Fast Community Detection Based on Distance Dynamics

Lei Chen
the School of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China.

Jing Zhang
the College of Electrical and Information Engineering, Hunan University, Changsha 410082, China.

Lijun Cai
the College of Information Science and Engineering, Hunan University, Changsha 410082, China.

Ziyun Deng
the Department of Economics and Trade, Changsha Commerce and Tourism College, Changsha 410082, China.

Follow this and additional works at: https://tsinghuauniversitypress.researchcommons.org/tsinghua-science-and-technology

Part of the Computer Sciences Commons, and the Electrical and Computer Engineering Commons

Recommended Citation
Lei Chen, Jing Zhang, Lijun Cai et al. Fast Community Detection Based on Distance Dynamics. Tsinghua Science and Technology 2017, 22(6): 564-585.
Fast Community Detection Based on Distance Dynamics

Lei Chen, Jing Zhang, Lijun Cai*, and Ziyun Deng

Abstract: The distance dynamics model is an excellent tool for uncovering the community structure of a complex network. However, one issue that must be addressed by this model is its very long computation time in large-scale networks. To identify the community structure of a large-scale network with high speed and high quality, in this paper, we propose a fast community detection algorithm, the F-Attractor, which is based on the distance dynamics model. The main contributions of the F-Attractor are as follows. First, we propose the use of two prejudgment rules from two different perspectives: node and edge. Based on these two rules, we develop a strategy of internal edge prejudgment for predicting the internal edges of the network. Internal edge prejudgment can reduce the number of edges and their neighbors that participate in the distance dynamics model. Second, we introduce a triangle distance to further enhance the speed of the interaction process in the distance dynamics model. This triangle distance uses two known distances to measure a third distance without any extra computation. We combine the above techniques to improve the distance dynamics model and then describe the community detection process of the F-Attractor. The results of an extensive series of experiments demonstrate that the F-Attractor offers high-speed community detection and high partition quality.

Key words: community detection; interaction model; complex network; graph clustering; graph mining

1 Introduction

Community detection is a research hotspot with respect to complex networks and its goal is to determine the community structure in complex networks[1–3]. A community in a network is a group of cohesive nodes having dense connections within the group and sparse connections with other groups[4]. Generally, nodes of the same community have the same or similar features due to their strong cohesiveness. Therefore, community detection plays a very important role in the analysis of complex networks. For instance, it can help to identify the internal structure of the network, detect potentially useful information, and mine the relationships between individuals[5].

With the development of information technology, the scale of complex networks is growing larger all the time. Social networks are a typical example. By the end of March 2014, the number of active monthly Facebook users exceeded 1.3 billion, and the active daily users of Sina Weibo (a popular microblog service in China that is similar to Twitter) exceeded 66.6 million[6]. In addition, 30 billion pieces of content and 2.7 billion likes and comments are posted to Facebook every day. Four billion views occur on YouTube every day. Therefore, in a large-scale network environment, a high-speed and high-quality community detection algorithm is critical.

In the past decade, many community detection
algorithms have been proposed and these can be divided into three categories: graph partition methods, modularity-based methods, and dynamic methods. Graph partition methods typically split a network into several sub-groups of a fixed number. The well-known normalized cut method and spectral bisection method\cite{17, 18} are typical examples of algorithms of this category. However, these methods strongly depend on a pre-defined number of communities as part of the input, which is unrealistic in real-world environments. In modularity-based methods, the basic idea is to optimize the modularity criterion\cite{9} for different goals, including a large graph\cite{10, 11}, high accuracy, and low complexity\cite{12, 13}. However, modularity-based methods usually encounter the “resolution limit” problem\cite{14}. That is, these methods cannot detect small communities of a network. The dynamic methods usually introduce a dynamic process to detect the community structure of the network. Using different dynamic strategies, such as label propagation\cite{15, 16}, random walks\cite{17}, and synchronization-inspired approaches\cite{18–20}, these methods can achieve good performance and solve the “resolution limit” problem. However, they usually require long computation times in large networks.

