of having the same or similar properties. For example, the groups in a social network[1] are likely based on a common interest or background. For example, in the World Wide Web[2], community structures may have some common themes and relevant pages; in metabolic, cell, and genetic-related biochemical or neural networks[3], association implies the existence of similar features. These network collections can simplify the functional analysis of the overall network.

Because community detection research has important significance, many community detection algorithms[4] have been proposed. These algorithms[5] can be divided into several categories, including modular[6],

• Wei Zhang is with School of Computer Science, Nanjing University of Posts and Telecommunications and Jiangsu High Technology Research Key Laboratory for Wireless Sensor Networks, Nanjing 210023, China.
• Feng Kong, Liming Yang, and Yunfang Chen are with School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210023, China. E-mail: chenyf@njupt.edu.cn.
• Mengyuan Zhang is with Concordia Institution System Engineering (CIISE), Concordia University, Montreal QC H3G 1M8, Canada.
∗ To whom correspondence should be addressed.
Manuscript received: 2016-08-25; revised: 2016-10-18; accepted: 2016-10-20

spectral clustering[7], and dynamic community detection[8] algorithms. The most common is the modular algorithm, in which modules are used as a posterior measurement. Algorithms based on modules use different clustering methods, including greedy algorithms, simulated annealing, external optimization, and spectrum optimization to maximize the modularity. In addition, the spectral method is based on the eigenvalues and eigenvectors of the adjacency matrix in networks. Dynamic community detection involves the incorporation of the community timestamps model or snapshots of a network for the prediction of the next state.

Although many different community detection algorithms have been proposed, there remain some unresolved issues. When analyzing large-scale networks, most algorithms have low efficiency and high time complexity.

The random walk is irregular and is the mathematically ideal state of Brownian movement. In many areas, it has important applications, of which one well-known example in the computer field is Google's PageRank algorithm. In recent years, probabilistic methods based on random walks have been introduced to community detection. Using the random walk model to compute the similarity between two nodes and the concept of nodes jumping is similar to flow information from one node to another. Given a graph and an initial node, the random walk algorithm finds the next node based on transition probabilities. As such, the sequence of a random walk is a Markov chain. From the perspective of a smooth Markov chain process, using random walks in iterative calculations, we can count similar information nodes that they have a greater probability of belonging to the same community, and thus we can gradually define a community structure.

However, in the face of large-scale networks, computing transfer matrix iterations using the random walk method to achieve convergence requires excessive calculation. Yet, there is a property of power-law distribution in networks in which core nodes play an important role. Accordingly, we propose a hierarchical community detection algorithm based on the partial matrix convergence in random walks. This method is innovative in the following three respects:

(1) Utilizing the natural power-law distribution and the community structure in a network, we substitute core nodes for all the nodes in networks to measure the degree of stability of the transfer matrix.

(2) We use the error function of the transfer matrix convergence to dynamically determine the number of random walk steps. The use of an appropriate number of steps significantly reduces the computational complexity.

(3) We use the partial convergence transition matrix to cluster communities around core nodes and repeat to merge communities, based on the community quality evaluation index, to obtain a reasonable dendrogram of the communities.

This paper proceeds as follows: In Section 2, we introduce related work in community detection via the random walk. In Section 3, we present relevant theoretical foundations and our conceptual basis. In Section 4, we describe hierarchical community detection based on Partial Matrix Approximation Convergence (PMAC) of random walks. In Section 5, we present our experimental analysis results and in Section 6, we draw our conclusions and suggest future research directions.

2 Related Work

Community detection algorithms that have been developed recently relative to graph partitioning have opened a new realm in network analysis. Moreover, the probabilistic random walk method has been used to infer the structural properties of networks in community detection studies. The reason why random walk tools are well-suited to community clustering detection is obvious: they locally explore the neighborhood of a vertex and, with a reasonably high probability, trap in a cluster rather than jump to a different cluster.

Van Dongen[9] proposed the Markov CLuster process (MCL) to address the graph partition problem. MCL simulates the flow diffusion process of networks and partitions networks into clusters using predefined algebraic operations on Markov transition matrices. Let n be the number of nodes and the required matrix multiplication operation will take $O(n^3)$ time. The main limitation of MCL is that predefined parameters are difficult to choose and different community division parameters can produce some unexplained differences. Moreover, due to the uncertainty of the number of iterations, the stability and convergence performance of the algorithm will be significantly reduced in a large-scale network.

Zhou and Lipowsky[10] introduced a dissimilarity index based on the Euclidean commute time distance and the average first-passage time of walkers, and then used the index to design a hierarchical algorithm, called Netwalk. Unfortunately, their approach also requires $O(n^3)$ time and cannot manage networks with more than a few thousand vertices.

Pons and Latapy[11] put forward an algorithm called Walktrap, which is computationally efficient. For most real-world complex networks, the complexity is $O(n^2 \log n)$. The intuition of this algorithm is that random walks on a graph tend to become trapped into densely connected areas of corresponding communities. Therefore, the authors used some properties of random walks on graphs to measure the structural similarity between vertices and between communities, and defined a distance to iteratively merge the vertices into communities.

Rosvall and Bergstrom[12] proposed an information-theoretic algorithm, called Infomap, to detect network communities, in which a random walk model is adopted to simulate the process of information diffusion in networks, with the objective being to minimize the description length of the information from a geographical coding schema. In this way, they transformed the task of community detection into a coding problem of identifying partitions of networks that can minimize the description length of an infinite random walk process. Infomap optimizes this length through a greedy search method combined with a simulated annealing strategy, which usually converges slowly.

Alamgir and Von Luxburg[13] proposed the Multi-Agent Random Walk (MARW) in place of the random walk. MARW consists of several agents connected by a fixed length of rope. All agents move independently as in a standard random walk on the graph, but they are constrained to distances from each other of this fixed length at most. The core insight of this method is that it is harder for several agents to travel simultaneously over the bottleneck of a graph than it is for just one agent. Hence, the MARW has less probability of mistakenly merging two different clusters than does the original random walk.

To deal with the lack of consensus on how to quantify and rank the quality of partitions, Delvenne et al.[14] introduced the concept of partition stability, a measure of its quality as a community structure based on the clustered auto-covariance of a dynamic Markov process within the network. Because the stability is inherently dependent on the time scale of the graph, we can compare and rank partitions at each moment in time and establish time spans over which the partitions are optimal. Hence, Markov time acts effectively as an intrinsic resolution parameter that establishes a hierarchy of increasingly coarse communities. This method provides a unifying framework for several standard partitioning measures.

