Journal of Computers Vol. 28, No. 5, 2017, pp. 70-81
doi:10.3966/199115992017102805007

A Novel Community Detection Algorithm Based on E_FEC

Lidong Wang[1*], Yun Zhang[2], Yin Zhang[3] and Huixi Zhang[1]

[1] Qianjiang College, Hangzhou Normal University,
Hangzhou, Zhejiang, 310018 P.R. China
Violet_wld@163.com, zhhuixi@126.com

[2] Zhejiang University of Media and Communications,
Hangzhou, Zhejiang, 310018 P.R.China
zhangyun_zju@zju.edu.cn

[3] College of Computer Science and Technology, Zhejiang University,
Hangzhou, Zhejiang, 310027 P.R.China
yinzh@zju.edu.cn

Received 22 June 2016; Revised 7 February 2017; Accepted 8 February 2017

Abstract. FEC adopts an agent-based heuristic that makes the algorithm efficient and is presented with two phases that are Finding Community (FC) and Extraction Community (EC). Although designed with linear running time, original FEC can not obtain ideal results on the graph whose community structure is not well defined. This paper extend FEC as E_FEC to seek a good trade-off between effectiveness and efficiency. In FC phase, we calculate the accumulative transition probability to find the existence of communities, and propose an automatic selection algorithm for the sink node. In EC phase, we present another simpler cut criterion based on Average cut (*Acut*) which costs less running-time in EC phase. The performance of E_FEC is rigorously validated through comparisons with other representative methods against both synthetic and real-world networks with different scales.

Keywords: community detection, E_FEC algorithm, FEC algorithm, random walk

1 Introduction

1.1 Research Background

Community detection is the task of clustering the vertices of the network into groups taking into consideration the structure of the graph. As the result of community detection, there should be many edges within each group and relatively few between the groups. Therefore, community detection is an effective method that measures vertex closeness based on structural similarity (e.g., the number of common neighbors between two vertices), which can help people to understand the structure of complex networks, discover the latent information and predict user behaviors in social networks [1]. Andrea and Santo carried out a comparative analysis of the performance of various community detection algorithms, and concluded that the random walk methods performed rather better compared with others [2]. Unfortunately, most of the methods based on random walk are computationally expensive, and the performance on the network whose community structure is not well defined are unsatisfactory. The objective of seeking a good trade-off between effectiveness and efficiency remains challenging.

1.2 Related Works

In the literature, community detection algorithms can generally be divided into the following categories:

[*] Corresponding Author

The methods based on modularity optimization. Mei, He, Shi, Wang and Li proposed a contraction-dilation algorithm for modularity optimization [3]. Newman adopted greedy strategy to search the maximal Q [4]. The time complexity of this algorithm is $O(mn)$, where m is the number of edges, and n is the number of nodes. Jin et al. proposed a fast algorithm based on the optimization of the local index to replace the optimization of Q, which obtained a better trade-off between efficiency and clustering quality [5]. Fortunato and Barthelemy showed that modularity-based methods failed to identify modules smaller than a scale [6], so it was not a robust method for group detection of some social networks that may not contain large amount of nodes. In 2016, Newman demonstrated an equivalence between the method of modularity maximization and the method of maximum likelihood applied to the degree-corrected stochastic block model [7]. This equivalence provides a mathematically principled derivation of the modularity function.

The methods based on division. The idea of this method is removing the edges in the network according to some certain rules, such as the number of shortest paths between pairs of vertices that run along it. The most popular algorithm is GN (Girvan-Newman) algorithm proposed by Girvan and Newman [8].The disadvantage of this method is the high running time. Some researchers have attempted to solve this problem at the expense of low clustering accuracy [9].

The methods based on label propagation. The idea of this method is that each node in a network with community structure should be located in a same community with most of its neighbors. Ugander and Backstrom proposed a balanced label propagation by combining the computational efficiency of label propagation with the guarantees of constrained optimization [10]. Lin, Zheng, Xin and Chen proposed a novel label propagation-based model with community kernel, which achieved better performance than related algorithms [11]. This kind of methods can obtain fast running speed, especially for the community detection problem in large scale networks. However, clustering performance should be further improved since it causes unstable quality in the detection results. To improve the accuracy, Li, Huang, Wang and Chen divided networks using a stepping framework and propagated labels based on the similarity among nodes or subnetworks [21].