In 2015, inspired by synchronization clustering\cite{18}, a distance dynamics model\cite{21} was proposed for detecting the community structure of a network from a new perspective: the edges. In the network, all edges are classified into two categories: internal edges located within the same community and external edges that make contact with two different communities. The distance dynamics model assumes that multiple potential communities are connected to each other by a few external edges. After removing all external edges, the community structure of the network is naturally exposed. Based on the above idea, we introduce a dynamic interaction process to identify the type of each edge in the network. The process is as follows. First, a unique initial distance is associated with each edge based on the structural similarity of two end nodes. Next, a dynamic interaction process is initiated. As time evolves, influenced by the interaction between neighbors, the distance between internal edges gradually shrinks, while that between external edges gradually expands. Finally, all external edges naturally emerge by virtue of their long distances, thereby revealing the community structure of the network for the purpose of detection. The model has several associated benefits, including “intuitive community detection”, “small community detection”, and “anomaly detection”, which are very important and desirable aspects. We present the model in greater detail in Section 2.

In large network environments, however, the distance dynamics model requires very long computation times, and at time this computation time can be unacceptable. By carefully analyzing the interaction process, we found three reasons to account for this long computation time. (1) The number of edges is very large in a large network. If each edge is involved in the interaction process, the computation time of the distance dynamics model will be very long. (2) The number of neighbors is large for each edge in the network, especially in a high-density network, where each edge typically has hundreds or thousands of neighbors. Because each edge interacts with each neighbor, a long time is required to renew the distance of each edge. (3) In the first time step of the interaction process, we must also calculate the initial distance for each exclusive neighbor of each edge. And for each edge, there are many exclusive neighbors. This further increases the time overhead.

Our goal in this paper is to quickly detect the community structure of a large network using the distance dynamics model. To do so, we improve the speed of community detection via two optimization strategies. (1) We employ a process known as internal edge prejudgment, which significantly reduces the number of edges involved in the distance dynamics model. Here, we posit that an external edge of the network is more important than an internal edge with regard to the distance dynamics model. In addition, it is easy to realize that the number of internal edges is far greater than the number of external edges in a network, based on the intuitive definition of a community. Therefore, we introduce a process of internal edge prejudgment to enhance the computation speed of the distance dynamics model by predicting whether an edge is internal. If an edge is determined to be an internal edge, we assume that it does not participate in the model. If an edge is determined not to be an internal edge, we employ the distance dynamics model to determine the final status of this edge. (2) The second strategy is the acceleration of the interaction process. The goal is to further accelerate the interaction process of the distance dynamics model by reducing the computation time with respect to the initial distance for all exclusive neighbors from each edge. To do so, we introduce the triangle distance to quickly predict
the initial distance between exclusive neighbors of each edge without any extra computation. In summary, based on the distance dynamics model, we propose a fast community detection algorithm for large-scale networks, which we call the F-Attractor. The significant contributions of this paper can be summarized as follows:

- We develop two prejudgment rules for predicting the internal edges of a network and to reduce both the number of edges taking part in the distance dynamics model and the number of neighbors of each edge.
- We introduce the triangle distance concept for measuring the initial distance of each exclusive neighbor from two known distances without any extra calculation. The triangle distance further speeds up the process of community detection in the distance dynamics model.
- We propose a fast community detection algorithm, the F-Attractor, that quickly uncovers the community structure of a large-scale network using the distance dynamics model.

The remainder of this paper is organized as follows. We describe the traditional distance dynamics model in Section 2. In Section 3, we present our fast community detection algorithm, the F-Attractor. We present the results of our series of extensive experiments in Section 4. Lastly, in Section 5, we draw our conclusions.

## 2 Traditional Distance Dynamics Model

The distance dynamics model, which was proposed in 2015, is a typical dynamic model for community detection[21]. The model process is as follows: in the beginning, each edge is associated with an initial weight of the corresponding edge $e(u,v)$. Each edge $e(u,v)\in E$ implies a communication connection between nodes $u$ and $v$. $w(u,v)$ is the weight of the corresponding edge $e(u,v)$.

### 2.1 Related background

**Definition 1 (Undirected graph).** Let $G=(V,E,W)$ be an undirected graph where $V$ is the nodes set, $E$ is the edges set, and $W$ is the corresponding weight set of all edges. Each edge $e(u,v)\in E$ implies a communication connection between nodes $u$ and $v$. $w(u,v)$ is the weight of the corresponding edge $e(u,v)$.