Backstrom and Leskovec[15] developed an algorithm based on supervised random walks that naturally combines information from the network structure with node-level and edge-level attributes. Because a random walker is more likely to visit nodes for which future links will be created, the goal of the supervised learning task is to learn a function that can assign appropriate strengths to edges.

Yang et al.[16] explored the nature of community structure from a probabilistic perspective and introduced a community detection algorithm, called Probabilistic Mining Communities (PMC). In PMC, community detection is modeled as a constrained quadratic optimization problem that can be efficiently solved by a random-walk-based heuristic. The PMC has heuristic and optimization phases. In its heuristic phase, PMC uses a random-walk-based heuristic to reduce the space occupied by candidate community structures of a given network. In its optimization phase, PMC searches for an optimal community structure in this reduced space by optimizing a constrained quadratic objective.

Fu et al.[17] proposed a scalable community detection method based on threshold random walkers, called CD-TRandwalk, which selects highly active nodes as seed nodes, and detects core communities by random walkers according to predefined thresholds. The threshold random walkers start from active seed nodes and randomly walk only to those nodes with association degrees higher than a given threshold. CD-TRandwalk is a two-stage community detection method. In the first stage, the core nodes of the communities are detected by the threshold random walk, and then the remaining non-core nodes are allocated according to a voting policy. The drawback of this method is that it is difficult to accurately determine an appropriate threshold value for different networks.

Park and Lee[18] proposed a clustering method for maximizing a new measure called group dependence. Group dependence quantifies how precise is a certain division of a graph in terms of its distance dependence. Built upon statistical dependence measures between points driven by Markovian transitions, group dependence incorporates the geometric structures of input data. The method provides an optimal aspect as theoretical justification based on the posterior transition probabilities of the input data.

In this paper, we propose a hierarchical community detection algorithm based on random walks, and as in the PMC[16] and CD-TRandwalk[17] algorithms, our proposed algorithm also uses a limited of number of iterations. The difference is that the number of iterations in our algorithm is determined by the degree of convergence of the transition probability matrix rather than a predefined fixed value. Because this decision is based on the degree of convergence of the core nodes rather than that of all the nodes in the transfer matrix, it is possible to more rapidly complete the iteration calculation process. In addition, although our proposed algorithm considers only the core nodes when judging whether the transition probability matrix is approximately stationary, all the nodes are involved in iteratively calculating the multi-step transition matrix. We simply relax the judgment conditions to accelerate the convergence process.

3 Preliminaries on Random Walks

Let $G = (V, E)$ denote a network graph, where $V = \{v_1, ..., v_n\}$ and $E = \{e_1, ..., e_m\}$ are the node and link sets, respectively. We only consider undirected graphs in this paper. The graph G is associated with its adjacency matrix $\boldsymbol{A}$, and in the matrix, $a_{ij} = 1$ if nodes i and j are connected, and $a_{ij} = 0$ otherwise. The degree $d(i) = \sum_j a_{ij}$ of node i is the number of its neighbors.

A discrete Markov chain is a sequence of random variables, X_1, X_2, X_3, ..., with the Markov property. In a Markov chain, if given the present state, the future and past states are independent. Formally, $\boldsymbol{P}\{X_{t+1} = x|X_1 = x_1, X_2 = x_2, ..., X_t = x_t\} = \boldsymbol{P}\{X_{t+1} = x|X_t = x_t\}$ and the possible values of X_i form a countable set called the state space of the chain.

A random walk is a mathematical formalization of a path that consists of a succession of random steps. Suppose node j is the neighbor of node i, and let p_{ij} denote the probability of going to node j from node i after one step. Then we have $P_{ij}{=}P_{ij}^{(1)}{=}a_{ij}/d(i)$, where $\boldsymbol{P}$ is the one-step transition matrix of X. For t steps on random walks, let $P_{ij}^{(t)}$ be the probability of going to node j from node i after t steps, where $\boldsymbol{P}^{(t)}$ is the t-step transition matrix of X.

In the matrices, $\boldsymbol{P} = \boldsymbol{D}^{-1}\boldsymbol{A}$, where $\boldsymbol{A} = (a_{ij})_{n\times n}$ and $\boldsymbol{D} = \text{diag}(d_1, ..., d_n)$ are the adjacency and degree matrices of G, respectively.

To traverse a number of nodes through t steps of random walks is to achieve a t-step transition probability matrix, formally, $\boldsymbol{P}^{(t)} = \boldsymbol{P}^t$, whereby the t-step transition probability matrix is equal to a t-time multiplication of the one-step transition probability matrix.

In the transition probability matrix, we have a well-known property of the random walk process[11]:

Property 1 When the length t of a random walker starting at node i to node j tends toward infinity, the probability of arriving at node j from any another node, $P_{ij}^{(t)}$, only depends on the degree of the destination node:

$$\forall i, \lim_{t\geqslant\infty} P_{ij}^{(t)} = d(j)/\sum_k d(k) \tag{1}$$

To prove this property, we need the following technical lemma:

Lemma 1 The eigenvalues of the matrix $\boldsymbol{P}$ are real and satisfy:

$$1 = \lambda_1 > \lambda_2 \geqslant \cdots \geqslant \lambda_n > -1.$$

Moreover, there exists an orthonormal family of vectors $(\boldsymbol{s}_\alpha)_{1\leqslant\alpha\leqslant n}$ such that the vectors $\boldsymbol{v}_\alpha = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{s}_\alpha$ and $\boldsymbol{u}_\alpha = \boldsymbol{D}^{\frac{1}{2}}\boldsymbol{s}_\alpha$ are, respectively, right and left eigenvectors associated with the eigenvalue λ_α:

$$\forall\alpha, \boldsymbol{P}\boldsymbol{v}_\alpha = \lambda_\alpha\boldsymbol{v}_\alpha,\ \boldsymbol{P}^{\mathrm{T}}\boldsymbol{u}_\alpha = \lambda_\alpha\boldsymbol{u}_\alpha,\ \forall\alpha, \forall\beta, \boldsymbol{u}_\alpha^{\mathrm{T}}\boldsymbol{v}_\beta = \delta_{\alpha\beta}.$$

Proof The matrix $\boldsymbol{P}$ has the same eigenvalues as its similar matrix $\boldsymbol{S} = \boldsymbol{D}^{\frac{1}{2}}\boldsymbol{P}\boldsymbol{D}^{-\frac{1}{2}} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$. The matrix $\boldsymbol{S}$ is real and symmetric, so its eigenvalues λ_α are real. $\boldsymbol{P}$ is a stochastic matrix ($\sum\limits_{j=1}^{n} P_{ij} = 1$), so its largest eigenvalue is $\lambda_1 = 1$. The graph G is connected and primitive, therefore we can apply the Perron-Frobenius theorem, which implies that $\boldsymbol{P}$ has a unique dominant eigenvalue. Therefore we have $|\lambda_\alpha| < 1$ for $2 \leqslant \alpha \leqslant n$.