The methods based on random walk. Recently, lots of random walk based methods were proposed in networks of different structures [12]. Dongen [13] described a random walk based algorithm named Markov Cluster Algorithm(MCL), which simulates a peculiar process of flow diffusion in a graph. As of now, the MCL is one of the most used community detection algorithms. However, the algorithm should scale as $O(n^3)$ (n is the node number in the graph). Pons and Latapy [14] used random walk to define a distance measure between vertices. The algorithm runs to completion in a time $O(n^2 d)$ on a sparse graph, where d is the depth of the dendrogram. Hu, Li, Zhang and Fan [15] designed a graph clustering technique based on signaling process with random walk scheme between vertices. The complexity of the algorithm is $O[(\langle k\rangle +1)n^2]$, where $\langle k\rangle$ is the average degree of the graph. Wang, Liu, Liu and Pan proposed a novel overlapping community detection algorithm based on local random walk and multidimensional scaling [16]. The time complexity of the algorithm is $O(n^2)$. Xin, Xie and Yang proposed ANRW (Adaptive Non-Homogeneous Random Walk) to resolve the issue of instability caused by fixing the random walking step, but the method is adequate to the parallel computing and large data analysis [19].

From the above researches, we can see that random walk has been successfully used in community detection. Yang, Cheung and Liu presented a random walk method named FEC that is demonstrated to be fast and accurate to identify groups for both positive and signed network [17]. FEC is presented with two phases, Finding Community (FC) and Extraction Community (EC). The former transforms the adjacency matrix to compute their transition probability vector and sorts them for each row. The latter applies a cutoff criterion to the transformed adjacency matrix and divides it into two block matrices, which correspond to two subgraphs. One of these subgraphs is the identified community, and another is the matrix to be processed, recursively. However, original FEC algorithm makes the detection result sensitive to the selection of sink node, and the experimental results on networks do not define good community structure remains unsatisfactory.

In this paper, to overcome the shortcomings of the original FEC algorithm, we extend it as E_FEC with several improvements. E_FEC can achieve a good trade-off between effectiveness and efficiency. The contributions of E_FEC mainly focus on four aspects:

(1) In FC phase, we propose to calculate accumulative transition probability, which is a more reasonable way to find the existence of communities.
(2) In FC phase, we present a mathematical conclusion for determining the value of random walk steps.
(3) Original FEC method does not provide the sink node sensitivity analysis, which will result in unreasonable clustering results. So, an automatic sink node selection algorithm is proposed to solve this problem.
(4) Original EC phase has the time complexity of $O(m+n)$. We present an apparently simpler cut criterion based on Average cut [17] which costs less running-time.

2 E_FEC Algorithm

2.1 Improved FC Phase

The FC phase adopts an agent-based approach to model the problem of finding the group that contains a specific node for graph. An imaginary random walker walks freely from one node to another, following the links of a given graph. The walker's route can be viewed as a stochastic process defined based on the links' attributes. In particular, when the walker arrives at a node, it will select one of its neighbors at random and then go there. Let $X=\{X_l, l \geq 0\}$ denotes a random walk series. Let $P\{X_l = N_l, 1 \leq N_l \leq n\}$ be the probability that the walker will arrive node N_l after going exactly l steps. X is a discrete Markov chain if we have:

$$P\{X_l = N_l \mid X_0 = N_0, X_1 = N_1, ..., X_{l-1} = N_{l-1}\} = P\{X_l = N_l \mid X_{l-1} = N_{l-1}\}. \quad \textbf{(1)}$$

Let $P_{i \to j}$ be the probability of the agent walking from node i to its neighbor node j. In a weighted social network, this probability can be computed as follow:

$$P_{i \to j} = \frac{W_{ij}}{\sum_j W_{ij}}, \quad \textbf{(2)}$$

where W_{ij} represents the weight of link <*i,j*>, $\sum_j W_{ij}$ is the weighted degree of node i. According to the homogeneous Markov chain, we have:

$$P\{X_l = j \mid X_{l-1} = i\} = P_{i \to j}. \quad \textbf{(3)}$$

Then, let $P_t^l(i)$ be the probability that agent starting from node i can eventually arrive at a specific sink node *t* after exactly *l* steps. The value of $P_t^l(i)$ can be estimated iteratively by

$$P_t^l(i) = \sum_{<i,j>} P_{i \to j} \cdot P_t^{l-1}(j). \quad \textbf{(4)}$$

If $i = t$, then $P_t^l(i) = 1$. The main idea of FC phase is that the random walker starting from nodes within the community of the sink node should reach the sink node more easily within l steps since more paths can be chosen. Therefore, the random walk will hit the sink at a high probability if it is within the sink community. Otherwise, the probability will be very low. However, $P_t^l(i)$ can not indicate whether there exist more paths between two nodes when the length of paths are smaller than l. To solve this problem, we calculate $T_{i,t}^l$ to denote the accumulative transition probability between the sink node and node i, which is defined as:

$$T_{i,t}^l = \sum_{l'=1}^{l} P_t^{l'}(i). \quad \textbf{(5)}$$

Based on the above analysis, given a sink node t and its corresponding community C_t, we have:

$$T_{i,t}^l > T_{k,t}^l, \text{ for } i \in C_t, k \notin C_t. \quad \textbf{(6)}$$

Correspondingly, the steps for improved FC phase can be designed as follows:

Step 1 Specify a node t as the sink node;

Step 2 Calculate $T_{i,t}^{l}$ for each node i in terms of the sink node t;

Step 3 Rank all nodes according to their associated value $T_{i,t}^{l}$.