**Definition 2 (Neighbors of node $u$).** Given an undirected graph $G=(V,E,W)$, the neighbors of node $u$, $N(u)$, comprise a node set that consists of node $u$ and its connected nodes, which is defined as follows:

$$N(u) = \{ v \in V \mid (u,v) \in E \} \cup \{ u \}$$

**Definition 3 (Jaccard distance).** Given an undirected graph $G=(V,E,W)$, the Jaccard distance[25] between node $u$ and node $v$ is defined as follows:

$$d(u,v) = 1 - \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|}$$

In the above equation, $|*|$ indicates the object number of set * and $N(u)$ represents the neighbors of node $u$. The Jaccard distance measures the similarity of the neighbors in nodes $u$ and $v$. The more common neighbors there are for these two nodes, the greater is the similarity of the two nodes and the lower is their Jaccard distance, and vice versa.

In a weighted undirected graph, because each edge has a different weight, the model for computing the Jaccard distance is different and must be extended as follows:

$$d(u,v) = 1 - \frac{\sum_{x \in N(u) \cap N(v)} (w(u,x) + w(v,x))}{\sum_{(x,y) \in E : x,y \in N(u) \cup N(v)} w(x,y)}$$

### 2.2 Interaction model

In the distance dynamics model, three interaction patterns, as shown in Fig. 1, have been proposed to simulate the dynamics of each distance.

**Pattern 1: Influence from directly linked nodes.** The distance $d(u,v)$ between nodes $u$ and $v$ is obviously influenced by the two directly linked nodes $u$ and $v$. Through mutual interactions, one node attracts the other to move toward it, thereby resulting in the decrease of $d(u,v)$ (see Fig. 1b). To characterize the change in $d(u,v)$, $DI$ is defined to indicate the influence of two directly linked nodes, as follows:

$$DI = -\frac{f(u,v)}{\text{deg}(u)} - \frac{f(u,v)}{\text{deg}(v)}$$

In the pattern DI, $\text{deg}(u)$ indicates the degree of node $u$ and $f(\cdot)$ is a coupling function. The term $1 - d(u,v)$ implies the similarity of the characteristic structure or properties in nodes $u$ and $v$. The term $1/\text{deg}(u)$ is a normalized factor used to consider the different influences between linked nodes of diverse degrees.

**Pattern 2: Influence from common neighbors.** The distance $d(u,v)$ is influenced by the common neighbors...
CN = (N(u) − u) ∩ (N(v) − v) of nodes u and v. Here, each common neighbor is a node connected to both nodes u and v. In the dynamic interaction process, because each common neighbor connects with nodes u and v, the two nodes are attracted and gradually move toward each other, thus leading to a decrease in d(u, v) (see Fig. 1c). The second interaction pattern is called CI, and is defined as follows:

\[
CI = \sum_{x \in CN} \left( \frac{f(1 - d(x, u)) \cdot (1 - d(x, v))}{\deg(u)} \right) - \sum_{x \in CN} \left( \frac{f(1 - d(x, v)) \cdot (1 - d(x, u))}{\deg(v)} \right)
\]

(5)

Here, we use the two terms (1 − d(x, u)) and (1 − d(x, v)) for each common neighbor x to further measure the difference in the influence of patterns DI and CI.

**Pattern 3: Influence from exclusive neighbors.**

The influence of exclusive neighbors is the third interaction pattern. Each exclusive neighbor EN(u) = N(u) − (N(u) ∩ N(v)) or EN(v) = N(v) − (N(u) ∩ N(v)) connects to only one node u or v (see Fig. 1d). In the dynamic interaction process, each exclusive neighbor attracts only one node (node u or v) to move toward it. However, we do not know whether there is another node close to this exclusive neighbor, and thus we cannot obtain the influence of this exclusive neighbor. To determine the positive or negative influence of an exclusive neighbor on this distance, a cohesion parameter \( \lambda \) is introduced to measure whether the exclusive neighbor of node u (or node v) is similar to node v (or node u). This measurement function is defined as follows:

\[
\rho(x, u) = \begin{cases} (1 - d(x, v)) & 1 - d(x, v) \geq \lambda; \\ (1 - d(x, v)) - \lambda, & \text{otherwise} \end{cases}
\]

(6)

Here, \( \rho(x, u) \) indicates the positive or negative influence from the exclusive neighbor x in EN(u) on \( d(u, v) \). When the similarity between the exclusive neighbor x and node v is more than or equal to the cohesion parameter \( \lambda \), \( 1 - d(x, v) \geq \lambda \), we posit that the exclusive neighbor x of node u is similar to node v and that the exclusive neighbor x yields a positive influence on \( d(u, v) \), which leads to a decrease in \( d(u, v) \). In contrast, when the similarity between the exclusive neighbor x and node v is less than \( \lambda \), \( 1 - d(x, v) < \lambda \), the exclusive neighbor x of node u is dissimilar to node v and the exclusive neighbor x yields a negative influence on \( d(u, v) \), thereby leading to an increase in \( d(u, v) \).

