Expert Systems With Applications 171 (2021) 114536

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Evolutionary community discovery in dynamic social networks via resistance distance

Weimin Li [a,*], Heng Zhu [a], Shaohua Li [a], Hao Wang [b], Hongning Dai [c], Can Wang [d], Qun Jin [e,f,*]

[a] School of Computer Engineering and Technology, Shanghai University, 99 Shangda Road, Baoshan District, Shanghai, China
[b] Department of Computer Science, Norwegian University of Science and Technology, NO-7491 Trondheim, Norway
[c] Faculty of Information Technology, Macau University of Science and Technology, Avenida Wai Long, Taipa, Macao
[d] School of Information and Communication Technology, Griffith University, 170 Kessels Road, Nathan Qld 4111, Australia
[e] College of Information Engineering, China Jiliang University, No. 258 Xueyuan Street, Xiasha Higher Education Park, Hangzhou, Zhejiang Province, China
[f] Faculty of Human Sciences, Waseda University, 1-104 Totsukamachi, Shinjuku-ku, Tokyo 169-8050, Japan

ARTICLE INFO

Keywords:
Dynamic social networks
Community discovery
Community evolution
Resistance distance

ABSTRACT

Traditional social community discovery methods concentrate mainly on static social networks, but the analysis of dynamic networks is a prerequisite for real-time and personalized social services. Through the study of community changes, the community structure in a dynamic network can be tracked over time, which helps in the mining of dynamic network information. In this paper, we propose a method of tracking dynamic community evolution that is based on resistance distance. Specifically, we model the time-varying features of dynamic networks using the convergence of a resistance-based distance. In our model, the heterogeneity of neighboring nodes can be obtained in the local topology of nodes by analyzing the resistance distance between nodes. We design a community discovery algorithm that essentially discovers community structures on dynamic networks by identifying the so-called core node. During the process of community evolution analysis, both the dynamic contribution of ordinary nodes and core nodes in each community are considered. In addition, to avoid the inclusion of spurious communities in the community structure, we define the notion of noise community and account for it in our algorithm. Experimental results show that the method proposed in this paper can yield better accuracy than other existing methods.

1. Introduction

Complex networks commonly have community structures, in which pairs of nodes belonging to the same community are closely connected, while pairs of nodes belonging to different communities are relatively sparsely connected. Community discovery is an important research area in the field of social network analysis, and many classical community discovery algorithms have been proposed in this sense (Gregory, 2009; Liu, Liu, & Jiang, 2017; Lyu, Shi, & Sun, 2019; Saad & Nosratinia, 2018; Teng, Liu, & Li, 2019; Zhang, Pan, Su, Zhang, & Niu, 2017; Zhou, Bo, & Jin, 2017). Traditional community discovery methods mostly focus on static networks, in which it is assumed that nodes and edges do not change over time. However, in realistic scenarios, users join and leave social networks in a dynamic way; consequently, the relationships between users change regularly. Therefore, dynamic social networks have received increasing attention. Fig. 1 is an example dynamic network with two snapshots, where each node represents a user, and the edges between nodes represent their relationship. Nodes and relationships will change from time $t-1$ to time t. Examples of changes in dynamic social networks include the addition of new members, the departure of previous members, and changes in the relationships between members (e.g., the establishment of new relations, the undermining of existing relationships, or changes in the weights of relationships). In a dynamic network, the community structure evolves with the dynamics of nodes and edges communities may merge, split, grow, shrink, be born, and die. Fig. 1 shows the merging process of dynamic social network communities. C_i^{t-1} and C_j^{t-1} merge to C_i^t at time t. Tracking community evolution is a major challenge in the analysis of

* Corresponding authors at: School of Computer Engineering and Technology, Shanghai University, China (Weimin Li); College of Information Engineering, China Jiliang University, China (Qun Jin).
E-mail addresses: wmli@shu.edu.cn (W. Li), awd82571@shu.edu.cn (H. Zhu), flowingfog@shu.edu.cn (S. Li), hawa@ntnu.no (H. Wang), hndai@ieee.org (H. Dai), wang@griffith.edu.au (C. Wang), jin@waseda.jp (Q. Jin).

https://doi.org/10.1016/j.eswa.2020.114536
Received 2 March 2020; Received in revised form 19 December 2020; Accepted 23 December 2020
Available online 29 December 2020

social network dynamics.

The concept of *distance* usually reflects the difficulty of information propagation between nodes in a social network. Node pairs in the same community are intimately connected; that is, paths between them are short, and there may be multiple paths between them. Thus, nodes tend to interact with other nodes in the same community, rather than with nodes outside the community. Nodes in the same community tend to have similar interests, so that information spreads more easily between them. As a result, the distance between two nodes in the same community should typically be short. Using the distance between nodes can help us discover the community structure. However, traditional distance measurement methods usually do not account for the time-varying nature of a dynamic network. Changes in nodes and the relationship between nodes have an essential impact on the stability and continuity of the community. How to better capture these changes and perform effective measurement is the challenge of community evolution analysis in a dynamic environment.

Dynamic networks can usually be represented by continuous-time slice networks. To determine the evolution of communities in a dynamic network, one can obtain the community structure at each time step, match communities at adjacent time steps, and calculate their similarity. The similarity between communities at adjacent time steps is usually measured by the number of overlapping nodes: generally, the more overlapping nodes, the more similar the communities are considered to be. However, this notion of similarity fails to account for the fact that different nodes in the same community may have different levels of importance to the community. Nodes of higher importance in a community should be more critical in identifying that community. Determining the significance of each node to the community it belongs to is a significant difficulty in analyzing the evolution of social networks.

To address the above challenges, in this paper, we propose a method called Evolutionary Community Discovery via Resistance distance (ECDR) for tracking community structure and evolution events in dynamic networks. In 1993, Klein and Randić proposed the resistance distance (Klein & Randić, 1993). Xujun Li et al. proposed a conductance eigenvector centrality (CEC) model to measure peer influence in the complex social network (Li, Liu, Jiang, & Liu, 2016). Wenjun Xiao et al. gave relevant proof for that the distance between two vertices of a graph G is equal to the resistance between the respective two points of an electrical network (Xiao & Gutman, 2003). In this paper, the network was modeled as a resistance network by treating the edges in the connected network as resistors. The equivalent resistance between nodes is the resistance distance between them, and the number of paths between nodes and the length of paths will affect their resistance distance.

In ECDR, the network is modeled as a resistance network, with the edges considered as resistors; this makes it possible to more accurately measure the distance between nodes, using jointly the number of paths between nodes and the length of the paths between them, both their resistance distance and their common neighbors are considered. An improved density-based method (Sander, Ester, Kriegel, & Xu, 1998) is utilized to determine the community structure. Both the dynamic contribution of nodes and the core nodes in the community are considered in the process of community evolution analysis. The ECDR method can reduce the time needed for comparison of communities at adjacent time steps. The evolution of the community is tracked by identifying birth, death, continuation, growth, shrinking, merging, and splitting events in the dynamic network. The main contributions of this work are as follows:

1. Based on the assumption that it is easier to propagate information between nodes in the same community, the network is modeled as a resistance network, in which each edge is regarded as a resistor. In such a resistance network, the equivalent resistance value between nodes reflects the difficulty of direct information transmission between them. The smaller the equivalent resistance between two nodes, the more likely they are to belong to the same community. Treating the equivalent resistance between nodes as the resistance distance fully accounts for the impact of the number of paths and the lengths of paths on information transmission.
2. The traditional density-based method requires the input of two global parameters, including the radius parameter ε and the minimum clustering parameter μ, consequently increasing the necessary prior knowledge and ignoring the differences in community density in the network. The community detection algorithm presented in this paper only requires the input of a single local parameter α to calculate the local threshold for each node, and it finds communities with different densities in the network.
3. Most community detection algorithms generate community structures that include noise communities, that is, communities with only one or two nodes. This paper analyzes the reasons for this situation in density-based algorithms and optimizes them based on local topological information about the noise communities.
4. In this paper, instead of matching all the community pairs on adjacent time slices, we match only those that have overlapping core nodes. This way dramatically reduces the number of matching calculations needed. We define the notion of *dynamic contribution degree*, and use this to give higher weight to the critical nodes in the community when performing matching calculations, thereby obtaining more accurate matching results.

The structure of the article is as follows. Section 2 describes the related work; Section 3 introduces the RDB approach and describe the process of it; Section 4 compares the RDB algorithm with other community detection algorithms and concludes in Section 5.

2. Related work

There are two main types of existing evolutionary community discovery methods: the incremental methods, and the time step methods (Spiliopoulou, 2011).