Based on the above steps, we should extract the community of the sink node by dividing the sequences of nodes during the EC phase to be introduced in Section 2.4. The algorithm for calculating accumulative transition matrix $\boldsymbol{T}$ is given as follows:

Algorithm1 Computing accumulative transition matrix $\boldsymbol{T}$

```
Input:  W, the adjacency matrix of a network;
        $t$, sink node;
        $l$, number of steps;
Output: $T_{i,t}^{l}$, accumulative transition probability;
for $i$=1:$n$
    $P_t^0(i) = 0 \cdot I_{i \neq t} + 1 \cdot I_{i=t}$;
end;
for $l'$=1:$l$
    for $i$=1:$n$
        $P_t^{l'}(i) = \sum_{<i,j>} \frac{W_{ij}}{\sum_j W_{ij}} \cdot P_t^{l'-1}(j)$;
    end;
end;
calculate $T_{i,t}^{l}$ for each node $i$ according to Eq. (5);
return $T_{i,t}^{l}$;
```

2.2 Determining the Sink Node

The network constituted by community C$_1$ and C$_2$ is shown in Fig. 1, where node 5 and node 6 are the border nodes of C$_1$ and C$_{2.}$ Yang et al. did not provide the theory demonstration for the selection of sink node, which was randomly chosen in the literature [17]. Fig. 1 gives an example to show that whether the selection of sink node will influence $P_t^l(i)$. As shown in Fig. 1, the simple network contains two communities. When we set node 5 as the sink node, node 6 would have higher probability of eventually arriving at the sink within l ($l=3$) steps than node 1 (mathematically speaking, $T_{6,5}^{l} > T_{1,5}^{l}$). This means that node 6 has higher probability to be clustered in the group that node 5 belongs to. However, when node 3 is set as the sink node, node 1 would have higher probability of eventually arriving node 3 (mathematically speaking, $T_{6,3}^{l} < T_{1,3}^{l}$), which conforms to the real community structure.

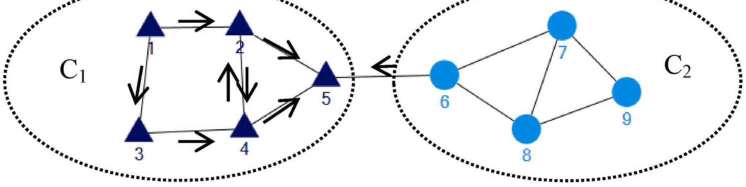


Fig.1. A simple network

Table 1 shows the detecting results on two social networks by selecting different nodes as sink node. The definition of *Ncut* can be seen in Section 3.2. This result indicates that different sink node would influence the effectiveness of community detection. Selecting nodes with lower degree obtains better performance than selecting nodes with higher degree. It is obvious that the node with minimum degree has low probability to be border nodes of one community. If we select the node located at the boundary between communities as sink node, the agent that starts from nodes outside the community will have a much higher probability of eventually arriving at the sink. Thus, it would be better to select sink node by avoiding these nodes. Radicchi [18] concluded that the edges connecting nodes in different communities are included in few or no triangles. On the other hand, many triangles exist within clusters. Based on this, we propose a method of automatic sink node selection, which is listed as follows:

Step 1 Compute the degree of all nodes d_i in network *G*.

Step 2 Let $r = \arg\max_i d_i$, $k = \arg\min_i d_i$. If d_r =2, then set the node *r* as the sink node; If d_k =1, then set node *k* as the sink node; else go to Step 3.

Step 3 For each pair of nodes (m,n), compute the number of triangles or rectangles $N_{(m,n)}$ containing m and n.

Step 4 Let $(u,v) = \arg\max_{(m,n)} N_{(m,n)}$, then randomly select the destination node *t* from $\{u,v\}$.

In Step 2, it is easy to prove that the network *G* is presented as lines or circles when d_r =2. Thus, it is reasonable to set node *r* as the destination node. When the degree of node k is 1, it will certainly not connect the nodes in other communities.

Table 1. The results of community detection by selecting different sink nodes

Sink node	Dolphin (*Ncut*)	Football (*Ncut*)
Sink = 1	1.4593	5.7039
Maximum degree	1.9684	4.8796
Minimum degree	1.4059	4.0781

2.3 Determining the Number of Random Walk Steps

Given a sink node t , how can we determine a reasonable l such that we have $T_{i,t}^l > T_{k,t}^l$, for each $i \in C_t$ and each $k \notin C_t$. In our paper, we estimate the value of l by a preset error that can explicitly measure the distance between transition probabilities and their limitations. Mathematically speaking, given a preset error threshold ε , to obtain a reasonable l is to satisfy:

$$error_{(l)} < \varepsilon \,. \tag{7}$$

We define:

$$\begin{aligned} error_{(l)} &= \sum_i \frac{d_i}{d_t} \left| P_t^l(i) - \lim_{l\to\infty} P_t^l(i) \right| \\ &= \sum_i \left| \frac{d_i}{d_t} P_t^l(i) - \frac{d_i}{d_t} \lim_{l\to\infty} P_t^l(i) \right| . \\ &= \sum_i \left| P_i^l(t) - \lim_{l\to\infty} P_i^l(t) \right| \end{aligned} \tag{8}$$