Based on the cohesion parameter \( \lambda \), the third interaction pattern is called EI, and is defined as follows:

\[
EI = -\sum_{x \in EN(u)} \left( \frac{f(1 - d(x, u)) \cdot \rho(x, u)}{\deg(u)} \right) - \sum_{y \in EN(v)} \left( \frac{f(1 - d(y, v)) \cdot \rho(y, v)}{\deg(v)} \right)
\]

(7)

Finally, by considering the three interaction patterns together, the dynamics of the distance \( d(u, v) \) between nodes u and v over time is governed by

\[
d(u, v, t + 1) = d(u, v, t) + DI(t) + CI(t) + EI(t)
\]

(8)

where \( d(u, v, t + 1) \) is the renewed distance at time step \( t + 1 \). DI(t), CI(t), and EI(t) are the three different influences of the directly linked nodes, common neighbors, and exclusive neighbors, respectively.

3 F-Attractor: Fast Community Detection Based on Distance Dynamics

In this section, we present a fast community detection algorithm, the F-Attractor, which is based on the distance dynamics model. Our proposed algorithm mainly has two optimization strategies for improving the speed of community detection in a large network.

3.1 Internal edge prejudgment

3.1.1 Problem analysis

To detect the community structure of a network, the
distance dynamics model must identify all the external edges. To do so, it uses a dynamic interaction process to simulate the distance dynamics of each edge and determine the final status of each edge. However, the number of edges is very large in a large network, so a long computation time is required to determine the community structure of the network. To accelerate the work of the distance dynamics model, our first optimization strategy is to reduce the number of edges that participate in the model’s dynamic interaction process.

In accordance with the basic definition of a community, the number of internal edges is far larger than the number of external edges in a network. This is because the connections within a community are dense, whereas connections between communities are sparse. Moreover, by deleting all the external edges, the community structure of the network can be detected using the distance dynamics model. Therefore, in the distance dynamics model, the external edges are more important than the internal edges. Considering the above two features in a large network, we attempt to quickly exclude various internal edges to reduce the number of edges participating in the distance dynamics model. To do so, we attempt to quickly prejudge the type of each edge and decide whether or not an edge is internal. If we predict an edge be internal, this edge does not participate in the interaction process of the distance dynamics model. If we predict the opposite, this edge must take part in the interaction process. Generally, the number of internal edges we prejudge is greater and the number of edges participating in the interaction process is lower, so the speed of the distance dynamics model is faster.

To show the above idea more clearly, here, we consider a simple example, as shown in Fig. 2. In the figure, a social network contains four classes with different colors, 76 nodes, and 388 edges. Of all the edges, 373 are internal, representing the largest proportion, as shown by the gray solid lines. The external edges number only 15, as shown by the thick blue dashed lines. If we delete the 15 external edges, the community structure of the social network will be naturally exposed. However, in the distance dynamics model, all edges must take part in the interaction process to identify the 15 external edges. Therefore, we make prejudgments each edge, and exclude various internal edges, thereby reducing the number of edges taking part in the interaction process and enhancing the speed of community detection.

To more accurately prejudge the internal edges of a network, we propose two prejudgment rules: a node-based and an edge-based prejudgment rules.

3.1.2 Node-based prejudgment

If an edge is internal, the two end nodes of this edge must belong to the same community. Therefore, we make a prejudgment regarding whether an edge is internal based on the node perspective, which we refer to as the node-based prejudgment rule.

**Node-based prejudgment rule.** If the neighbor set of node \(v \ (N(v))\) is a subset of the neighbor set of node \(u \ (N(u))\) and the number of entries in \(N(v)\) is more than half the number of entries in \(N(u)\), then we consider node \(v\) to be strongly dependent on node \(u\), and that node \(v\) belongs to the same community as node \(u\) \((N(v) \subseteq N(u) \text{ and } |N(v)|/|N(u)| > 1/2)\).