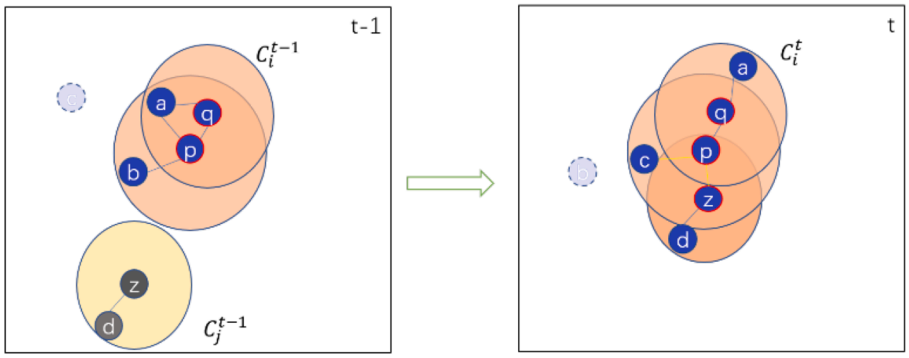

Fig. 1. An example dynamic social network.

2.1. Incremental methods

Based on the smoothness assumption, most of the topologies remain relatively stable, with only a small part of the structure varying during the evolution of dynamic communities. Therefore, incremental detection only recalculates the community membership of the changed part in the network while retains the permanent part of the community membership, thereby improving the computational efficiency.

In the dynamic network, identifying important nodes can help track communities. To find these important nodes, some distance and structure metrics were proposed. The TILES (Rossetti, Pappalardo, Pedreschi, & Giannotti, 2017) algorithm uses a triangular structure to find the core nodes in the community. If the node forms at least one triangle structure with other nodes in the community, it is the core node of this community, which can propagate the membership of the community. The HOCTracker (Bhat & Abulaish, 2015) divides the distance metric between nodes into two layers, and the evolution of the community is tracked through the method of label propagation. The concept of topological potential field models was used in Wang et al. (2018). Each node is a field source, and each field source forms a topological potential field around it. The topological potential field interaction makes each node have a specific topology potential value. The magnitude of the topological potential value represents the ability of the node to affect the surrounding nodes. According to the size of the topological potential value, the nodes are divided into three types: core nodes, overlapping nodes, and internal nodes. For increments, only the topological potential values of the newly emerging node and its neighbor nodes need to be calculated, and then the community structure is updated.

In Xin, Xie, and Yang (2016), random walk was used to simulate information propagation, and the distance between nodes is calculated by the frequency of information propagation back to the initial node. Such distance measurement method reflects the distance between two points in the information dissemination process, but it requires too many parameters to be input and is evenly random in the process of random walk, which cannot reflect the influence of edge weight or node similarity on propagation.

However, the incremental method intensely relies on the results of the initial community detection. If the initial community partitioning results are poor, it is difficult to correct in the later process. When the increment is generated, the algorithm only adjusts the communities of some affected nodes (usually the changing node and their neighbors) and ignores the impact on other nodes, which can cause errors. When the incremental scale increases, the error will accumulate.

2.2. Time step methods

Time step methods divide the network into time steps, and the community detection algorithm is executed at each time step. If the communities in adjacent time steps can be determined, we can judge the relationship between them and analyze their evolution. Although this method usually has a higher time consumption than the incremental algorithm, it can generally achieve higher accuracy.

Some time-step methods use the classic static community discovery algorithm on each time step to obtain the community structure on each time slice network. By using the topology of the network, a density-based clustering was used to find the local community at each time step (Kim & Han, 2009), and an information theory-based mapping method was proposed to identify the birth, death, and evolution of the community. An evolutionary co-clustering algorithm was proposed in Ji, Zhang, and Liu (2012) to discover communities in dynamic networks. To improve the classical spectral clustering method, a spectral clustering method suitable for dynamic network community discovery was proposed in Tang, Liu, and Zhang (2011). The algorithm proposed in Du, Jia, Gao, Gopalakrishnan, and Zhang (2015) can find the community structure at each time slice by defining the strength of the community, and track changes in community strength during the evolution of the community. P. Brodka et al. defined the social position to track the evolution of the community (Bródka, Saganowski, & Kazienko, 2013). The social status can reflect the importance of the node in the network, but it cannot reveal its stability in the dynamic network.

Community discovery in dynamic networks was also considered a multi-objective optimization problem (Mao-Guo, Ling-Jun, Jing-Jing, & Li-Cheng, 2012).One main object is the clustering accuracy of the current time step, and the other is the cluster drift between successive time steps. In Folino and Pizzuti (2014), a dynamic multi-objective genetic algorithm (DYNMOGA) was proposed to discover communities with genetic algorithms in dynamic networks. Normalized mutual information NMI and modularity are the two objects to be optimized. The concept of consensus community was proposed in Zhang and Chiang (2016) to minimize the clustering drift of adjacent time steps. By extracting the consensus community from the best community structure of the previous time step and retaining such a consensus community during the evolution process, the community at the current time step can have a similar evolutionary direction with the community in the previous time step. In addition, Ma and Di (2017) proposed evolutionary nonnegative matrix factorization (ENMF) frameworks that can maximize the accuracy of community detection and minimize the clustering drift between adjacent at the same time.

3. Evolutionary community discovery via resistance distance

This paper uses a resistance distance model to measure the closeness between nodes in a dynamic social network and proposes an improved density-based method for discovering the community structure at each time step. Finally, community evolution is tracked by performing the community matching calculations for adjacent time steps by using the dynamic contribution degree of each node. Unlike the traditional density-based method, our improved density-based method does not require the input of two global parameters, only one local parameter *alpha*, and local thresholds can be calculated from local information about each node at different times. The ECDR method can also manage noise communities (that is, unrealistic communities having only one or two nodes) that may appear in the output. This ensures that the community partitioning results are more accurate.

During the analysis of community evolution, community matching calculations are performed only on the community pairs that have overlapping core nodes at adjacent time steps; this reduces the number of matching calculations needed. To differentiate the nodes of a community for matching calculations, we define the notion of dynamic contribution of nodes; nodes with higher dynamic contributions are considered to be the more important nodes in a community. The ECDR method can track seven evolution events: birth, death, continuation, growth, shrinking, merging, and splitting.

3.1. Finding core node based on resistance distance

A dynamic network with T time steps can be represented by a network sequence $G = G_1, \cdots, G_T$. In this work, $G_t = (V^t, E^t)$ denotes the network structure at time step t ($1 \leqslant t \leqslant T$), G_t is a connected network, V_t denotes the node set in G_t network,and E_t denotes the set of edges in the G_t network.

In a network, some nodes with typical structure and attribute characteristics may form a community. Compared with cross-community nodes, information transmission between nodes in the same community is relatively smooth. Defining the distance between nodes as the difficulty of information transmission is in accordance with the notion of a community. For this definition, both the number of paths between two nodes and the length of each path will affect the cost of propagation. This is due to randomness in the choice of path and in whether a given attempt at transmission is successful or not. The shorter the path length between nodes and the greater the number of paths, the higher the probability of successful transmission, and the lower the cost of

transmission. Both the existence of multiple paths and the shortness of paths are essential factors in how closely connected a community is. Still, the general community partitioning algorithm cannot meet these two conditions for distance measurement at the same time.

One distance measure commonly used in social networks is the shortest path. Although such a path is the optimal path for disseminating information, in practice users may not know the shortest path to each other, so they cannot necessarily choose to use this path to propagate information. For example, in the network as shown in Fig. 2a, if node 6 wants to propagate a message to node 11, it will broadcast the message to its neighbor nodes without knowing the shortest path with each other. These neighbor nodes have a certain probability to forward this message so that it continues to propagate. In other words, there is some randomness in the choice of the final path through which information is successfully propagated to the target node. And because of the time-varying nature of a dynamic network, the shortest path between a pair of nodes may vary dramatically as nodes or edges in the network change. Distance measurement methods based on common neighbors such as Jaccard distance (Kosub, 2016), are also widely used. Such methods consider the impact of common neighbors on the similarity between nodes. However, information transmission between two nodes does not occur only through their common neighbors; it may also occur through paths that do not include common neighbors, and ignorance of this structural information may lead to inaccurate distance measurements. For example, node 2 and node 6 have a common neighbor (i.e., node 5), and node 2 and node 7 also have a common neighbor (i.e., node 1). Some distance measurements based on common neighbors cannot distinguish which of the two nodes is closer to node 2.