As indicated, $X = \{X_r, r \geq 0\}$ denotes a random walk series, and $\varphi_t = \lim_{l\to\infty} P_t^l(i)$. Because X is ergodic, according to the limit theory of Markov chain, *X* will have a unique stationary distribution $\Psi = (\varphi_1, \varphi_2, ... \varphi_n)$ (also known as limit distribution), which satisfies $\Psi = \Psi P$, where P is one-step transition probability matrix. It is easy to verify that:

$$\varphi_t = \lim_{l\to\infty} P_t^l(i) = \frac{d_t}{\sum_k d_k} \,. \tag{9}$$

Thus, we have:

$$error_{(l)} = \sum_i \left| P_i^l(t) - \frac{d_i}{\sum_k d_k} \right| , \tag{10}$$

where $d_k = \sum_j W_{kj}$ is the weighted degree of node k . In our experiments, we iteratively calculate the error function given by Equation (10) until the value is below a preset error.

2.4 Improved EC Phase

The density of links is a very important criterion used for clustering networks. To take into account the link density, a number of criterion functions have been proposed for networks, such as average cut, normalized cut [17]. In graph theory, a cut corresponding to a bipartition of a network with the node set V is defined as:

$$cut(V_1,V_2) = \sum_{i\in V_1, j\in V_2} W_{ij} \ , \tag{11}$$

where W_{ij} is the weight of the edge <*i,j*>. In this paper, we define the cut as follows:

$$cut'(V_1,V_2) = \sum_{i\in V_1, j\in V_2} \frac{W_{ij}}{\sum_j W_{ij}} = \sum_{i\in V_1, j\in V_2} P_{i\to j} \ . \tag{12}$$

The optimal bipartition of a network is the one that minimizes the cut value, which is also called the minimum cut. In some cases, minimum cuts will lead to unnatural biased clustering results, where some partitions are simply isolated nodes. To alleviate this problem, the average cut has been proposed, which computes the density of links, and is defined as:

$$Acut(V_1,V_2) = \frac{cut'(V_1,V_2)}{|V_1|} + \frac{cut'(V_2,V_1)}{|V_2|} \ . \tag{13}$$

As indicated, if (V_1,V_2) is a "good" bipartition of communities, the *Acut* should be very small. Based on the sorted node list from FC phase, the main step of our EC subroutine is to calculate *Acut* for each possible cut through a top-down sorted node list, which is more simple than original EC phase. A smaller $Acut(V_1,V_2)$ indicates a better bipartition. Thus, the steps of our EC phase can be designed as follows:

Step 1 Calculate $cut'(V_1,V_2)$ for each position by top-down;

Step 2 Calculate the *Acut* for each position;

Step 3 Find the best cut position x that produces the local minimum value of *Acut*;

Step 4 $V = V_2$, and return to Step 1.

Then, we define $\overline{Acut}(V_1,V_1)$ to denote the probability of a random walker being trapped in set V_1 or V_2 .

$$\overline{Acut}(V_1,V_2) = \frac{cut'(V_1,V_1)}{|V_1|} + \frac{cut'(V_2,V_2)}{|V_2|} \ . \tag{14}$$

It is obvious that $\overline{Acut}(V_1,V_2)$ will be less than $Acut(V_1,V_2)$ when a network can be divided into two communities, otherwise, there is no community structure in the network and no further division is required. Based on this, a reasonable criteria to stop EC phase is:

$$Acut(V_1,V_2) \geq \overline{Acut}(V_1,V_2) \ . \tag{15}$$

We apply it to two real networks, Dolphin Network and American College football. The clustering result from this method is identical to the results from original EC phase. However, the method based on *Acut* has the advantage in time complexity. Original EC phase will take $O(m+n)$ time, where n and m correspond to the numbers of nodes and the links of the network, respectively. The method based on *Acut* needs to calculate the $P_{i\to j}$ on edges between different communities for each cut, which only takes $O(n)$ time.

3 Experiments

3.1 Data Sets

We apply our algorithm to the following real-world networks:

(1) Dolphin network, an undirected social network of frequent associations between 62 dolphins in a community living off Doubtful Sound;

(2) US college football network, which consists of 115 college teams represented as nodes and has edges between teams that played each other in the year 2000;

(3) Polbooks network, which consists of 105 vertices and 441 edges, represents a network of U.S. political books;

(4) Herbal network, which contains 642 edges and 332 vertices (herbs), represents the relationships among Chinese herbs. The network is created by ourselves based on combinational rule mining (see Fig. 2). All of these herbs are classified manually into 49 categories according to TCM (Traditional Chinese Medicine) expert.