**Proof** This node-based prejudgment rule implies the following: (1) \(N(v) \subseteq N(u)\) indicates that node \(v\) directly connects with node \(u\), i.e., that there is an edge \(e(v, u)\). (2) Because \(N(v) \subseteq N(u)\), the common neighbor set of \(e(v, u)\) is \(N(v)\), and the exclusive neighbor set is \(N(u) - N(v)\). (3) From \(N(v) \subseteq N(u)\), we also know that node \(v\) has no exclusive nodes. (4) Due to the fact that \(N(v)/N(u) > 1/2\), the number of common neighbors is more than the number of exclusive neighbors in node \(u\). According to the three interaction patterns of the distance dynamics model, when proceeding through the interaction process, the distance \(d(v, u)\) of \(e(v, u)\) will gradually approach...
When two nodes satisfy the above rule, we consider them to belong to the same community, and the edge between the two nodes is treated as an internal edge. To present this rule more clearly, let us take a simple social network as an example, as shown in Fig. 3. In the figure, the social network contains 13 nodes and 19 edges. If we focus on the green edge \( e(u, v) \), the neighbor set \( N(v) \) of node \( v \) is a subset of the neighbor set \( N(u) \) of node \( u \). The number of nodes in \( N(v) \) is three, and the number of nodes in \( N(u) \) is five, so the number of entries in \( N(v) \) exceeds half the number of entries in \( N(u) \). Therefore, we determine that node \( v \) is strongly dependent on node \( u \), that nodes \( v \) and \( u \) belong to the same community, and that edge \( e(u, v) \) is internal, as shown in Fig. 3b.

### 3.1.3 Edge-based prejudgment

The node-based prejudgment rule is very strict and only works for a few edges. To predict more internal edges, we must find new ways to take the edge perspective. An internal edge implies that two end nodes are located in the same community. That is, the clustering coefficient of this edge is usually large. Therefore, we introduce a revised edge-clustering coefficient as a metric for measuring whether an edge is internal.

**1) Revised edge-clustering coefficient.** The concept of an edge-clustering coefficient originated from research on community detection in complex networks\(^{[22]}\). This coefficient characterizes the closeness between two end nodes of an edge and the other nodes around them. Edges with higher clustering coefficients tend to be included in the communities of the network, as proven in many previous works\(^{[23, 24]}\). Hence, the edge-clustering coefficient is a measure that can both evaluate the importance of edges and describe the relative proximity of two end nodes. In this paper, we demonstrate the concept of the edge-clustering coefficient and propose a revised edge-clustering coefficient.

**Definition 4 (Edge-clustering coefficient).** For an edge \( e(u, v) \), we consider how many neighbor nodes adjoin nodes \( u \) and \( v \). The edge-clustering coefficient of \( e(u, v) \) can be defined as follows:

\[
\text{ECC}(u, v) = \frac{z_{u,v}}{\min(\deg(u) - 1, \deg(v) - 1)}
\]

where \( z_{u,v} \) denotes the real number of triangles including nodes \( u \) and \( v \), \( \deg(u) \) and \( \deg(v) \) are the degrees of nodes \( u \) and \( v \), respectively, and \( \min(\deg(u) - 1, \deg(v) - 1) \) is the maximum number of triangles including nodes \( u \) and \( v \). Figure 4a shows the edge-clustering coefficient. Let us take the green edge \( e(6, 8) \) as an example. The two end nodes are \( n_6 \) and \( n_8 \), and their degrees are four and three, respectively. In theory, edge \( e(6, 8) \) can constitute \( \min(4 - 1, 3 - 1) = 2 \) triangles at most. However, there are actually two real triangles, \( \Delta 678 \) and \( \Delta 698 \). Therefore, \( \text{ECC}(6, 8) = \frac{2}{2} = 1.0 \).

We select the edge-clustering coefficient for two reasons. (1) It is a local metric and its computation is very fast. (2) In the distance dynamics model, the Jaccard distance is used to calculate the initial distance of each edge. Since the Jaccard distance is very similar to the edge-clustering coefficient in that they have some common computational content, this can save further computation time.