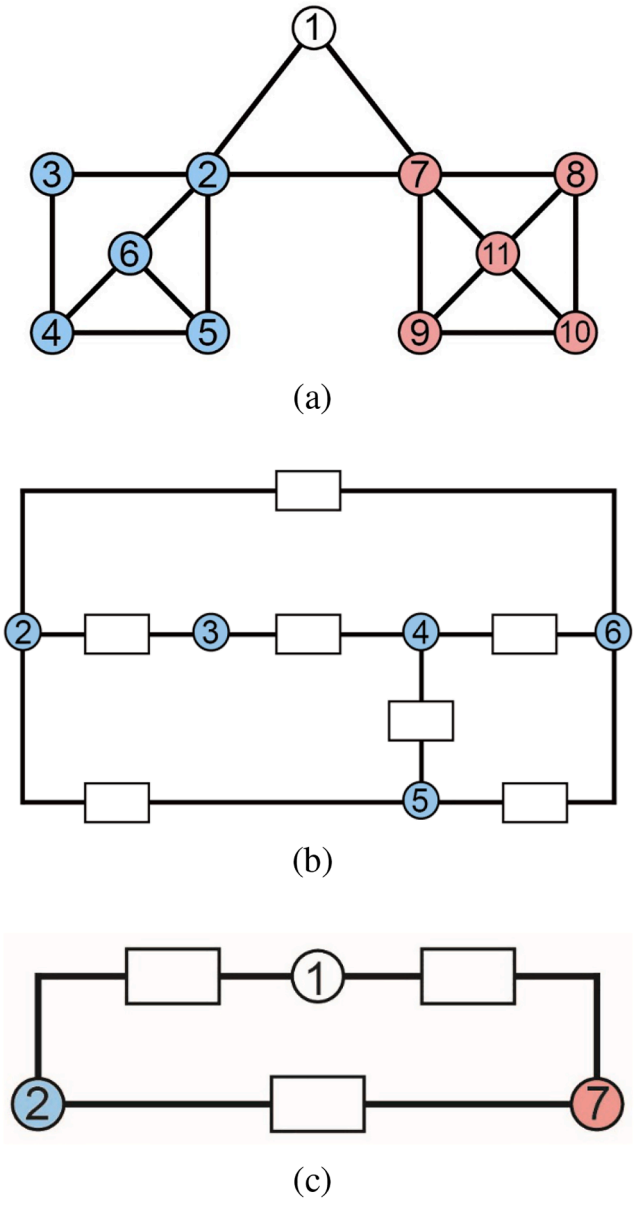

Fig. 2. (a) is an example unweighted network, (b) and (c) are partial network diagrams after the example network in (a) is partially converted into a restance network.

In order to take into account the impact of both the number of paths between nodes and the length of the paths on information transmission, this paper converts each time slice network into a resistance network like a circuit. The edges in the network are considered as resistances, and the resistance value is the inverse of the edge weight. This is 1Ω in the unweighted network, and the equivalent resistance between a pair of nodes is used as the resistance distance.

As shown in Fig. 2a, there are four paths between node 2 and node 6: $2\rightarrow6, 2\rightarrow5\rightarrow6, 2\rightarrow3\rightarrow4\rightarrow6, 2\rightarrow5\rightarrow4\rightarrow6$. After converting Fig. 2a into a resistance network, the resistance network between node 2 and node 6 is shown in Fig. 2b, path $2\rightarrow6$ is converted into a 1 Ω resistors in series, according to the calculation method of equivalent resistance of series resistors in physics, they represent a 2 Ω resistor, path $2\rightarrow3\rightarrow4\rightarrow6$ is converted into a 3Ω resistor, and path $2\rightarrow5\rightarrow4\rightarrow6$ is converted into a 3Ω resistor. Therefore, according to Ohm' law, the resistance distance between node 2 and node 6 is 0.54. Resistance distance can be used to express the influence of path distance and path length between nodes on information propagation. The resistance network between node 2 and node 7 is shown in Fig. 2c. The resistance distance between them is 0.67.

However, there are often a large number of edges in the network, which make the traditional equivalent resistance calculation method to be very complicated, so Klein and Randić (1993) proposed a method of calculating the resistance distance through the Laplacian matrix of the network in the connected network. The method proposed by Klein and Randić can convert any connected network into a circuit diagram and calculate the resistance distance between each pair of nodes. Although not every social network is a connected network, the nodes in the same community should belong to the same connected subgraph, so we only need to pay attention to the resistance distance of the node pairs in the same connected subgraph.

Once each time slice network is converted into a resistance network, the resistance distance between nodes on each time slice can be calculated (Klein & Randić, 1993). Some further refinements are then needed to produce a useful distance measurement for our purposes. It was found during experiments that if a node has a neighbor with a high degree, then the resistance distance with the node and the high degree neighbor is often minimal. This is because high degree nodes are connected to many other nodes in the network. Suppose node A has a high degree, and node C is one of its neighbors. Node A has many neighbors apart from node C, and there may be many paths between nodes A and C passing through these neighbors of node A. Even though these paths may be long, the large number of them greatly reduces the resistance distance between A and C. We hope to avoid this situation and find the neighbors of each node that are really closely connected to it, instead of just neighbors with high degrees. To compensate for such errors, we jointly consider the resistance distance between neighboring nodes and their common neighbors. The distance between neighboring nodes is measured as follows:

$$D^t_{pq} = R^t_{pq} * \min\left(1 - \frac{\sum_{u\in V^t_p \wedge u\in V^t_q} W^t_{pu}}{\sum_{u\in V^t_p} W^t_{pu}}, 1 - \frac{\sum_{u\in V^t_p \wedge u\in V^t_q} W^t_{pu}}{\sum_{u\in V^t_q} W^t_{pu}}\right) \quad (1)$$

where D^t_{pq} represents the distance between node p and node q time t, q is a neighbor node of p. V^t_p represents the neighbor set of p at time t,

r^t_{pq} represents the resistance distance of p and p at time t, and w^t_{pu} represents the edge weight between p and u at time t. Among all the neighbors of a node, the closeness of the connection can be compared by the distance metric. The neighbors with smaller distances are more likely to belong to the same community, but a threshold is needed to determine whether they belong to the community. In the traditional density-based clustering method, it is necessary to input the global threshold (ε), that is, for any node in the network, only neighbor whose distance is less than a uniform value ε may belong to the nodes in the same community. The setting of a global threshold not only increases the requirement for prior knowledge, but also fails to consider the different degrees of connection that occur within different communities in the network. Therefore, in this paper we leverage local topological information about each node to adaptively calculate its local close neighbor threshold. The node's local close neighbors are then defined based on its local close neighbor threshold.

Definition 1. **Local Close Neighbor Threshold**: The local close neighbor threshold ε^t_p of node p is the geometric mean of the distance of node p from all neighbors at time t. ε^t_p is defined as follows:

$$\varepsilon^t_p = \left(\prod_{q \in V^t_p} D^t_{pq} \right)^{1/|V^t_p|} \tag{2}$$

where $|V^t_p|$ represents the number of neighbor nodes of node p at time t.

Definition 2. **Local Close Neighbor**: Among all the neighbors of node p at time t, if the distance between p and q is not greater than the local close neighbor threshold ε^t_p of p or the local close neighbor threshold ε^t_q, then q is considered a local close neighbor of p.

The local close neighbor set of a node is defined as follows:

$$N^t_p = \{q : q \in V^t_p \wedge (D^t_{pq} \leqslant \varepsilon^t_p \vee D^t_{pq} \leqslant \varepsilon^t_q)\} \tag{3}$$

A local close neighbor of a node is a neighbor that is more closely connected to it than other nodes, and that may belong to the same community. The more nodes in a node's local close neighbor set, the more nodes are closely connected to it, and so more likely it is to be important to the community. In the traditional density-based methods, the global minimum clustering threshold μ is used. Nodes with a larger number of close neighbors than the minimum clustering threshold are considered as important nodes in the network. Using the global minimum clustering threshold has the same problem as using the global close neighbor threshold. It increases the requirement of prior knowledge and ignores the different shapes and sizes of different communities in the network. To find communities of different sizes, we use local information of nodes in this paper.

Definition 3. **Local Minimum Clustering Threshold**: Each node has its own local minimum clustering threshold μ^t_p according to its own local information and the local parameter α input by the user, and is defined as follows:

$$\mu^t_p = \alpha * |V^t_p| \tag{4}$$

Definition 4. **Core Node**: A node $p \in V^t$ is a core node if the number of its local close neighbor $|N^t_p|$ is not less than its local minimum clustering threshold μ^t_p. Core node set in time step t is given as follows:

$$\mathrm{Core}^t = \{p : p \in V^t \wedge \left|N^t_p\right| \geqslant \mu^t_p\} \tag{5}$$

If a node meets the conditions of the core node, there are many nodes around it that are closely connected with it, and these nodes are likely to be backbone nodes in the community. If node p is a core node, node p forms a community C^t_p together with the nodes in his close neighbor set N^t_p. If the two core nodes are local close neighbors of each other, then they are likely to belong to the same community. Therefore, the two communities with these cores were merged into one community.

3.2. Community discovery algorithm by searching the core node

For any community C^t_i in the network, we can define the core node set $CN^t_i = \{p : p \in Core^t \wedge p \in C^t_i\}$. Furthermore, for each of the core nodes in CN^t_i, all the core nodes in its local close neighbor set are also in CN^t_i. This set of core nodes merges their close neighbors with themselves into the same community through a process of community building and merging. The process of discovering the community structure in the network as shown in Algorithm 1 is to find the largest core node set of each community.