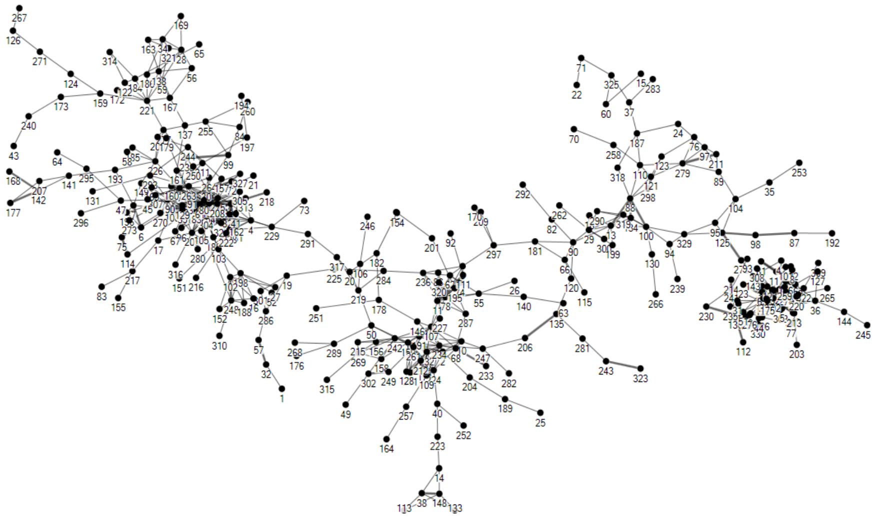


Fig.2. Herbal network

Most previous algorithms perform well on networks with good community structure, but perform poor on the network whose community structure is not well defined. Thus, we use the herbal graph, a not well defined network in community structure, to demonstrate whether our method is sensitive to the structure of networks.

3.2 Evaluation Measures

Two evaluations are considered, that are NMI (Normalized Mutual Information) and *Ncut*. NMI is used to evaluate the effectiveness of our proposed algorithm on randomly created network, and the *Ncut* are applied to the evaluation of group detection quality over real-world graphs.

NMI is currently very often used in tests of community detection algorithms based on information theory. Let M and N be two partitions, given a community i in M and a community j in N , let n_i^M , n_j^N , n_{ij}^{MN} be the sizes of community i , community j and their intersection, respectively, k^M , k^N be the numbers of communities in M and N . The closer M is to N , the larger *NMI* to be obtained. NMI is defined as:

$$NMI = \frac{-2\sum_{i=1}^{k^M}\sum_{j=1}^{k^N} n_{ij}^{MN} \log(\frac{n_{ij}^{MN} n}{n_i^M n_j^N})}{\sum_{i=1}^{k^M} n_i^M \log(\frac{n_i^M}{n}) + \sum_{j=1}^{k^N} n_j^N \log(\frac{n_j^N}{n})} . \qquad \textbf{(16)}$$

NCut is objective of the normalized cut algorithm. Given a group partition $C = \{C_1, C_2, ..., C_k\}$, the normalized cut is defined as：

$$NCut(C_1,...,C_k) = \sum_{i=1}^{K} \frac{Cut(C_i, \overline{C}_i)}{\sum_{p \in C_i} \sum_q W_{pq}} \quad (9), \tag{17}$$

where $\overline{C}_i$ denotes the set of nodes that are not in C_i and

$$Cut(C_i, \overline{C}_i) = \sum_{p \in C_i, q \in \overline{C}_i} W_{pq} . \tag{18}$$

Obviously, the smaller the value of *NCut* is, the better the partitioning quality becomes.

3.3 Overall Performance

To demonstrate the effectiveness of our method, we conduct two kinds of clustering methods. The first kind is based on Kmeans, and the second kind is based on different graph detection algorithms, such as original FEC, E_FEC, GN, CNM (Clauset-Newman-Moore) and MCL. Original FEC algorithm randomly select the sink node to calculate transition probability.

Clustering performance on randomly created networks. LFR benchmark was proposed by Lancichinetti et al. [20] to synthesize random networks with predefined community structures and power-law distributions. The model is defined as:

$$LFR(Num\_nodes, average\_k, \max\_\deg ree, e_1, e_2, \mu) . \tag{19}$$

where Num_nodes denotes the total number of nodes, $average\_k$ and $\max\_\deg ree$ are average degree, maximum degree, respectively. e_1 and e_2 are the exponent of the degree distribution and community size distribution, respectively. Mixing parameters μ is defined as the fraction of all edges going from a node that connect it to other communities. Communities are well defined when μ gets small. In our experiment, parameters for the model are set as follows: $LFR(500, 20, 40, 1, 1, \mu)$.