However, the traditional edge-clustering coefficient ECC does not consider the difference of degree between the two end nodes. This results in the inability to accurately predict the closeness of two nodes, as well as the inability to accurately prejudge whether two nodes belong to the same community. Consider the green edge \( e(0, 6) \) in Fig. 4a. The edge-clustering coefficient is \( \text{ECC}(0, 6) = \frac{2}{3} = 0.67 \), which is very large. However, the edge \( e(0, 6) \) is external. To solve this problem and make a more accurate prediction, we introduce a revised edge-clustering coefficient.

**Definition 5 (Revised edge-clustering coefficient).**
For one edge \(e(u, v)\), our revised edge-clustering coefficient not only presents how many neighbors adjoin nodes \(u\) and \(v\), but also considers the difference of degree between nodes \(u\) and \(v\). It is defined as follows:

\[
\text{DECC}(u, v) = \frac{z_{u,v}}{\text{deg}(u) - 1} \times \frac{z_{u,v}}{\text{deg}(v) - 1}
\]

(10)

Figure 4b shows the revised edge-clustering coefficient. To compare the traditional and our revised edge-clustering coefficients, let us focus on the two green edges \(e(6, 8)\) and \(e(0, 6)\). For the internal edge \(e(6, 8)\), the traditional edge-clustering coefficient ECC is 0.67 and the revised edge-clustering coefficient DECC is 0.13. Compared to the traditional edge-clustering coefficient ECC, DECC is smaller for external edges.

Therefore, using the revised edge-clustering coefficient DECC, we can more accurately predict the proximity of the two nodes.

(2) Edge-based prejudgment.

Based on the revised edge-clustering coefficient, we propose the second prejudgment rule.

Edge-based prejudgment rule. The \(\text{DECC}(u, v)\) of edge \(e(u, v)\) is larger, so the proximity of nodes \(u\) and \(v\) is higher and the probability of edge \(e(u, v)\) being internal is also higher.

Proof The fact that \(\text{DECC}(u, v)\) is larger leads to the following three conclusions. (1) The difference of degree between nodes \(u\) and \(v\) is very small. (2) For the edge \(e(u, v)\), common neighbors make up the greatest proportion. (3) The exclusive neighbors of \(e(u, v)\) are fewer in number. Therefore, according to the three interaction patterns of the distance dynamics model, the distance \(d(u, v)\) of edge \(e(u, v)\) will ultimately become zero. This is because the positive influence of nodes \(u\) and \(v\) and common neighbors is much larger than the negative influence from exclusive neighbors. That is, the edge \(e(u, v)\) is internal.

To quantify the above rule, we introduce a prejudgment function for determining whether an edge is internal, which we have formulated as follows:

\[
\text{Pre}(x, y) = \begin{cases} 
    1, & \text{DECC}(x, y) \geq \varepsilon; \\
    0, & \text{DECC}(x, y) < \varepsilon
\end{cases}
\]

(11)

where the term \(\varepsilon\) is a user-defined threshold parameter. With respect to the edge \(e(x, y)\), when \(\text{DECC}(x, y)\) is greater than or equal to \(\varepsilon\), the edge \(e(x, y)\) is considered to internal. In contrast, when \(\text{DECC}(x, y)\) is less than \(\varepsilon\), we cannot determine the type of \(e(x, y)\), i.e., whether it is an internal or external edge, so we refer to this edge as fuzzy. To obtain the final status, each fuzzy edge \(e(x, y)\) must take part in the interaction process of the distance dynamics model. Generally, with higher values of \(\varepsilon\), an edge has a smaller probability of being internal, and the number of internal edges we prejudge is smaller. By modulating the parameter \(\varepsilon\), it is easy to adjust the strength of prejudgment and the number of internal edges prejudged.

3.1.4 A simplified validation

To validate the effectiveness of the two prejudgment rules, we selected three well-known real-world networks with ground truth, used the F-Attractor algorithm to detect the community structure, and considered the prejudged internal edges. These three real-world networks are publicly available from the UCI network data repository (https://network data.ics.uci.edu/index.php).

The first network is Zachary’s karate club network, which consists of 34 vertices and 78 undirected edges. Each node represents a member of the club and each edge represents a link between two members. Figure 5a shows the ground truth of the karate club network, which consists of three classes. Figure 5b shows the detection result, which contains two classes and one outlier partitioned by the thick red dashed line. Figure 5c shows all the prejudged internal edges determined based on our two prejudgment rules. From the figure, we can see that 53 internal edges were prejudged and that each internal edge is completely accurate.