The symmetry of $\boldsymbol{S}$ implies that there also exists an orthonormal family s of eigenvectors of $\boldsymbol{S}$ that satisfy $\forall\alpha, \forall\beta, \boldsymbol{s}_\alpha^{\mathrm{T}}\boldsymbol{s}_\beta = \delta_{\alpha\beta}$ (where $\delta_{\alpha\beta} = 1$ if $\alpha = \beta$ and 0 otherwise). We then directly obtain that the vectors $\boldsymbol{v}_\alpha = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{s}_\alpha$ and $\boldsymbol{u}_\alpha = \boldsymbol{D}^{\frac{1}{2}}\boldsymbol{s}_\alpha$ are, respectively, right and left eigenvectors of $\boldsymbol{P}$ satisfying

$$\boldsymbol{u}_\alpha^{\mathrm{T}}\boldsymbol{v}_\beta = \delta_{\alpha\beta}.$$

We can now prove Property 1.

Proof Lemma 1 makes it possible to write a spectral decomposition of the matrix $\boldsymbol{P}$: $\boldsymbol{P} = \sum_{\alpha=1}^{n} \lambda_\alpha \boldsymbol{v}_\alpha \boldsymbol{u}_\alpha^{\mathrm{T}}$ and $\boldsymbol{P}^t = \sum_{\alpha=1}^{n} \lambda_\alpha^t \boldsymbol{v}_\alpha \boldsymbol{u}_\alpha^{\mathrm{T}}$, so $P_{ij}^t = \sum_{\alpha=1}^{n} \lambda_\alpha^t \boldsymbol{v}_\alpha(i) \boldsymbol{u}_\alpha^{\mathrm{T}}(j)$.

When t tends towards infinity, all the terms $\alpha \geqslant 2$ vanish. It is easy to show that the first right eigenvector $\boldsymbol{v}_1$ is constant. By normalizing we have $\forall i, \boldsymbol{v}_1(i) = \frac{1}{\sqrt{\sum_k d(k)}}$ and $\forall j, \boldsymbol{u}_1(j) = \frac{d(j)}{\sqrt{\sum_k d(k)}}$. We obtain Property 1:

$$\lim_{t\to+\infty} P_{ij}^t = \lim_{t\to+\infty} \sum_{\alpha=1}^{n} \lambda_\alpha^t \boldsymbol{v}_\alpha(i) \boldsymbol{u}_\alpha(j) = \boldsymbol{v}_1(i)\boldsymbol{u}_1(j) = \frac{d(j)}{\sum_k d(k)}.$$

Based on the above proof, when the length of steps tends to infinity, the probability value depends only on the destination node value, rather than that of the initial node. Meanwhile, for the whole social network, it is reasonable to assume that with limited steps t, the probability of random walks within the community between nodes is greater than that of random walks among the communities, that is, $P_{ij}^{(t)} > P_{kj}^{(t)}$, where nodes i and j belong to same community, and node k belongs to another community.

4 PMAC Algorithm

4.1 Overview

In a real social networks interactive process, the initial behavior of people is often characterized by a lot of randomness. However, as time evolves, random relationships will tend toward a steady state and gradually form relatively stable social circles that we call communities. Inspired by this observable phenomenon, here, we adopt a Markov random walk model for the social network node, and we convert the links between nodes to random walk processes. Since social networks have a large number of nodes, generally, the time complexity of the random walk process grows exponentially with network size. To reduce the complexity of the process, we work from two aspects. First, we view the social network as a collection of coarse-grained units, or communities, rather than fine-grained units, or nodes. This means that we require community detection. Second, we propose the partial matrix convergence concept based on the core nodes, which is a trade-off between the computation time of multiple-step random walks, and the degree to which the transition probability matrix is stationary. The proposed method does not require an initial setup of the number of communities, and is not sensitive to the initial selection of core nodes.

In Fig. 1, we show the four main steps of our algorithm.

Step 1 Select the initial core nodes according to a node centrality metric.

Step 2 Compute the number of iterations, called the optimal steps, using an error index, and obtain the true core nodes.

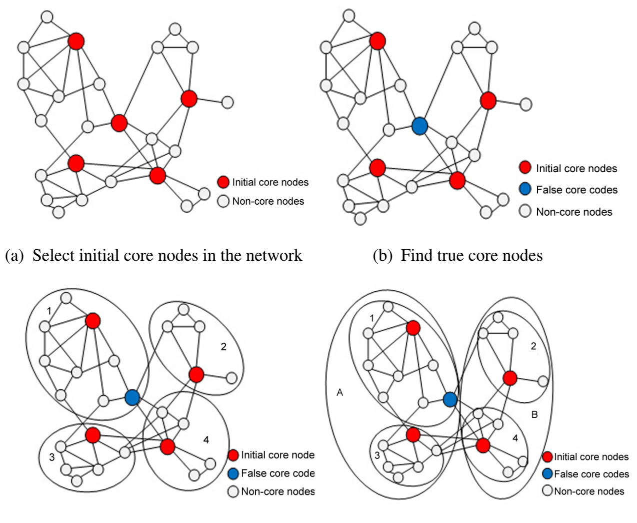
Step 3 For each non-core node, select the closest core node in the true-core node set to follow. As such, initial communities form around the core nodes.

Step 4 Repeat this process to evaluate the quality of the communities and merge close communities.