Algorithm 1. Community discovery algorithm based on searching core node

```
Input: G = {G_1, G_2, ..., G_T}, α.
Output: C = {C^1, C^2, ..., C^T}.
1: for G_t in G do
2:   Core^t = ∅, C^t = ∅, k = 1
3:   for node v ∈ V_t do
4:     Mark v as unvisited
5:     if node v is core node then
6:       add node v to core node set Core^t
7:     end if
8:   end for
9:   for unvisited node p do
10:    if p ∈ Core^t then
11:      Candidate=N^t_p
12:      C^t_k = {p}, mark p as visited
13:      while Candidate ≠ ∅ do
14:        q = Candidate.pop()
15:        if q ∈ Core^t then
16:          for node x ∈ N^t_q do
17:            if x is unvisited then
18:              add x into Candidate
19:            else if x is border then
20:              add x into C^t_k
21:            end if
22:          end for
23:          Add q into C^t_k and mark q as visited
24:        else if q ∈ Core^t and (q is marked as unvisited or border) then
25:          Add q into C^t_k and mark q as visited
26:        end if
27:      end while
28:      Add C^t_k into C^t
29:      k = k + 1
30:    else
31:      label p as a border
32:    end if
33:  end for
34:  for nodes in border do
35:    Label as an outlier
36:  end for
37: end for
```

In Algorithm 1, the parameters are initialized (line 2). $Core^t$ represents the core node set of the network at time t, C^t represents the community set of the network at time t, and k represents the number of communities. Each node in the network is then marked as unvisited, and the core nodes in the network are discovered (line 3). The following procedure is iterated over all unvisited nodes: if an unvisited node p is a core node, a new community is formed, and the largest connected core nodes set in this community and the set of local close neighbors of these core nodes are then determined (line 10–29). If p is not the core node, it is labeled as a border to temporarily (line 31). Nodes marked as borders may be given the identity of community members during the process of searching for community members (line 20), and if there are still border nodes after all unvisited nodes have been accessed, these nodes are

marked as outliers that do not belong to any community (line 34–36).

The key of Algorithm 1 is to determine whether a node is a core node based on the resistance distance. As we mentioned in Section 3.1, all connected network can be converted into resistance networks, through which the resistance distances between pairs of nodes in the same connected network can be obtained. Since in the process of finding the core node, we only care about the resistance distance between neighboring nodes, and the adjacent nodes must be connected and in the same connected network, so we can ensure that the core node can be found correctly by Algorithm 1.

For each time slice of T time steps, calculating the resistance distance between nodes is equivalent to inverting the n-order matrix, which takes $O(n^3)$. The process of identifying the community to which the node belongs is performed only once for each node. The time complexity of finding whether a node belongs to the community is $O(nlogn)$. Therefore, the time in the algorithm is mainly consumed in the calculation of the resistance distance between nodes, and the total time complexity of the algorithm is $O(Tn^3)$.

3.3. Optimize the quality of community discovery

The community sequence $C = \{C^1, C^2, \cdots, C^T\}$ is obtained using Algorithm 1, where C^t represents the community set in time step t.

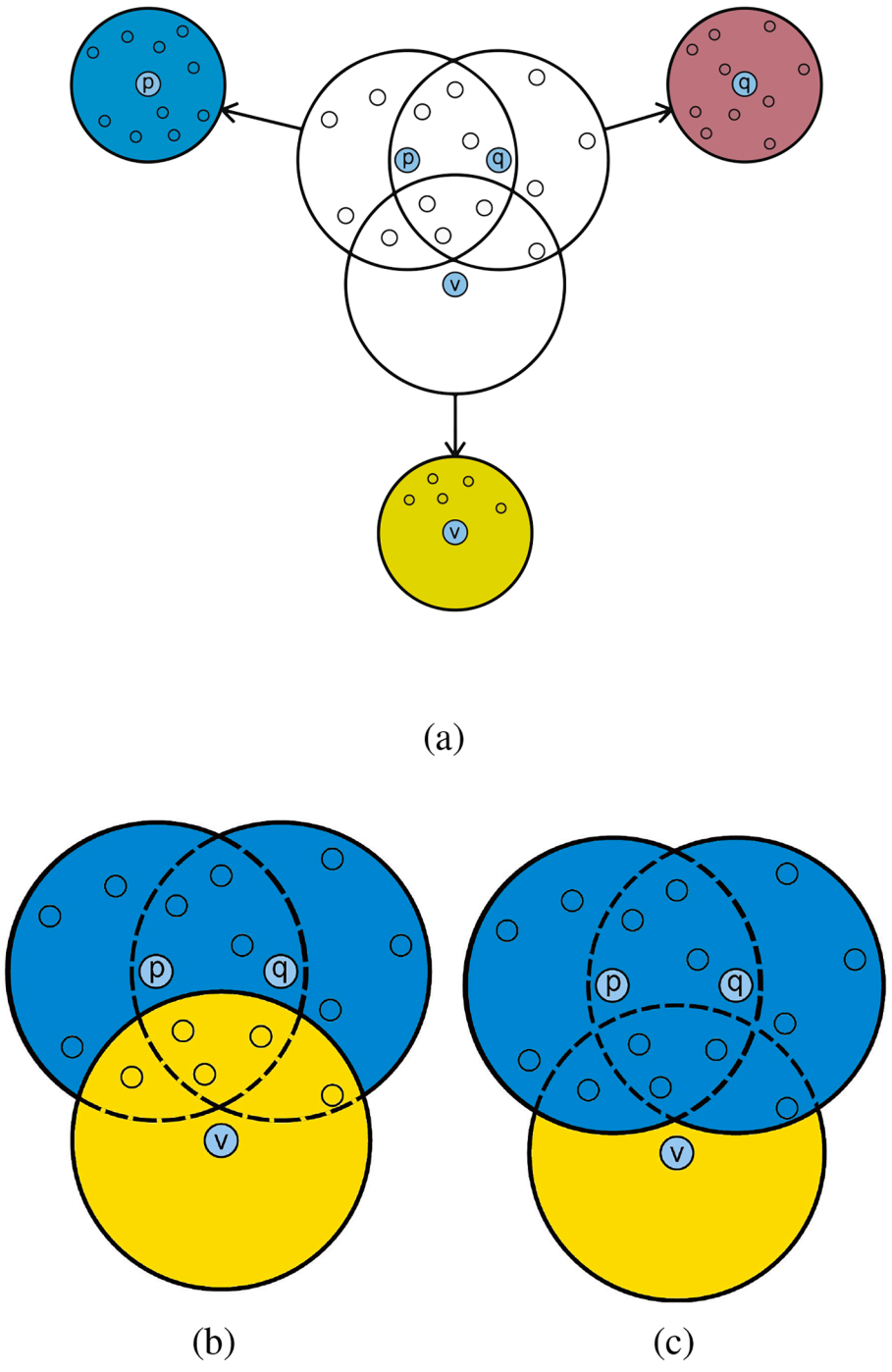

Fig. 3. (a) is a network with three core nodes (nodes that can form local communities). (b) is the community division in the case where v is the first discovered core node. (c) is the community division in the case where p or q is the first discovered core node.

However, in these community division results, it may appear that some communities contain only one or two nodes. Minimal sets of nodes cannot be tightly connected, and are therefore not considered to form realistic communities. They are instead considered noise communities; their inclusion in the community structure reduces the quality of the result.

Definition 5. **Noise Community**: The community C_i^t of the network at time step t is a noisy community if the number of members is less than 3 ($|C_i^t| < 3$). The noise community set at time step is represented by $Noise^t$.

The emergence of such noise communities is one of the problems caused by the characteristics of density-based non-overlapping community discovery methods. If an overlap is introduced into the community division, the possibility of noise communities can be greatly reduced. Many overlapping community discovery methods based on density have been proposed. But in some application scenarios, the community is required to be non-overlapping. In order to avoid the emergence of noise communities in the community structure of non-overlapping communities, we analyze the causes of noise communities in density-based non-overlapping community discovery methods and provides a solution to optimize the noise communities.

In non-overlapping community discovery, a node can belong to only one community, but since a node may be a local close neighbor of multiple nodes at the same time, different community structures may be obtained depending on the order in which the nodes were visited during the community structure discovered process. In Fig. 3, p, q and v are core nodes. The radius of the circle centered on the core node is the local neighbor threshold of the core node, and the hollow node in the circle is the local neighbor of the core node. The process of community discovery is a process of finding the largest set of connected core nodes and their connected core node sets. If the order of the core nodes is found to be different, the community structure may be also different. Fig. 3b shows an example of a community structure when v is the first discovered core node. In this case, the community to which v belongs has 6 nodes. But in Fig. 3c, if p or q is the first discovered core node, because p and q are both core nodes and q is a local close neighbor of p, both p and q and all their local close neighbors (all hollow nodes in Fig. 3) will be assigned to the same community C_1. When searching for a community from p, as p and q are both core nodes and q is a local close neighbor of p, it follows that p, q, and all their local close neighbors (all hollow nodes in Fig. 3) will be assigned to the same community C_1. Node v is also a core node and creates the community C_2, but as its local close neighbors are already members of another community, the community C_2 will have only one node. In other words, in some communities, the local close neighbors of the core nodes have already been assigned to other communities. This way results in some communities with a small number of nodes, which are noise communities. A community that steals nodes from a noise community is a marauder of that noise community.