The second network comprises the books about US politics, which consists of 105 nodes and 441 edges. This network is derived from books about US politics published around the time of the 2004 presidential election and sold by the online bookseller “Amazon.com”. Figure 6a shows the ground truth of the network, which contains three classes. Figure 6b presents the detection results of the F-Attractor algorithm, which found two classes and one outlier (node 28), as partitioned by the thick red dashed line. Figure 6c shows a plot of 318 prejudged internal edges. This network is derived from books about US politics published around the time of the 2004 presidential election and sold by the online bookseller “Amazon.com”. Figure 6a shows the ground truth of the karate club network, which consists of three classes. Figure 6b presents the detection results of the F-Attractor algorithm, which found two classes and one outlier (node 28), as partitioned by the thick red dashed line. Figure 6c shows a plot of 318 prejudged internal edges. This network is derived from books about US politics published around the time of the 2004 presidential election and sold by the online bookseller “Amazon.com”. Figure 6a shows the ground truth of the network, which contains three classes. Figure 6b presents the detection results of the F-Attractor algorithm, which found two classes and one outlier (node 28), as partitioned by the thick red dashed line. Figure 6c shows a plot of 318 prejudged internal edges. This network is derived from books about US politics published around the time of the 2004 presidential election and sold by the online bookseller “Amazon.com”. Figure 6a shows the ground truth of the network, which contains three classes. Figure 6b presents the detection results of the F-Attractor algorithm, which found two classes and one outlier (node 28), as partitioned by the thick red dashed line. Figure 6c shows a plot of 318 prejudged internal edges. This network is derived from books about US politics published around the time of the 2004 presidential election and sold by the online bookseller “Amazon.com”. Figure 6a shows the ground truth of the network, which contains three classes. Figure 6b presents the detection results of the F-Attractor algorithm, which found two classes and one outlier (node 28), as partitioned by the thick red dashed line. Figure 6c shows a plot of 318 prejudged internal edges. This network is derived from books about US politics published around the time of the 2004 presidential election and sold by the online bookseller “Amazon.com".
network, which is derived from the schedule of American Division I football games for the regular season of fall 2000. This network contains 115 vertices, each of which represents a team, and each edge represents regular-season games between two teams. This network is considered to be a high-density network because each node has a large degree. This network covers 12 classes with different colors, as shown in Fig. 7a. Using the F-Attractor algorithm, we detected and accurately partitioned 12 communities, as shown by the thick red dashed lines, as shown in Fig. 7b. In Fig. 7c, we plotted 390 prejudged internal edges, and each edge we prejudged is completely accurate. Since the total number of edges in the network is 613, these prejudged edges comprise the largest proportion of all the edges.

Based on the above three figures, we can make the following remarks. (1) Our prejudgment rules are very accurate when applied to different networks, whether they are normal or dense networks. (2) The F-Attractor algorithm can accurately uncover the community structures of different networks.

### 3.2 Acceleration of the dynamic interaction process

In the distance dynamics model, the dynamic interaction process simulates the distance dynamics of each edge. However, for a large network, the dynamic interaction process requires a long computation time to determine the status of each edge. In this section, we describe how we accelerate the dynamic interaction process.
3.2.1 Problem analysis

In the distance dynamics model, the neighbors of each edge are classified into three categories as two directly linked nodes, common neighbors, or exclusive neighbors, as shown in Fig. 1. In the dynamic interaction process, each edge interacts with each neighbor, and the distance of this edge is influenced by three different neighbors. The degree of influence of one neighbor is determined by the distances between this neighbor and two end nodes. To do so, at the beginning of the dynamic interaction process, each real edge in the network is associated with an initial distance. Generally, each neighbor connects to both end nodes of one edge. However, exclusive neighbors are exceptions. Each exclusive neighbor directly links with only to one end node. This results in another initial distance between the exclusive neighbor and the indirectly linked end node, which is not calculated. Hence, in the first time step of the interaction process, we must also calculate the initial distances between exclusive and indirectly linked end nodes. Moreover, the number of exclusive neighbors is far larger than the number of real edges in a network, as shown in Table 1, in which we can see that the number of exclusive neighbors is usually several times larger than the number of real edges in the network. In summary, in a large network environment, there are two main reasons for the long computation time of the interaction process: (1) the number of neighbors is large for each edge, and is very large in a high-density network; (2) the time overhead of additional calculation is very large for the initial distances of exclusive neighbors. To enhance the speed of the interaction process, we must also calculate the initial distances between exclusive and indirectly linked end nodes.