4.2 Selection of core nodes

Typically, a network consists of some communities, and each community has a core node. On this basis, we propose a reasonable assumption that as long as the core node of a community can reach the core nodes of the other communities, we can consider that all nodes of a community can reach all the nodes of the other communities, which means that the entire network is connected. This is similar to a capital city or big cities in each province being interconnected, so all the cities throughout the country are interconnected. In most social networks, the degree of distribution of the nodes follows a power law, whereby only a few are highly interconnected out of the total number of nodes in the network, which we call core nodes. The method used to select the initial core nodes is critical, and we must avoid selecting a false core node and missing a true core node.

Traditionally, we recognize the importance of a node in social networks as being based on



(a) Select initial core nodes in the network (b) Find true core nodes

(c) Construct initial communities (d) Merge close communities

Fig. 1 Steps of the PMAC algorithm.

the following metrics: degree centrality, closeness centrality, betweenness centrality, and eigenvector centrality. In this paper, we adopt the degree centrality, the most basic metric, to identify the core node of a network community. We do so for two reasons: (1) calculating the degree centrality is relatively simple, and (2) degree centrality and the metric used in the random walks transition matrix are consistent.

Note that degree centrality not only reflects the relevance of each node and other nodes, but is also associated with the network size, i.e., the number of nodes. As the size of the network increases, the maximum possible value of the degree centrality also increases. To eliminate the effects of the size of the network on the degree centrality, we use standardized measurement formulas for node importance, as proposed by Wasserman and Faust[19] in the t-step transition matrix, $\boldsymbol{P}^t$, and define the importance of node i as follows:

Definition 1

$$\text{Important}(v_i)_{p^k} = d_i' = \sum_{j}^{n} a_{ij}/n-1\ (i \neq j) \quad (2)$$

If a node i is $\text{Important}(v_i)_{p^1} \geqslant \tau$, then we called it an initial core node, and after one step, the initial core node set is as follows:

$$\text{CenterSet}_{p^1} = \{v_i \,|\, \text{Important}(v_i)_{p^1} \geqslant \tau\}.$$

The core node set after t-step random walks is as follows:

$$\text{CenterSet}_{p^t} = \{v_i \,|\, \text{Important}(v_i)_{p^t} \geqslant \tau\}.$$

τ is a preset threshold value.

According to the characteristics of random walks, as the steps tend to infinity, the element values of the transfer matrix constantly tend to their final stable value. In the iterative process, the probability of the transition of one node to another node increases or gradually stabilizes at a large value, which shows that the destination node is more central, and is thus a true core node. Conversely, if the importance of a node in the initial core node set gradually decreases during the iterative process, it is a false core node and will be discarded from the core node set in the next iteration step. Therefore, we use classical centrality measures, combined with random walks, to solve the challenge of selecting the initial core node set. If we set the threshold τ to be higher, the initial core node set is relatively small; if we set it to be lower, the initial core node set is relatively large. However, this change has little effect on the determination of the final core nodes. The more important issue is how to calculate the appropriate step number, t.

4.3 Computing the step number of random walks based on PMAC

In a random walks process, the method used to determine the number of steps is a key issue. In the community detection of a given network, a specific number of steps will correspond to a specific but not necessarily the most reasonable solution. Random walks with too few steps cause the network to become decentralized, with fewer edges between the nodes, which results in many smaller local clusters scattered throughout a network, and which is very different from the real structure. On the other hand, too many steps cause the community borders to become blurred and rough, and make it difficult to partition the community structure.

In addition, if the random walk has too many steps, it will increase the number of iterations in the transition probability matrix, which increases the calculation time complexity. Pon and Latapy[11] proposed that time complexity is close to n^3. If we imagine a network comprising hundreds of thousands of nodes, this will lead to a great loss of time, which limits the usefulness of the algorithm in a real network. We propose the followed definition for determining the appropriate number of steps, which avoids this problem.

Definition 2

$$\text{error}_t = \sum |P_{ij}^{(t)} - \phi_j|,\ v_i, v_j \in \text{Centerset}_{p^t}, i \neq j \quad (3)$$

In Definition 2, i and j represent the two core nodes' indices, respectively; ϕ_j is equal to the value of Eq. (1), and t is the number of walk steps. When t tends to infinity, error_t is approximately zero. But the decrease of the error will be very small and slow with the time. The optimal number of steps refers to the value that will result in an the error of smaller value. We note that in Definition 2, the error values for transition probabilities are calculated only for core nodes, but the transition probabilities computed in each iteration requires the participation of all nodes. For this reason, we called it the PMAC.

Definition 3

The optimal number of steps, $k = \min\{t \,|\, \text{error}_t < \varepsilon\}$, ε is a preset threshold value.

Support $\boldsymbol{P}$ is the one-step transition matrix. Let $C(\boldsymbol{P}) = \frac{1}{2}\sup_{i,k} \sum \left|P_{ij} - P_{kj}\right|$ represent the Dobrushin's contraction coefficient, in which v_i and v_j

belong to the same community and v_k and v_j belong to different communities. To verify the existing steps, we use the coefficient. According to the Dobrushin theorem[20], if $C(\boldsymbol{P}) < 1$, then there exist two constants α and t_0 such that $\text{error}_t < \alpha(C(\boldsymbol{P}))^k, \forall t > t_0$. Based on above equation, we have the following:

$$\alpha(C(\boldsymbol{P}))^t < \varepsilon \Rightarrow \text{error}_t < \varepsilon \Rightarrow t > \frac{\log\varepsilon - \log\alpha}{\log(C(\boldsymbol{P}))} \tag{4}$$

However, $C(\boldsymbol{P})$ is usually greater than 1. Let m be the minimum number satisfying $C(\boldsymbol{P}^m) < 1$, then we have $t > m \cdot \dfrac{\log\varepsilon - \log\alpha}{\log(C(\boldsymbol{P}^m))}$. Eventually, we have the following:

$$x = \min\{t \,|\, \text{error}_t < \varepsilon\} = \max\left\{t_0, \left\lceil m \cdot \frac{\log\varepsilon - \log\alpha}{\log(C(\boldsymbol{P}^m))} \right\rceil\right\} \tag{5}$$

According to the final equation, we have $\log(\varepsilon) = a \cdot t + b$, where $a = \log(C(\boldsymbol{P}^m))/m$ and $b = \log\alpha$. Therefore, for a given network, the steps certainly exist[13].