Definition 6. **Marauder**: If the community C_j^t contains a local close neighbor of a core node in the noise community C_i^t, then C_j^t is called a marauder of C_i^t. The set of marauders of the noise community C_i^t is represented by M_i^t.

While the marauder weakens the noise community, it is also a community closer to the noise community in the network. Multiple communities may exist in the set of marauders of a noise community. In order to distinguish the distance between different marauders and the noise community, the community distance is defined.

Definition 7. **Community Distance**: The community distance between a noise community and its marauder is the minimum resistance distance between any pair of core nodes pair belonging to them, expressed as follows:

$$CD(i,j) = \min(R_{pq}^t) \quad (C_i^t \in Noise^t, C_j^t \in M_j^t) \tag{6}$$

where $CD(i,j)$ is the distance between the community C_i^t and C_j^t. p and q are the core nodes belonging to C_i^t and C_j^t respectively.

Definition 8. **Master Community**: In the marauder set M_i^t of the noise community C_i^t, the community with the shortest community distance from C_i^t is called the master community of C_i^t; it is represented by $Master_i^t$.

The master community of the noise community is the community closest to it in the network structure. By finding and merging the master for each noise community, the problem of noisy communities with concentrated output results is solved.

3.4. Community evolution based on dynamic contribution

In a dynamic network, the network changes dynamically as a result of events such as the joining of new members, the departure of previous members and changes in relationships between members (for example, the establishment of new relationships, the undermining of existing relationships, or changes to the weight of relationships). The community structure will also change with these dynamic changes and generate community evolution events. Evolution events are changes in the same community between two adjacent time slices. This article tracks seven types of community evolution events: birth, death, continuation, growth, shrinking, merging, and splitting.

Tracking community evolution events means to track communities on adjacent time slice networks, and the goal is to match pairs of communities C_1^{t-1}, C_2^t from adjacent time slices to determine whether C_2^t has evolved from C_1^{t-1}. The matching between community pairs is performed through the calculation of community similarity. Communities are considered successfully matched if they are sufficiently similar, i.e., if their similarity exceeds a certain threshold, in this case, there may be an evolutionary relationship. The evolution of the community on adjacent time steps is then judged by the matching situation and the community shape.

The similarity calculation between communities on adjacent time slices determines the accuracy of the evolution results. The traditional community evolution analysis method matches each community pair in adjacent time slices and only considers the overlap between community pairs in the community similarity calculation. The more nodes two communities have in common, the more similar they are. However, the members of a community have varying levels of importance to the community. For example, in various complex networks, especially social networks, the Pareto principle holds: approximately 20% or fewer of the nodes are leader nodes and have significantly more influence on the formation of the community than the other nodes. This feature has a significant impact on the evolution of network topology and the dissemination of information in the network. These leader nodes should play a more critical role in the community during the evolution of the network topology.

The core nodes found in the process of community discovery algorithms are critical nodes to the known communities. If two communities have no core node in common, they are unlikely to have an evolutionary relationship, so there is no need to perform similarity calculations on them. By judging the overlap of core nodes of community pairs on adjacent time slices, many unnecessary community similarity calculations can be reduced. The diversity of members in the community should also be taken into account when performing community similarity calculations. Based on the closeness of connection of a node to other members of the community, node contributions are defined as follows:

If node x belongs to the community C_j^t, then the contribution of node x to community C_j^t is given in Eq. (7):

$$CT_x^t = 1 - \frac{\sum_{y \in C_j^t} R_{xy}^t}{\sum_{Z \in C_j^t} \sum_{y \in C_j^t} R_{Zy}^t} \tag{7}$$

The higher the contribution of a given node, the higher its degree of connection to other nodes in the community, and the greater its contribution to the overall closeness of the community. Note that in a dynamic network, the contribution of a node may be different in different time slices. However, a node with a relatively stable contribution can provide stable support for the closeness of a community. Therefore, the dynamic contribution is defined using Eq. (8):

$$IM_x^t \begin{cases} CT_x^t & ,t = 1 \\ CT_x^t - CT_x^t * (CT_x^t - IM_x^{t-1}), IM_x^{t-1} \neq 0. \\ CT_x^t * 0.5 & ,IM_x^{t-1} = 0 \end{cases} \tag{8}$$

where IM_x^t denotes the dynamic contribution of node x at t. Because nodes with larger dynamic contributions are more important in identifying the community, the dynamic contribution of overlapping nodes should be considered when calculating the similarity of communities at adjacent time steps. Therefore, if C_i^t and C_j^{t+1} are the two communities at time i and $i+1$, respectively, their similarity is calculated as follows:

$$I(C_i^t, C_j^{t+1}) = \frac{\sum\limits_{x \in (C_j^t \cap C_j^{t+1})} IM_x^t + \sum\limits_{x \in (C_j^t \cap C_j^{t+1})} IM_x^{t+1}}{\sum\limits_{x \in C_i^t} IM_x^t + \sum\limits_{x \in C_i^{t+1}} IM_x^{t+1}} \tag{9}$$

If $I(C_i^t, c_j^{t+1}) \geqslant \beta$, then C_i^t and C_j^{t+1} are considered to be successfully matched; β is an input parameter and can be adjusted as needed. The occurrence of evolutionary events is analyzed according to the matching situation and the community shape. The analysis process of the 7 evolutionary events is as follows:

1. Death: Community C_i^t is death if there is no community at C_j^{t+1} match success with it.
2. Birth: Community C_j^{t+1} is birth if there is no community at C_j^{t+1} match successfully with it.
3. Continuation: Community C_i^t is defined as a continuation to C_j^{t+1} if C_i^t match successful with C_j^{t+1} and $|C_i^t| = |C_j^{t+1}|$, this means the amount of nodes in both communities is the same, with only a few nodes that may be different.
4. Shrinking: If community C_i^t only match successful with C_j^{t+1} in time $t+1$ and $|C_i^t| > |C_j^{t+1}|$, then C_i^t shrinks to C_j^{t+1}. If community match successful with several communities $\{C_j^{t+1}, C_{j+1}^{t+1}, \ldots, C_{j+n}^{t+n}\}$ in time t + 1 and only the size of C_j^{t+1} is smaller than C_i^t ($|C_i^t| > |C_j^{t+1}|$), then C_i^t shrinks to C_j^{t+1}.
5. Growth: If community C_i^t only match successful with C_j^{t+1} in time t + 1 and $|C_i^t| < |C_j^{t+1}|$, then C_i^t is grow to C_j^{t+1}. If community C_i^t match successful with several communities $\{C_j^{t+1}, C_{j+1}^{t+1}, \ldots, C_{j+n}^{t+1}\}$ in time t + 1 and only the size of C_j^{t+1} is bigger then C_i^t ($|C_i^t| < |C_j^{t+1}|$), then C_i^t is grow to C_j^{t+1}.
6. Merging: If community C_j^{t+1} match successful with several communities $\{C_i^t, C_{i+1}^t, \ldots, C_{i+n}^t\}$ in time t and more than one of those communities have smaller size than C_j^{t+1}, then those smaller communities merge to C_j^{t+1}.
7. Splitting: If community C_i^t match successful with several communities $\{C_j^{t+1}, C_{j+1}^{t+1}, \ldots, C_{j+n}^{t+1}\}$ in time t + 1 and more than one of those communities have smaller size than C_i^t, then C_i^t split into those smaller communities.

Note that in order to reduce the matching time, only pairs of community pairs from adjacent time steps having at least one common core node will be matched.

4. Experiment

In this section, the ECDR method proposed in this paper is evaluated in comparison to representative state-of-the-art community detection methods on both dynamic and static networks. It should be noted that, to the best of the authors' knowledge, there is no generally accepted method of measuring the quality of the results of community evolution analysis; most methods only provide the final community structure at a given time step. Therefore, for all of the algorithms, the results of community detection at various time steps or on static networks are provided. At the same time, evolution analysis is performed only for the evolution results of the ECDR.

4.1. Evaluation criteria

In this paper, Normalized mutual information (NMI) and modularity scores are used as evaluation criteria to measure the performance of ECDR.

NMI is used to measure the similarity between the community results discovered by the community division algorithm and the real community structure of the network. The calculation formula is defined as follows:

$$\mathrm{NMI}(A,B) = \frac{-2\sum\limits_{i=1}^{C_A}\sum\limits_{j=1}^{C_B} c_i \log\left(\frac{C_{ij})^{*N}}{C_i * C_i}\right)}{\sum\limits_{i=1}^{C_A} C_i \log\left(\frac{C_i}{N}\right) + \sum\limits_{j=1}^{C_B} c_{ij} \log\left(\frac{C_i}{N}\right)} \tag{10}$$

where N represents the number of nodes in the network, A and B respectively represent the community results discovered by the community discovery algorithm and the real community structure, C is a mixed matrix and the element C_{ij} in the matrix represents the number of nodes belonging to the ith community in the community structure A, and belonging to jth community in community structure B. $C_A(C_B)$ represents the number of communities in the community structure A (B), and $C_{i\cdot}(C_j)$ represents the sum of the elements in the matrix C.