Fig. 3 shows the results of our analysis. Each point of every curve corresponds to an average over 10 realizations of the network. The variable on x-axis is the mixing parameter μ, the value on y-axis denotes NMI. As shown in Fig. 3, the difference among the performance of the algorithms is remarkable. Most methods start to fail when μ is close to 0.5. Compared with FEC, modularity-based methods (GN, CNM) perform poor, due to the well- known resolution limit problem. Among the random walk based methods (MCL, FEC and E_FEC), FEC and E_FEC obviously perform better than MCL. E_FEC performs best when $\mu > 0.3$. The performance shows that E_FEC is a stable and robust algorithm even the network does not define good community structure.

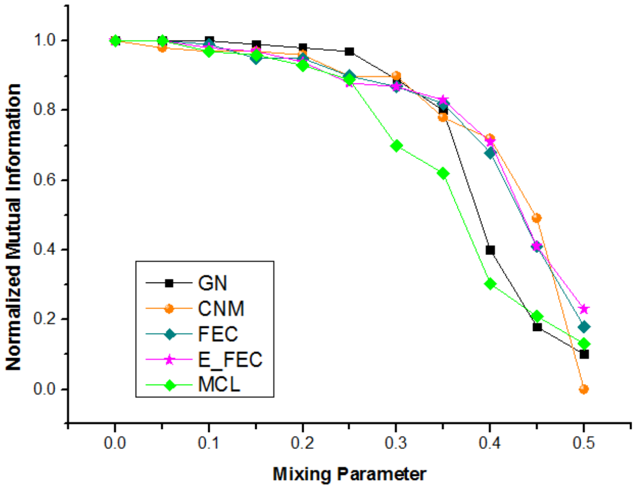
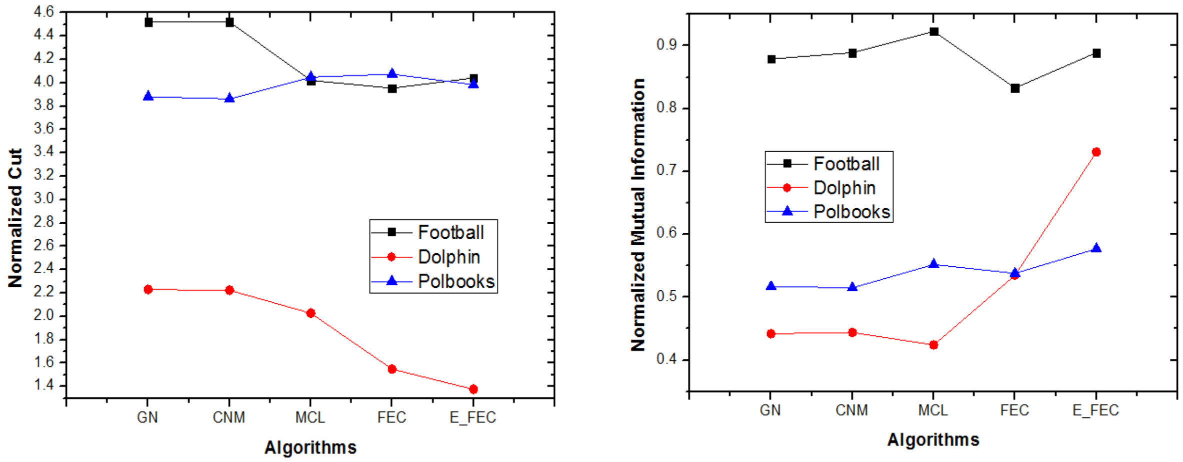


Fig. 3. Tests of algorithms on LFR benchmark

Clustering performance on real-world network. Firstly, we present the group detection analysis on three networks by E_FEC, FEC, MCL, GN and CNM. As shown in Fig. 4, the performance shows that compared with GN and CNM, the FEC algorithm slightly improves the performance of group detection for most of networks. Moreover, E_FEC obtains better performance compared with other random walk-based algorithms.



(a) Ncut (b) NMI

Fig. 4. Ncut and NMI obtained by five algorithms on three networks

Secondly, we discuss on the situation that whether our proposed clustering method performs better than traditional data clustering algorithm on the herbal network that does not have good community structure.

Note that GN and CNM should transform the weighted herbal graph to the unweighted one. As for the herbal clustering based on Kmeans, we construct an attribute vector for each herb. Let $\mathbf{X} = \{X_1, X_2, ..., X_n\}$ be a set of attribute vectors of n herbs. Then, the K-means algorithm partitions $\mathbf{X}$ into k clusters.

As shown in Table 2, it is obvious that the clustering performance of community detection algorithms (GN, CNM, FEC, MCL and E_FEC) is better than Kmeans. With respect to NMI measure, E_FEC performs best and achieves an increase of 0.118 compared with FEC. As shown in Fig. 5, the algorithm partitions the network into two clusters, which is consistent with the original group number. Other algorithms all partition the network into more than 2 clusters. As shown in Fig. 6, a total of 35 groups are detected by our E_FEC algorithm, and each detected group is located in a block with the nodes given the same color and shape. We can learn from the result that there are many edges within each group and relatively few between the groups. The method can effectively cluster some herbs with intensive connection.