4.4 Choosing the communities to be merged

Once we have determined the optimal steps, t, we can guarantee the approximation stability of the transition matrix and obtain the final core node set of the network. With the t-step transition matrix, we determine which core node is nearest to each non-core node, and then the non-core nodes are put into the community of their nearest core node. The process is as follows: for each non-core node i, we compute the optimal community k with $k = \arg\max_j(P_{ij}^{(t)})$, where node j is the core node after t-step iterations.

The initial community formed around the core nodes in Algorithm 1 requires further agglomeration to form a hierarchical structure, which involves two steps: (1) Select two communities to be merged based on their similarity; (2) Decide whether the agglomeration process should stop. Traditional quality metrics, such as module Q, involves more computational complexity. From this perspective, we propose the community closeness metric to determine the similarity between two communities, which is defined as follows:

Definition 4

$$\text{Closeness}(C_i, C_j) = \frac{\dfrac{\text{Edges}_{\text{internal}}(C_i \cup C_j)}{\text{Edges}_{\text{external}}(C_i \cup C_j)}}{\dfrac{1}{2} \times \left(\dfrac{\text{Edges}_{\text{internal}}(C_i)}{\text{Edges}_{\text{external}}(C_i)} + \dfrac{\text{Edges}_{\text{internal}}(C_j)}{\text{Edges}_{\text{external}}(C_j)}\right)} \tag{6}$$

$\text{Edges}_{\text{internal}}(C_i \cup C_j)$ and $\text{Edges}_{\text{external}}(C_i \cup C_j)$ are the internal and external edges in the new merged community, respectively; $\text{Edges}_{\text{internal}}(C_i)$ and $\text{Edges}_{\text{external}}(C_i)$ are the internal and external edges of community C_i, respectively, and $\text{Edges}_{\text{internal}}(C_j)$ and $\text{Edges}_{\text{external}}(C_j)$ are the internal and external edges of community C_j, respectively. When the closeness value is greater than the threshold φ, we merge the two communities. Typically, the threshold φ is set between 1 and 2, and can be modified for different networks.

Algorithm 1 Repeat to merge two communities to construct hierarchical communities

```
Input: Communities $C$, parameter $\varphi$
Output: Hierarchical communities
1: for any two communities $C_i, C_j$ in $C$ do
2:     Compute Closeness $(C_i, C_j)$ if it never be computed;
3: end for
4: if (Closeness$(C_i, C_j)$) $> \varphi$ then
5:     Add community $C_i \cup C_j$ into $C$;
6:     Delete communities $C_i$ and $C_j$ from $C$;
7: else
8:     Go to 11;
9: end if
10: Repeating Step 1 to Step 9;
11: return a dendrogram of communities
```

The details of PMAC process are presented in Algorithm 1. The time complexity associated with calculating the initial core-node set is $O(n)$; the time complexity associated with calculating the probability transition matrix with the optimal step is $O(n^2)$. The time complexity associated with constructing a dendrogram of communities is multiplied by the number of agglomerations and the time complexity of one agglomeration. Let $k = h(V)$ be the number of communities, which is the number of core nodes after the t-step random walk, then the height of the community dendrogram is $\log k$, and the number of agglomeration is $k \log k$. As we know, the time complexity of one agglomeration is $O(n)$. So, the time complexity required to build a community tree is $O(nk \log k)$, and thus the total time complexity of the algorithm is $O(n^2 + nk \log k)$.

4.5 Validation metrics

To evaluate the effectiveness of different algorithms, we adopted two metrics: accuracy and Newman-Girvan modularity.

(1) Accuracy

To determine the accuracy, each node in a community must be labeled. For each node i, we suppose $l_{\text{t}}(i)$ to be its true community label and let $l_{\text{p}}(i)$ be its label predicted label by the algorithm. The accuracy is defined as the fraction of all nodes whose predicted label is equal to true label[21]:

$$\text{Accuracy} = \frac{|\{v_i | l_{\mathrm{t}}(i) = l_{\mathrm{p}}(i)\}|}{n} \tag{7}$$

(2) Newman-Girvan modularity

We proposed the Newman-Girvan modularity to evaluate the structural strength of the network community. For the network, let cp be a community partition predicted by an algorithm. The Newman-Girvan modularity of cp is defined as[22]

$$Q_{\mathrm{cp}} = \sum_{vw} \left[a_{vw} - \frac{d_v \times d_w}{(2m)(2m)} \right] \delta(C_v, C_w) = \sum_{i=1}^{c} (e_{ii} - a_i^2) \tag{8}$$

where a_{vw} is 1 between node v and node w in our paper, yet, it represents the weights in directed networks. $\delta(C_v, C_w) = 1$, when v and w belong to the same community, else its value is 0. C_v and C_w represent the communities to which node v and node w belong, respectively. e_{ij} is the fraction of edges with one end node in community i and the other in community j: $e_{ij} = \sum_{vw} \frac{A_{vw}}{2m}, v \in C_i, w \in C_j$. a_i is the fraction of edges attached to nodes in community i: $a_i = \frac{d_i}{2m} = \sum_j e_{ij}$.

5 Experimental Evaluation

To verify the performance of PMAC in different types of networks, we used three datasets for analysis: Zachary's Karate Club[16] network, the American College Football[12] network, and partial data from the Facebook[23] network. As shown in Table 1, in Zachary's Karate club network, there are a total of 34 nodes and 78 edges, which indicates a highly organized structure; the American College Football network is a moderately organized network with 115 nodes and 613 edges; and the Facebook network represents a large social network dataset that consists of 2888 nodes and 2981 edges. We compared the experimental results of our proposed PMAC with those of the GN[24] and Newman fast[25] algorithms.

Table 1 Test datasets.

Network	Number of nodes	Number of edges
Zachary's Karate Club	34	78
American College Football	115	613
Facebook	2888	2981

We conducted the experiments using an Intel Pentium dual-core P6000 processor, with 2 GB DDR3 of memory, a Windows 7 operating system, and Matlab R2012a data analysis tools.