The value range of NMI is [0, 1]. The larger the value, the more similar the community structure discovered by the community division algorithm is to the real community structure, and the more accurate the algorithm is. Otherwise, the less precise the algorithm is.

Modularity is a global objective function used to evaluate the quality of the community division algorithm when the real community structure of the network is not known. It is defined as follows:

$$Q = \frac{1}{2m}\sum_{i \neq j}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta(C_i, \quad C_j) \tag{11}$$

where A_{ij} represents the value of row i and column j in the adjacency matrix corresponding to the network, that is, the connection between node i and node j. If there is an edge connection between node i and node j, A_{ij} is 1, otherwise it is 0 (here only consider the case of undirected and unweighted graphs). m represents the total number of edges in the network, and k_i represents the degree of the i-node. $\delta(C_i, C_j)$ is used to judge whether node i and node j are in the same community. If they are in the same community, $\delta(C_i, C_j)$ of = 1, otherwise $\delta(C_i, C_j)$=0. The value range of modularity is [0,1]. When all communities in the network belong to the same community, the modularity is 0, which means that the network is a random network, and there is no community structure. The greater the modularity, the better the quality of the results found by the community.

4.2. Static real network

Five real static networks were used to evaluate ECDR and the comparative algorithms. Normalized mutual information (NMI) and modularity scores were used to evaluate the performance of these

algorithms. Two scores are often used as the measurement criterion of community detection methods including (Chang, Feng, & Ren, 2017; Fionda & Pirro, 2018; Pizzuti, 2018; Pizzuti & Socievole, 2019; Teng et al., 2019; Zhang, Zhou, Pan, Zhang, & Jin, 2018). The five static real networks include the American College football network (Girvan & Newman, 2002), which has 115 nodes represent the football team, those nodes can be divided into 11 large communities and 5 independent groups. There are 602 edges in this network, each edge between two nodes represents there is a match between two teams (NCAA), the Dolphin Network (Lusseau et al., 2003), which is an undirected and unweighted network consisting of 62 frequently contacted dolphins, divided into two communities (DOLP), the Polbooks network, with a network contains 105 books on American politics sold on Amazon (POLB), the Zachary's karate network (Zachary, 1976) consists of 34 nodes represent the members of a Karate club, 78 edges represent the friendship of them. There are two community in this network each with a core member (KARA), a collaboration network (Burt, 1987) of four communities consist by 241 physicians (PHYS). Those five real networks have been widely used to evaluate algorithm performance in social network filed and each of them have the benchmark community structure. The algorithms used for comparison include COPRA (Gregory, 2009), SHRINK (Huang, Sun, Han, & Feng, 2011), SLAP (Xie & Szymanski, 2012).

Figure 4 shows the modularity results of the ECDR method and other baseline methods. It can be seen that ECDR reach the same effective on modularity with the other methods overall. And COPRA is unstable on the KARA network. Figure 5 presents the NMI results. It can be seen that ECDR has superior NMI performance compared to most networks, and obtains the better NMI score on DOLP and PHYS. In the KARA network which has two communities formed by the administrator and the instructor of the club, the ECDR algorithm can determine the two communities in this network and identify nodes representing the administrator and instructor as the core nodes. The value of the parameter α affects the result significantly. If a low value is chosen for α, this means that nodes can easily become core nodes. A higher number of core nodes makes community membership more easily spread, which tends to form loosely connected large communities. On the other hand, if a high value is chosen for α, then it is more difficult for nodes to become core nodes, so the resulting communities are usually tightly connected. However, too high a choice of α also means that too few nodes are eligible to become core nodes, so many nodes in the network are not neighbors of any core nodes and therefore do not belong to any community. The value of α should be adjusted according to the requirements for community density. In this paper, the suggested value of α is in the range [0.4,0.6], as used in the experiment. For this range of values, it is possible to identify core nodes that are truly closely connected to neighbor nodes, not just nodes that have a large degree.

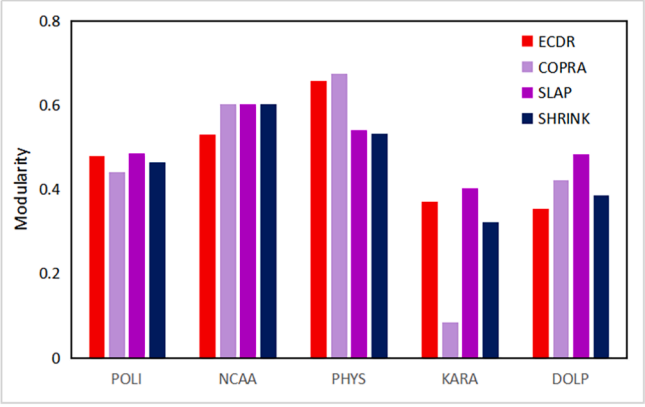

Fig. 4. Modularity score on real network.

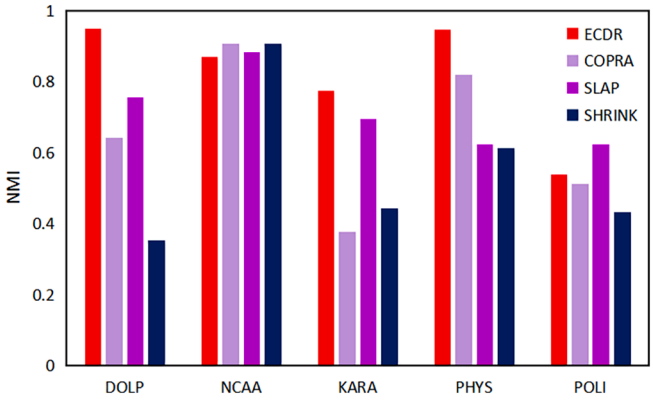

Fig. 5. NMI score on real network.

4.3. Dynamic real network results

The DBLP Co-authorship dataset is part of the DBLP dataset, and is a network of 2753 authors who published articles between 1990 to 2010. Each existing edge represents two authors have co-authored papers from 1990 to 2010. This dynamic network is divided into 9-time steps by time and has no benchmark community structure.

As illustrated in Table 1, from the time T1 to T5, as the network is first established, many communities are born. At later time steps, the number of communities experiencing birth and death decreases. The number of splitting and merging events is relatively small, while the number of shrinking and growth events is relatively high. This indicates that the authors included in the DBLP data tended to have long-term and stable cooperation, which is consistent with reality.

Threshold β is a parameter of great importance in evolution analysis and controls the accuracy of evolution results. The change of the DBLP dataset corresponding to the change of the parameter β and the number of growth time is illustrated in Fig. 6and 7. It can be observed that the number of shrinking and growth events decreases as β increases, which is because the value of β reflects the degree of tolerance for noise. The smaller the value of β, the easier it is for communities to match. If β is too small, many matching communities will have low similarity. In other words, although the number of events is increased, the accuracy is reduced. If the value of β is too large, then many similar communities cannot match. As seen in Figs. 6 and 7, the number of shrinking and growth events is relatively stable when β is 0.3 and 0.7. The same is true for the other events, except for death and birth. The number of death events increases as β increases, because for larger values of β it is more difficult for the communities at time t and at time $t+1$ to match. When β is 1, all communities at time t will be judged as dead at time $t+1$, except those that are completely unchanged. The trend of the birth events with respect to β is the same as that of death events.

4.4. Synthetic dynamic network

Lancichinetti-Fortunato-Radicchi (LFR) is commonly used to evaluate community detection methods with synthetic networks (Andrea & Santo, 2009). However, as LFR can only generate static synthetic network, Greene, Doyle, and Cunningham (2010) extend the LFR-generator to create dynamic network with several time steps. Using this generator, three dynamic networks were generated in this work ($N = 15000, k = 20, K_{max} = 40, C_{min} = 20, C_{max} = 60, \tau_1 = -2, \tau_2 = -1, \mu = 0.2, O_n = 0, O_m = 1$). The networks include *BirthDeath*, in which birth and death events will occur between an adjacent time steps; *MergeSplit*, in which merging and splitting events will occur between adjacent time steps; and *ShrinkGrow*, in which shrinking and growth events will occur between adjacent time steps. Each dynamic network had 10 time steps and 15,000 nodes. The ECDR method and three other methods, namely COPRA, SHRINK, and SLAP were applied to these

Table 1
Number of Evolve Event.