Table 2. Herbal clustering results from different algorithms

Methods	NMI	*Ncut*	Cluster Number
Kmeans	0.472	7.139	38
GN	0.492	4.876	39
CNM	0.494	4.915	35
MCL	0.464	4.429	34
FEC	0.417	3.807	35
E_FEC	**0.535**	**3.641**	**35**

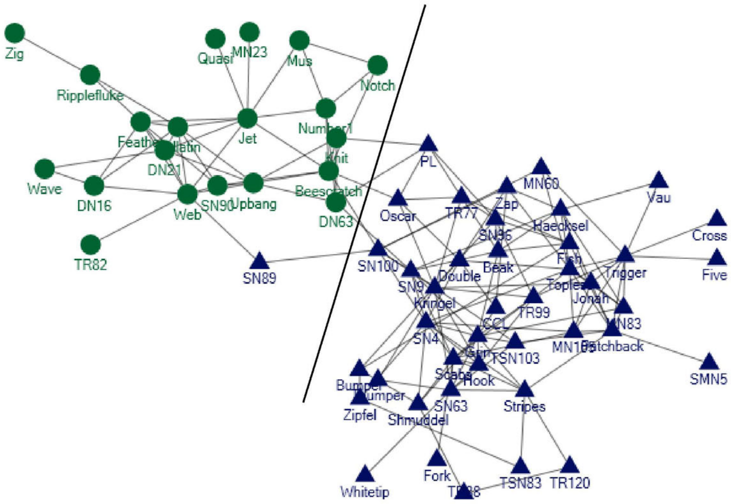


Fig. 5. Groups detected by E_FEC on dolphin network

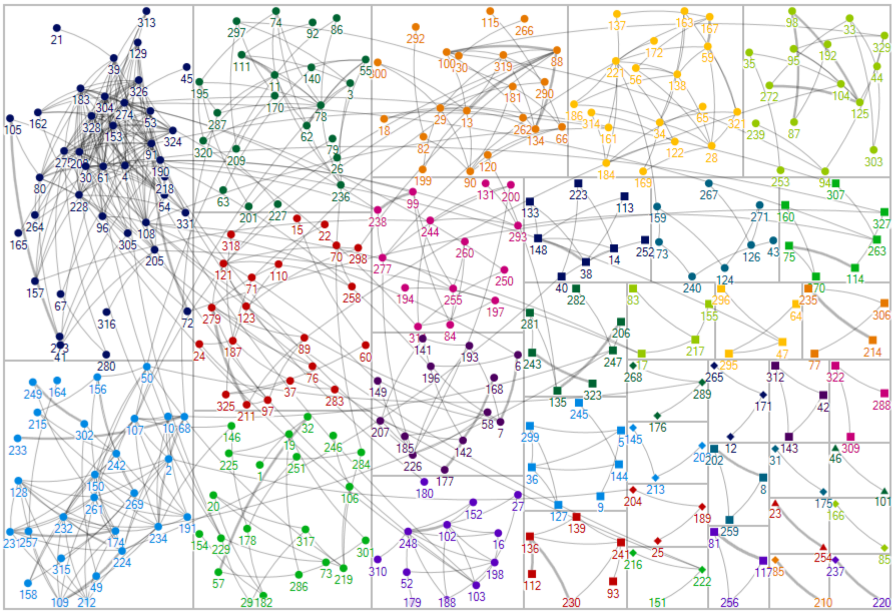


Fig. 6. Groups detected by E_FEC on herbal network

In general, the proposed E_FEC in our paper can definitely improve the performance compared with original FEC algorithm and several traditional algorithm. E_EFC is not sensitive to the structure of networks and needs no prior knowledge on the community structure (for instance, the number of communities or a good initial partition). Only one parameter (the step length l) is required in the improved FC phase.

Actual-time performance. We recorded the actual computational time needed for analyzing the networks. We ran the experiments on a computer with a CPU of 2.4 GHz, and the memory size is 8 Gbytes. The operating system was Windows 7, and the simulation was implemented and tested using Matlab 2012a. We repeated FEC and E_FEC 10 times for each network, and the averaged computational time taken was shown in Table 3.

Table 3. Actual running time of two algorithms on different networks

Networks	Running time(seconds)	
	FEC	E_FEC
Football	0.1136	0.1153
Dolphin	0.0311	0.0313
Herbal network	0.3373	0.3242
Random network($O(10^3)$ nodes)	2.3431	2.1894
Random network($O(10^4)$ nodes)	32.785	26.512

Based on Table 3, we note that when two methods are applied to real network with no more than $O(10^2)$ nodes, the running time is almost the same. Compared with FEC, it is clear that E_FEC uses less running time when applied to random network with more than $O(10^3)$ nodes.