5.1 Existence of limited steps

In the first experiment, we determined the steps of the random walks. In Figs. 2 and 3, the curves describe the step-error changes in the Zachary's Karate Club and Facebook networks, which represent small-scale and large-scale networks, respectively. From Figs. 2a and 3a, we can see that, with increasing steps, the probability of walks between nodes tends to be steady, that is, the value of Eq. (3) gradually decreases. However, the sharpness of the convergence and the time are related to the specific network. As shown in Fig. 2a, in Zachary's Karate Club, with increasing steps, the transition probability quickly converges. In Fig. 3a, for the Facebook network, however, this process is not clear and due to fluctuation, we cannot determine the optimal number of steps.

To solve this problem, we use the change in the error value caused by the adjacent steps, $\Delta\text{error} = |\text{error}_t - \text{error}_{t-1}|$, to determine the approximate optimal step, so that when Δerror obtains the relative minimum, the approximate optimal step is reached. In Figs. 2b and 3b, as the number of steps increases, the change in the error value shows a decreasing trend and then tends to a stabilized state. Thus, for Zachary's Karate Club, the

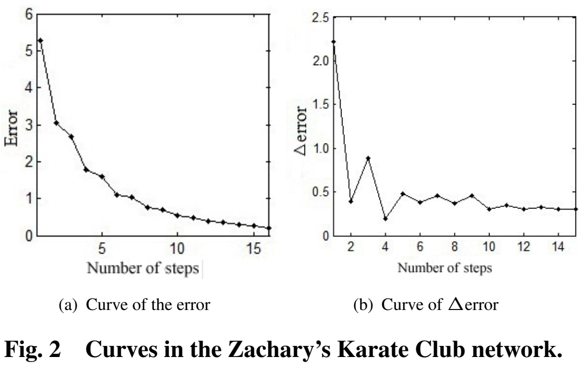


Fig. 2 Curves in the Zachary's Karate Club network.

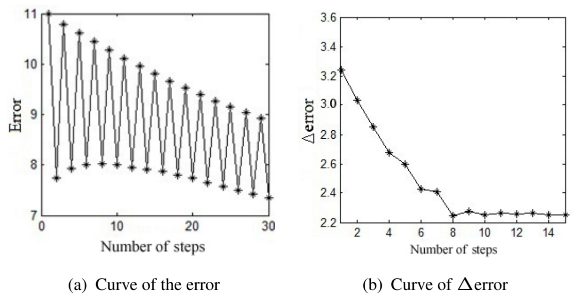


Fig. 3 Curves in the Facebook network.

approximate optimal number of steps is four, and for the Facebook network, the optimal number is eight.

5.2 Initial core nodes selection

As mentioned earlier, to select the initial core nodes, we use the degree centrality threshold (i.e., equal to or greater than the average degree of the network). Using the proposed algorithm, we determined the initial core node set to be {1, 2, 3, 4, 9, 14, 24, 32, 33, 34} in the small-scale Zachary's Karate Club network. We also used the second method to select the initial core nodes, to satisfy both the degree centrality and tightness centrality (i.e., equal to or greater than the shortest-path of the network), which then yielded another core node set, {1, 2, 4, 24}. However, after multi-step iterations, the true core nodes remained nodes 1 and 9. Although node 9 was not selected as an initial core node in the second method, after iterative calculations, it eventually became a real core node. We also used different degree centrality thresholds to select different initial core nodes, and the real core nodes were still nodes 1 and 9. In Table 2, we show the result for the above two standard methods for determining initial core nodes.

For the large-scale network Facebook, we repeated the test. In Table 3, we list the effects of different thresholds on the initial core node set. Although the initial core nodes are not same, the final real core nodes remained {1, 289, 716, 1524, 2688}. From the experimental results of different datasets, we found the PMAC results to not be affected by whether we had selected some false initial core nodes or had missed some true code nodes. As such, we can conclude that the PAMC is robust in selecting the criteria and threshold values of the initial nodes.

Table 2 Effect of the threshold value on the core nodes (Zachary's Karate Club network).

Threshold value selection	No. of initial core nodes	No. of final core nodes
Degree centrality = 4.021	1, 2, 3, 4, 6, 7, 8, 9, 14, 28 ,30, 31, 32, 33, 34	1, 9
Degree centrality = 5.010	1, 2, 3, 4, 9, 14, 24, 32, 33, 34	1, 9
Degree centrality = 4.588 and tightness centrality = 2.408	1, 2, 4, 24	1, 9
Degree centrality = 5.010 and tightness centrality = 2.010	1, 2, 4, 9, 14, 24, 32, 33, 34	1, 9

Table 3 Effect of threshold value on core nodes (Facebook network).

Threshold value selection	Number of initial core nodes	No. of final core nodes
Degree centrality = 2.064	16	1, 289, 716, 1524, 2688
Degree centrality = 3.010	16	1, 289, 716, 1524, 2688
Degree centrality = 4.010	10	1, 289, 716, 1524, 2688
Degree centrality = 5.010	10	1, 289, 716, 1524, 2688
Degree centrality = 6.010	10	1, 289, 716, 1524, 2688

5.3 Accuracy

We designed the third experiment to assess the algorithm's community detection capability, using 1, 2, 3, ... as the labeled nodes in Zachary's Karate Club and the American College Football networks, respectively.

As we know, in the small-scale Zachary's Karate Club network, the real results include two communities, one is {1, 2, 3, 4, 5 , 6, 7, 8, 11, 12, 13, 14, 17, 18, 20, 22} and the other is {9, 10, 15, 16, 19, 21, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34}. Comparing the results, the number of errors in the real divided community results and those of the PMAC algorithm is 0, whereas in the GN algorithm, node 3 is the wrong division, and in the Newman fast algorithm, node 10 was mistakenly classified, and all the mistakenly classified nodes are on the community margins.

Figure 4 plots the Receiver Operating Characteristic (ROC) curves of the PMAC, GN, and Newman fast algorithms for the Zachary's Karate Club network. Because Zachary's Karate Club network is a small-scale dataset, the PMAC result is similar to the community division of the real network. Since the results may be too idealistic in a small dataset, we wanted to know how

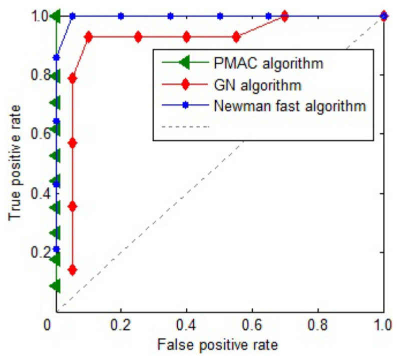


Fig. 4 ROC curves of Zachary's Karate Club network.

the algorithm would perform in a larger dataset.