	Birth	Death	Continuing	Shrinking	Growing	Merging	Splitting
T1-T2	29	11	8	10	15	1	0
T2-T3	25	15	7	11	21	5	1
T3-T4	35	13	4	14	14	4	2
T4-T5	20	13	7	16	12	4	1
T5-T6	11	13	2	8	20	3	1
T6-T7	5	7	5	6	14	1	2
T7-T8	3	5	4	10	10	2	2
T8-T9	6	5	1	12	13	1	1

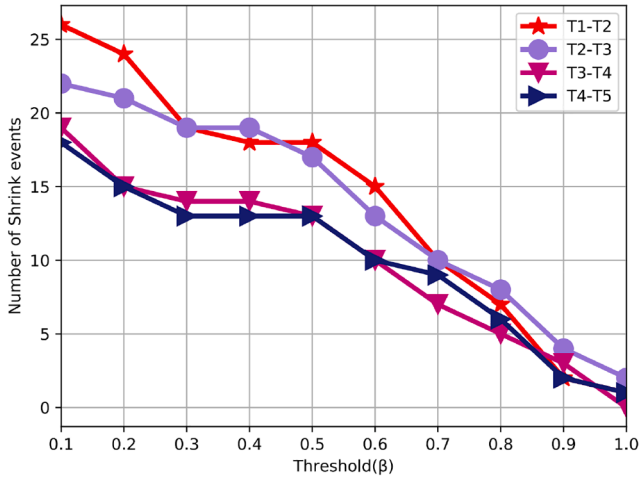

Fig. 6. Number of shrinking events.

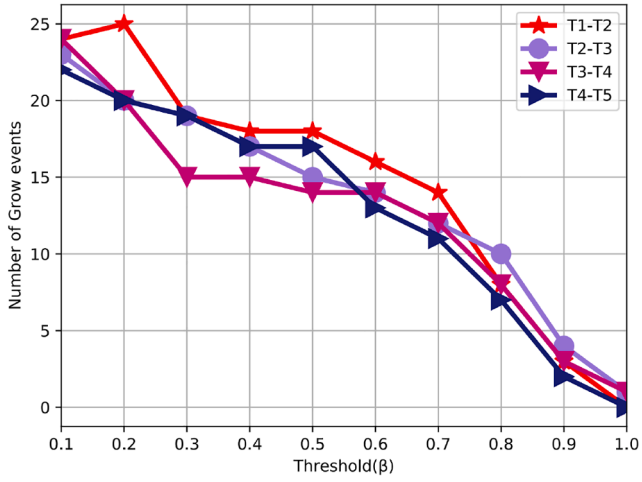

Fig. 7. Number of growth events.

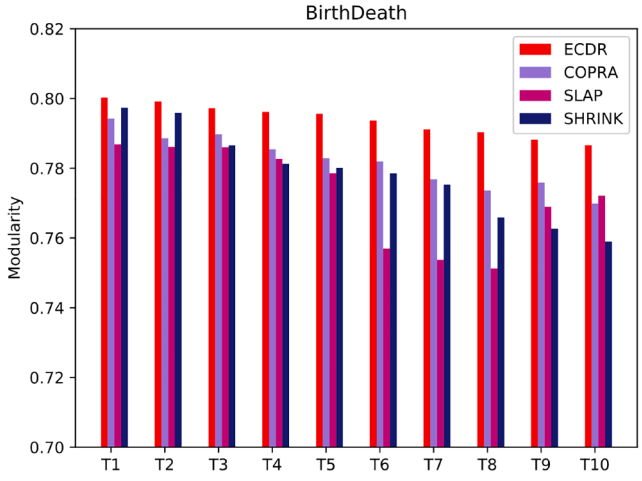

Fig. 8. Modularity score on BirthDeath.

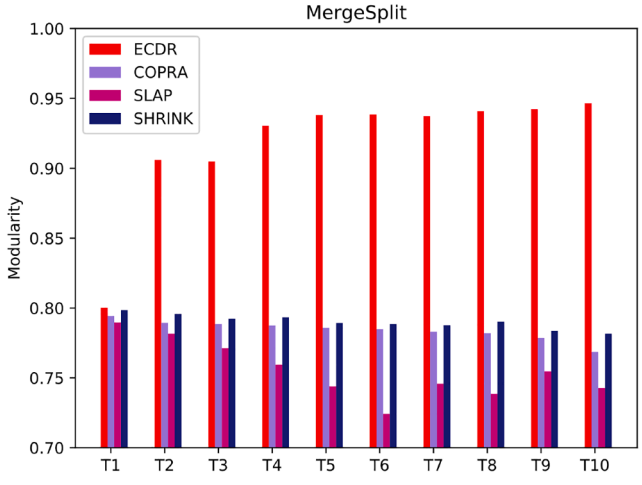

Fig. 9. Modularity score on MergeSplit.

networks and their modularity scores were compared.

Figs. 8–10 show the modularity score of the community structure obtain by the four methods on the three synthetic dynamic networks. It can be observed that ECDR perform better than the other methods, as the modularity scores are mostly close to 1 for all those networks, even in the most challenging case (*BirthDeath*) and it remains stable over ten time steps.

5. Conclusion

This paper presented the ECDR approach for determining community structures and tracking community evolution in dynamic networks. Unlike most existing methods, the ECDR method treats the network as a resistance network; it calculates the resistance distance between nodes and jointly considers the resistance distance and the common neighbors of the nodes to calculate the distance between them. This distance measurement method better reflects the tightness of node connections in a community context. An improved density-based method is utilized to determine the community structure at each time step; then the noise communities generated by the density-based clustering method are re-divided to produce more authentic community segmentation results. The core nodes found during the process of community detection are then used for tracking community evolution. Only communities from adjacent time steps with common core nodes are compared, reducing the number of comparisons needed. When computing the similarity of a

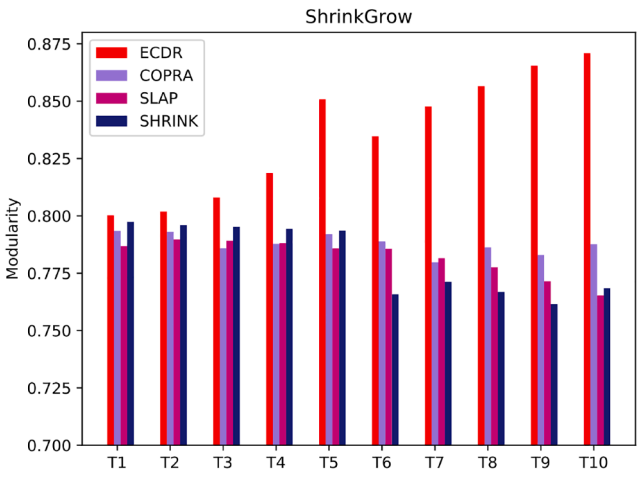

Fig. 10. Modularity score on ShrinkGrow.

pair of communities from adjacent time steps, the dynamic contribution of each node is calculated, and the nodes that are most representative of each community are identified. Our experimental results show that the ECDR method obtains superior results compared to other existing methods on well-known data sets.

CRediT authorship contribution statement

Weimin Li: Methodology, Writing - review & editing. **Heng Zhu:** Conceptualization, Writing - original draft. **Shaohua Li:** Investigation. **Hao Wang:** Writing - review & editing. **Hongning Dai:** Writing - review & editing. **Can Wang:** Writing - review & editing. **Qun Jin:** Resources.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported by the National Key R&D Program for China (No. 2017YFE0117500).

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, athttps://doi.org/10.1016/j.eswa.2020.114536.

References

Andrea, L., & Santo, F. (2009). Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E Statistical Nonlinear & Soft Matter Physics, 80*, Article 016118.

Bhat, S. Y., & Abulaish, M. (2015). Hoctracker: Tracking the evolution of hierarchical and overlapping communities in dynamic social networks. *IEEE Transactions on Knowledge & Data Engineering, 27*, 1019, 1013.

Bródka, P., Saganowski, S., & Kazienko, P. (2013). Ged: the method for group evolution discovery in social networks. *Social Network Analysis & Mining, 3*, 1–14.

Burt, R. S. (1987). Social contagion and innovation: Cohesion versus structural equivalence. *American Journal of Sociology, 92*, 1287–1335.

Chang, H., Feng, Z., & Ren, Z. (2017). Community detection using dual-representation chemical reaction optimization. *IEEE Transactions on Cybernetics, 47*, 4328–4341.

Du, N., Jia, X., Gao, J., Gopalakrishnan, V., & Zhang, A. (2015). Tracking temporal community strength in dynamic networks. *IEEE Transactions on Knowledge & Data Engineering, 27*, 3125–3137.

Fionda, V., & Pirro, G. (2018). *Community deception or: How to stop fearing community detection algorithms*. IEEE Transactions on Knowledge & Data Engineering. pp. 1–1.