4 Conclusions

This paper explores the defect of FEC algorithm and then extends it to E_FEC with several improvements on both FC phase and EC phase. Its performance with respect to effectiveness and efficiency has been validated by comparing it with other traditional methods against both synthetic and real-world networks. Experimental results show that E_FEC is able to achieve a good trade-off between effectiveness and efficiency, typically on herbal network that do not have good community structure. The method effectively produces a list of groups of functionally related herbs. The identification of herbal groups allows researchers find some similar herbs for their further study. However, E_FEC has its own disadvantages. Our method can not solve problems that identify communities of dynamic network which consists of a series of network snapshots. Also, during FC phase, the number of random walk steps has to be determined by iteratively calculation, which is inconvenient to conduct large-scale network analysis.

In our future work, we will focus on how to use E_FEC approach to further address, besides identifying communities from social networks, problems in other related domains such as big data mining on social network, group behavior mining. Additionally, with respect to the result of E_FEC, there may have several detected groups with large scale. For future work, we will focus on exploring adaptive algorithms to further subdivide large groups. Furthermore, the random walk scheme should be improved for detecting dynamic structures in some popular social networks.

Acknowledgements

This research was supported by Zhejiang Provincial Natural Science Foundation of China under Grant No.LQ14F020008, National Natural Science Foundation of China under Grant No.61602402, Chinese Knowledge Center for Engineering Science and Technology (CKCEST).

References

[1] S. Fortunato, D. Hric, Community detection in networks: a user guide, Physics Reports 659(2016) 1-44.

[2] L. Andrea, F. Santo, Community detection algorithms: a comparative analysis, Physics Review E 80(2010) 59-69.

[3] J. Mei, S. He, G. Shi, Z. Wang, W. Li, Revealing network communities through modularity maximization by a contraction-dilation method, New Journal of Physics 11(4)(2009) 59-71.

[4] M.E.J. Newman, Fast algorithm for detecting community structure in networks, Physical Review E 69(6)(2004) 066133.

[5] D. Jin, D.-Y. Liu, B. Yang, J. Liu, D.X. He, Y. Tian, Fast complex network clustering algorithm using local detection, Acta Electronic Sinica 39(11)(2011) 2540-2546.

[6] S. Fortunato, M. Barthelemy, Resolution limit in community detection, Proc. the National Academy of Science 104(1)(2007) 36-41.

[7] M.E.J. Newman, Equivalence between modularity optimization and maximum likelihood methods for community detection, Physical Review E 94(5)(2016) 052315.

[8] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proc. the National Academy of Science 99(12)(2002) 7821-7826.

[9] D.Y. Liu, D. Jin, D.X. He, J. Huang, J.N. Yang, B. Yang, Community mining in complex networks, Journal of Computer Research and Development 50(10)(2013) 2140-2154. (in Chinese)

[10] J. Ugander, L. Backstrom, Balanced label propagation for partitioning massive graphs, Proceedings of the sixth ACM international conference on Web search and data mining, WSDM6, February 6-8, Rome, Italy, ACM, New York, NY, USA, 2013, pp. 507-516.

[11] Z. Lin, X. Zheng, N. Xin, D. Chen, CK-LPA: efficient community detection algorithm based on label propagation with community kernel, Physica A: Statistical Mechanics and its Applications 416(2014) 386-399.

[12] Y. Xin, Z. Xie, J. Yang, An adaptive random walk sampling method on dynamic community detection, Expert Systems with Applications 58(2016) 10-19.

[13] S.V. Dongen, Clustering on graphs: The Markov cluster algorithm, [dissertation] Dutch, Netherlands: University of Utrecht, 2000.

[14] P. Pons, M. Latapy, Computing communities in large networks using random walk, Journal of Graph Algorithms Application 10(2006) 191-218.

[15] Y. Hu, M. Li, P. Zhang, Y. Fan, Z. Di, Community detection by signaling on complex networks, Physical Review E 78 (2008) 016115.

[16] W. Wang, D. Liu, X. Liu, L. Pan, Fuzzy overlapping community detection based on local random walk and multidimensional scaling, Physica A: Statistical Mechanics and its Applications 392(24)(2013) 6578-6586.

[17] B. Yang, W.K. Cheung W.K., J. Liu, Community mining from signed social networks, IEEE Trans. on Knowledge and Data Engineering 19(10)(2007) 1333-1348.

[18] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, D. Parisi, Defining and identifying communities in network, Proceedings of the National Academy of Science 101(9)(2004) 2658-2663.

[19] Y. Xin, Z. Xie, J. Yang, The adaptive dynamic community detection algorithm based on the non-homogeneous random walking, Physica A: Statistical Mechanics and its Applications 450(2016) 241-252.

[20] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Physical Review E 78(2008) 046110.

[21] W. Li, C. Huang, M. Wang, X. Chen, Stepping community detection algorithm based on label propagation and similarity, Physica A: Statistical Mechanics and its Applications 472(2017) 145-155.