Figure 5 depicts the community divisions of the real network in American College Football network. The different colors represent different communities, and nodes with the same color belong to the same community. We can see that there are 12 communities in the network, respectively, {4, 6, 11, 12, 41, 53, 73, 75, 82, 85, 99, 103, 108}, {1, 5, 10, 17, 24, 42, 91, 94, 105}, {2, 26, 34, 38, 46, 90, 104, 106, 110}, {13, 15, 19, 27, 32, 35, 37, 39, 43, 44, 55, 62, 72, 86, 100}, {47, 50, 54, 68, 74, 84, 89, 111, 115} , {7, 14, 16, 48}, {45, 49, 58, 67, 76, 87, 92, 93, 98, 113}, {18, 21, 28, 57, 59, 60, 63, 64, 66, 71, 77, 88, 9697114}, {25, 29, 70}, {8, 9, 22, 23, 51, 52, 69, 78, 79, 109, 112}, {20, 30, 31, 36, 56, 80, 81, 83, 95, 102}, and {3, 33, 40, 61, 65, 101, 107}.

For the medium-scale American College Football network, Fig. 6 shows the community division results of the GN algorithm and the Newman fast algorithm, and the number of PMAC algorithm errors with respect to the community divisions is still 0. Moreover, the node classification is 100 percent correct. With the GN algorithm, the number of communities is 12, which is correct, but nodes {83, 29, 91, 83, 43, 81, 68, 74, 84, 89, 111, 59} are divided wrong, and with the Newman fast algorithm, the number of communities is also correct at 12, and nodes {81, 83, 29, 91, 64, 59, 60, 98, 99, 75, 73, 53, 11, 103, 57, 108, 97, 111} are mistakenly classified. Since different networks have different community formation processes, the PMAC algorithm tends to be more reasonable for verge of nodes.

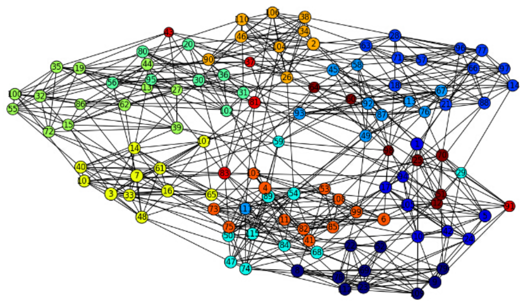


Fig. 5 PMAC community division (American College Football network).

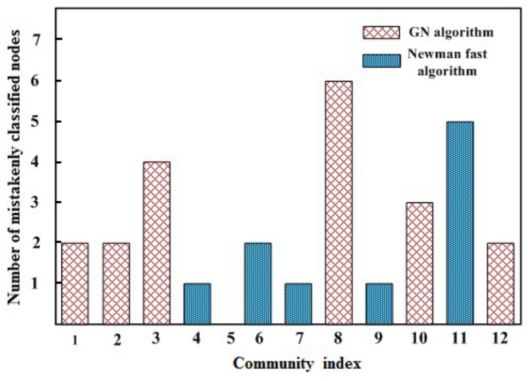


Fig. 6 Community division in American College Football network.

For the large-scale Facebook network, Table 4 shows the community division results for the three algorithms, as well as the modularity. The GN algorithm fails to successfully obtain the final number of communities successfully, whereas the number of communities was determined by the Newman fast algorithm to be five and by the PMAC algorithm to be eight. Since we cannot know how many communities is most reasonable, based on just one real network for which the index does is not suitable, determining the modularity of the divided community is a good option. The modularity result for the PMAC algorithm is 0.5671 and for the Newman fast algorithm is 0.2586. As such, the former has the better result based on its higher modularity value.

We also compared the modularity of the three algorithms in all three databases, as shown in Table 4. PMAC outperformed the two other algorithms. The PMAC and Newman fast algorithms yielded the similar approximation results in the small-scale databases, but differed in the large-scale database as mentioned above. Of the three algorithm, the GN algorithm is particularly unsuitable for processing large-scale networks.

We also compared the modularity of the three algorithms in all three databases, as shown in Table 5. PMAC outperformed the two other algorithms. The PMAC and Newman fast algorithms yielded the similar

Table 4 Community division results and modularity of the divided communities (Facebook network).

Algorithm	Number of communities	Modularity of the algorithm
PMAC	5	0.5671
GN	—	—
Newman fast	8	0.2586

Table 5 Comparison of modularity.

Algorithm	Modularity of Karate network	Modularity of Football network	Modularity of Facebook network
PMAC	0.3947	0.7729	0.5671
GN	0.2738	0.3181	—
Newman fast	0.3855	0.6416	0.2586

approximation results in the small-scale databases, but differed in the large-scale database, as mentioned above. Of the three algorithms, the GN is a basic algorithm whose performance is not very good, and is particularly unsuitable for processing large-scale networks.

Because the PMAC algorithm uses a limited number of steps, it has high efficiency. For the above three datasets, we compare the time complexity of PMAC, GN, and Newman fast algorithms in Table 6. We can see that the PMAC algorithm exhibits better execution time than the Newman fast algorithm for the large-scale network, and the GN algorithm fails to cluster the final community structures.

As we can see from Table 6, even though in the small-scale networks the PMAC algorithm has a little higher complexity than the Newman fast algorithms, it has obvious advantages in large-scale networks. Generally speaking, our algorithm trades off high accuracy for better efficiency.

6 Conclusion

In this paper, we proposed a hierarchical community detection method based on a limited number of random walk steps, using the local approximate convergence of core nodes rather than global smooth convergence in the probability transfer matrix. To the best of our knowledge, this approach is not mentioned in the existing literature. PMAC calculates all the initial communities based on the t-step probability transfer matrix. The initial community partitions reflect the true core nodes to represent the global structure information. As initial communities, we only consoli-dated ordinary nodes surrounding the core nodes, directly using local information to quickly form small communities. We then constructed a dendrogram of communities in the later phase, without recalculating the nodes to which a community belongs, which greatly improves the efficiency of the algorithm. The optimal step of our algorithm is nearly the best result. In the future work, we will focus on combining the specific characteristics of the target network to design a better stopping condition for random walks and determine how to take advantage of the information to merge tree communities.