Folino, F., & Pizzuti, C. (2014). An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge and Data Engineering, 26*, 1838–1852. https://doi.org/10.1109/TKDE.2013.131

Girvan, M., & Newman, M. E. J. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences, 99*, 7821–7826. https://doi.org/10.1073/pnas.122653799

Greene, D., Doyle, D., & Cunningham, P. (2010). Tracking the evolution of communities in dynamic social networks. In Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining ASONAM '10 (p. 176–183). USA: IEEE Computer Society.https://doi.org/10.1109/ASONAM.2010.17.

Gregory, S. (2009). Finding overlapping communities in networks by label propagation. *New Journal of Physics, 12*, 2011–2024.

Huang, J., Sun, H., Han, J., & Feng, B. (2011). Density-based shrinkage for revealing hierarchical and overlapping community structure in networks. *Physica A Statistical Mechanics & Its Applications, 390*, 2160–2171.

Ji, S., Zhang, W., & Liu, J. (2012). A sparsity-inducing formulation for evolutionary co-clustering. In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD '12 (p. 334–342). New York, NY, USA: Association for Computing Machinery.https://doi.org/10.1145/2339530.2339586.

Kim, M. S., & Han, J. (2009). A particle-and-density based evolutionary clustering method for dynamic networks. *Proceedings of the Vldb Endowment, 2*, 622–633.

Klein, D. J., & Randić, M. (1993). Resistance distance. *Journal of Mathematical Chemistry, 12*, 81–95.

Kosub, S. (2016). A note on the triangle inequality for the jaccard distance. *Pattern Recognition Letters, 120*.

Li, X., Liu, Y., Jiang, Y., & Liu, X. (2016). Identifying social influence in complex networks: A novel conductance eigenvector centrality model. *Neurocomputing, 210*, 141–154.

Liu, C., Liu, J., & Jiang, Z. (2017). A multiobjective evolutionary algorithm based on similarity for community detection from signed social networks. *IEEE Transactions on Cybernetics, 44*, 2274–2287.

Lusseau, D., Schneider, K., Boisseau, O. J., Haase, P., Slooten, E., & Dawson, S. M. (2003). The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations. *Behavioral Ecology & Sociobiology, 54*, 396–405.

Lyu, C., Shi, Y., & Sun, L. (2019). A novel local community detection method using evolutionary computation. *IEEE Transactions on Cybernetics*, 1–13. https://doi.org/10.1109/TCYB.2019.2933041

Ma, X., & Di, D. (2017). Evolutionary nonnegative matrix factorization algorithms for community detection in dynamic networks. *IEEE Transactions on Knowledge & Data Engineering, 29*, 1045–1058.

Mao-Guo, G., Ling-Jun, Z., Jing-Jing, M., & Li-Cheng, J. (2012). Community detection in dynamic social networks based on multiobjective immune algorithm. *Journal of Computer Science and Technology, 27*, 455. https://doi.org/10.1007/s11390-012-1235-y

Pizzuti, C. (2018). Evolutionary computation for community detection in networks: a review. *IEEE Transactions on Evolutionary Computation, 22*, 464–483.

Pizzuti, C., & Socievole, A. (2019). *Multiobjective optimization and local merge for clustering attributed graphs* (pp. 1–13). IEEE Transactions on Cybernetics.

Rossetti, G., Pappalardo, L., Pedreschi, D., & Giannotti, F. (2017). Tiles: an online algorithm for community discovery in dynamic social networks. *Machine Learning, 106*, 1213–1241.

Saad, H., & Nosratinia, A. (2018). Community detection with side information: Exact recovery under the stochastic block model. *IEEE Journal of Selected Topics in Signal Processing, PP*. pp. 1–1.

Sander, J., Ester, M., Kriegel, H., & Xu, X. (1998). Density - based clustering in spatial an efficient k - means clustering Algorithm 1175 databases: the algorithm gdbscan and its applications. *Data Mining and Knowledge Discovery – DATAMINE.*

Spiliopoulou, M. (2011). Evolution in social networks: A survey. In C. C. Aggarwal (Ed.), Social Network Data Analytics (pp. 149–175). Springer.https://doi.org/10.1007/978-1-4419-8462-3_6.

Tang, L., Liu, H., & Zhang, J. (2011). Identifying evolving groups in dynamic multimode networks. *Microcomputer & Its Applications*. Applications.

Teng, X., Liu, J., & Li, M. (2019). Overlapping community detection in directed and undirected attributed networks using a multiobjective evolutionary algorithm. *IEEE Transactions on Cybernetics*, 1–13. https://doi.org/10.1109/TCYB.2019.2931983

Wang, Z., Li, Z., Guan, Y., Sun, Y., Rui, X., & Xiang, X. (2018). Tracking the evolution of overlapping communities in dynamic social networks. Knowledge-Based Systems, (p. S0950705118302570).

Xiao, W., & Gutman, I. (2003). Resistance distance and laplacian spectrum. *Theoretical Chemistry Accounts, 110*, 284–289.

Xie, J., & Szymanski, B. K. (2012). Towards linear time overlapping community detection in social networks. In P.-N. Tan, S. Chawla, C. K. Ho, & J. Bailey (Eds.), *Advances in Knowledge Discovery and Data Mining* (pp. 25–36). Berlin, Heidelberg: Springer, Berlin Heidelberg.

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

Zachary, W. W. (1976). An information flow model for conflict and fission in small groups. *Journal of Anthropological Research, 33*, 452–473.

Zhang, L., Pan, H., Su, Y., Zhang, X., & Niu, Y. (2017). *A mixed representation-based multiobjective evolutionary algorithm for overlapping community detection* (pp. 1–14). IEEE Transactions on Cybernetics.

Zhang, X., Zhou, K., Pan, H., Zhang, L., & Jin, Y. (2018). *A network reduction-based multiobjective evolutionary algorithm for community detection in large-scale complex networks* (pp. 1–14). IEEE Transactions on Cybernetics.
Zhang, Y. F., & Chiang, H.-D. (2016). A novel consensus-based particle swarm optimization-assisted trust-tech methodology for large-scale global optimization. *IEEE Transactions on Cybernetics*, 1–13.
Zhou, X., Bo, W., & Jin, Q. (2017). Analysis of user network and correlation for community discovery based on topic-aware similarity and behavioral influence. *IEEE Transactions on Cybernetics*, 1–13.

Wei-min Li is currently a Professor with the School of Computer Engineering and Science, Shanghai University, China. He was a JSPS Research Fellow with the Department of Human Informatics and Cognitive Sciences, Waseda University, Japan, from 2012 to 2013. And he used to be a Visiting Scholar with the Department of Computer Science, University of California at Santa Barbara, supported by the China Scholarship Council, from 2015 to 2016. He has been involved in the extensively research works in the fields of computer science, service computing, business process management, and database technology. His current research interests include social computing, data mining and analytics, group behavior modeling and simulating, and service recommendations.

Heng Zhu is currently pursuing the master degree with the School of Computer Engineering and Science, Shanghai University, China. His research interests cover Social network, business process management and big data.

Shao-hua Li is PHD student in School of Computer Engineering and Science, Shanghai University, Shanghai, China. His research interests include social network analysis, neural language processing and graph structured data representation learning.

Hao Wang is an associate professor in the Department of Computer Science in Norwegian University of Science & Technology, Norway. He has a Ph.D. degree and a B.Eng. degree, both in computer science and engineering, from South China University of Technology. His research interests include big data analytics, industrial internet of things, high performance computing, safety-critical systems, and communication security. He has published 100 + papers in reputable international journals and conferences. He served as a TPC co-chair for IEEE DataCom 2015, IEEE CIT 2017, ES 2017, a senior TPC member for CIKM 2019, and reviewers for journals such as IEEE TKDE, TII, TBD, TETC, T-IFS, IoTJ, TCSS, and ACM TOMM, TIST.

Hong-Ning Dai is an associate professor in Faculty of Information Technology at Macau University of Science and Technology. His research interests include Internet of Things, Big Data Analytics and Blockchains. He has published more than 90 peer-reviewed papers in top-tier journals and conferences. He is a senior member of the Institute of Electrical and Electronics Engineers (IEEE) and a professional member of the Association for Computing Machinery (ACM).

Dr. Can Wang has been a Lecturer of Computer Science at Griffith University, Australia since May, 2017. Prior to this, she was a Postdoctoral Fellow at Data61, CSIRO, Australia from 2014 to 2016. She received her PhD degree in computer science from University of Technology Sydney (UTS), Australia in 2013, Master's degree and Bachelor's degree from Wuhan University, China in 2009 and 2007, respectively.

Qun Jin is Professor of Waseda University and Distinguished Professor and Academic Advisor of the College of Information Engineering, China Jiliang University. He mainly engaged in information systems, behavior and cognitive informatics, big data analysis and processing, information retrieval and Research and applications such as recommendation, human-computer interaction, personal modeling and personalized services. Professor Jin took the lead in proposing the concepts and methods of Personal Analytics and Unified Individual Modeling and the theoretical framework for the integration and individualized sustainable use of personal big data, opening up a sustainable and effective approach to utilize personal big data.