Table 6 Comparisons of complexity. (s)

Algorithm	Time consuming of Karate network	Time consuming of Football network	Time consuming of Facebook network
PMAC	0.19	3.42	335.74
GN	4.34	505.73	>10 000
Newman fast	0.09	0.12	516.29

Acknowledgment

The authors would like to express their thanks to the anonymous reviewers for their constructive comments and suggestions. This work was supported by the National Natural Science Foundation of China (Nos. 61272422, 61572260, 61373017, and 61572261).

References

[1] Y. Y. Ahn, J. P. Bagrow, and S. Lehmann, Link communities reveal multiscale complexity in networks, *Nature*, vol. 466, no. 7307, pp. 761–764, 2010.

[2] G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee, Self-organization and identification of Web communities, *Computer*, vol. 35, no. 3, pp. 66–70, 2002.

[3] V. Spirin and L. A. Mirny, Protein complexes and functional modules in molecular networks, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 100, no. 21, pp. 12123–12128, 2003.

[4] M. E. J. Newman, Detecting community structure in networks, *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 38, no. 2, pp. 321–330, 2004.

[5] S. Fortunato, Community detection in graphs, *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.

[6] M. E. J. Newman, Modularity and community structure in networks, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 23, pp. 8577–8582, 2006.

[7] J. Leskovec, K. J. Lang, and M. Mahoney, Empirical comparison of algorithms for network community detection, in *Proc. 19th Int. Conf. on World Wide Web*, Raleigh, USA, 2010, pp. 631–640.

[8] N. Aston and W. Hu, Community detection in dynamic social networks, *Communications and Network*, vol. 6, no. 2, p. 124, 2014.

[9] S. M. Van Dongen, Graph clustering by flow simulation, Ph.D. dissertation, University of Utrecht, Utrecht, the Netherlands, 2001.

[10] H. Zhou and R. Lipowsky, Network brownian motion: A new method to measure vertex-vertex proximity and to identify communities and subcommunities, in *Computational Science-ICCS*, 2004, pp. 1062–1069.

[11] P. Pons and M. Latapy, Computing communities in large networks using random walks, in *Proc. 20th Int. Conf. on Computer and Information Sciences*, Istanbul, Turkey, 2005, pp. 284–293.

[12] M. Rosvall and C. T. Bergstrom, Maps of random walks on complex networks reveal community structure, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, no. 4, pp. 1118–1123, 2008.

[13] M. Alamgir and U. Von Luxburg, Multi-agent random walks for local clustering on graphs, in *Proc. 10th Int. Conf. on IEEE Data Mining*, Sydney, Australia, 2010, pp. 18–27.

[14] J. C. Delvenne, S. N. Yaliraki, and M. Barahona, Stability of graph communities across time scales, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, no. 29, pp. 12755–12760, 2010.

[15] L. Backstrom and J. Leskovec, Supervised random walks: Predicting and recommending links in social networks, in *Proc. 4th Int. Conf. on Web Search and Web Data Mining*, Hong Kong, China, 2011, pp. 635–644.

[16] B. Yang, J. Di, J. Liu, and D. Liu, Hierarchical community detection with applications to real-world network analysis, *Data and Knowledge Engineering*, vol. 83, pp. 20–38, 2013.

[17] X. Fu, C. Wang, Z. Wang, and Z. Ming, Threshold random walkers for community structure detection in complex networks, *Journal of Software*, vol. 8, no. 2, pp. 286–295, 2013.

[18] H. Park and K. Lee, Dependence clustering, a method revealing community structure with group dependence, *Knowledge-Based Systems*, vol. 60, pp. 58–72, 2014.

[19] S. Wasserman and K. Faust, *Social Network Analysis: Methods and Applications*. Cambridge University Press, 1994.

[20] R. L. Dobrushin, Central limit theorem for non-stationary Markov chains, I and II, *Theory of Probability and Its Applications*, vol. 1, no. 1, pp. 65–80, 1956.

[21] K. Steinhaeuser and N. V. Chawla, Identifying and evaluating community structure in complex networks, *Pattern Recognition Letters*, vol. 31, no. 5, pp. 413–421, 2010.

[22] M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, *Physical Review E*, vol. 69, no. 2, p. 026113, 2004.

[23] Institute of Web Science and Technologies at the University of KoblemeClandau, KONECT, http://konect.uni-koblenz.de/networks/.

[24] M. Girvan and M. E. J. Newman, Community structure in social and biological networks, *Proceedings of the National Academy of Sciences of Sciences of the United States of America*, vol. 99, no. 12, pp. 7821–7826, 2002.

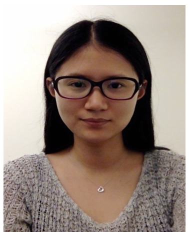
[25] M. E. J. Newman, Fast algorithm for detecting community structure in networks, *Physical Review E*, vol. 69, no. 6, p. 066133, 2004.

Wei Zhang is currently a professor with the School of Computer Science at the Nanjing University of Posts and Telecommunications. He received the PhD degree from Soochow University in 2008. In 2016, he was a visiting scholar at Purdue University, under supervision from Dr. Ninghui Li. He has served as a reviewer for many journal papers. His current research interests include social networks analysis, differential privacy, malicious code detection, etc.

Feng Kong received the master and bachelor degrees from Nanjing University of Posts and Telecommunications in 2017 and 2014, respectively. Her research interests include social network analysis and network community detection.

Liming Yang received the master and bachelor degrees from Nanjing University of Posts and Telecommunications in 2015 and 2012, respectively. His research interests include social network analysis and network link prediction.

Yunfang Chen is currently an associate professor with the School of Computer Science at the Nanjing University of Posts and Telecommunications. He received the PhD degree from Soochow University in 2008. He has served as a reviewer for many journal papers. He has published more than 50 papers in refereed journals and conferences. His main research interests include software defined networks, network security, social networks, etc.



Mengyuan Zhang received the master degree from Nanjing University of Posts and Telecommunications in 2012. She is currently pursuing the PhD degree at the Concordia Institute for Information Systems Engineering, Concordia University, Canada. Her research interests include network security, security metrics, and attack surface.