Table 1 Number of exclusive neighbors in three real-world networks.

| Network     | Nodes | Edges | Exclusive neighbors |
|-------------|-------|-------|---------------------|
| Karate club | 34    | 78    | 265                 |
| Books       | 105   | 44    | 1561                |
| Football    | 115   | 613   | 2306                |

Table 1 shows that the number of exclusive neighbors in real-world networks is significantly larger than the number of real edges, which seriously affects the speed of the interaction process. This characteristic of the distance of an exclusive neighbor is inconsistent with the nature of the distance dynamics model. Specifically, in the third interaction pattern EI, the influence \( \rho(x, u) \) of the exclusive neighbor \( x \) on the distance \( d(u, v) \) is unchanging over the entire dynamic interaction process, as shown in Eq. (6). Therefore, we find that the dynamics of each distance are not adequately simulated in the traditional distance dynamics model.

To simplify the description of an exclusive neighbor and its indirectly linked end node, we introduce the definition of an exclusive neighbor pair.

**Definition 6 (Exclusive neighbor pair).** If node \( x \) is an exclusive neighbor of edge \( e(u, v) \) and does not link to the end node \( v \), then we call the pair \( (x, v) \) an exclusive neighbor pair of edge \( e(u, v) \).

The term exclusive neighbor pair indicates that the edge between nodes \( x \) and \( v \) is a virtual edge, as shown by the red dotted line in Fig. 8a.

In summary, in a large network environment, there are two main reasons for the long computation time of the interaction process: (1) the number of neighbors is large for each edge, and is very large in a high-density network; (2) the time overhead of additional calculation is very large for the initial distances of exclusive neighbors. To enhance the speed of the interaction process, we must also calculate the initial distances between exclusive and indirectly linked end nodes.
process, we must optimize the above two deficiencies. First, by undertaking the process of internal edge prejudgment, we exclude many internal edges, thereby significantly reducing the number of edges taking part in the dynamic interaction process. Moreover, after making the internal edge prejudgment, the number of neighbors for each edge participating in the interaction process is also reduced because a given neighbor does not participate in the dynamic interaction process if the edge between this neighbor and any end node is prejudged to be internal. Second, we introduce the concept of triangle distance to measure the dynamic distance of an exclusive neighbor pair without any extra computation. This further enhances the speed of the dynamic interaction process and overcomes the problem whereby “the distance of an exclusive neighbor pair is forever unchanged”. In the next section, we present more details regarding the triangle distance.

3.2.2 Triangle distance

By the careful analysis of an exclusive neighbor pair, we find that each exclusive neighbor and two end nodes can compose a triangle, in which there are two real edges and a virtual edge, as shown in Fig. 8a. In the figure, neighbors of edge $e(u, v)$ are illustrated, for which all neighbors except the two end nodes $u$ and $v$ are exclusive neighbors. Let us take neighbor $x$ as an example, in which nodes $x$, $u$, and $v$ compose a triangle $\triangle xuv$, edges $e(u, v)$ and $e(x, u)$ are two real edges, and edge $e(x, v)$ is a virtual edge. We use the exclusive neighbor pair $\langle x, v \rangle$ to represent the virtual edge $e(x, v)$. To reduce the calculation time for the initial distance of the exclusive neighbor pair $\langle x, v \rangle$, we use two known distances, $d(x, u)$ and $d(u, v)$, to estimate or bound the distance $d(x, v)$. This enhances the speed of the dynamic interaction process. Therefore, we introduce the concept and definition of triangle distance. In Fig. 8b, which shows the concept of triangle distance, we know the value of two distances, $d(u, v)$ and $d(x, u)$, and assume that $d(u, v) \geq d(x, u)$. Then, we fix nodes $u$ and $v$ and plot a green circle with $d(x, u)$ as the radius. From the plot, we can see that node $x$ can be located anywhere in the green circle because $d(x, u)$ is fixed. That is, with the movement of node $x$ on the green circle, the distance $d(x, v)$ changes gradually, as shown by the red dotted line. When node $x$ is at point $x_{\text{max}}$, $d(x, v)$ is the largest. In contrast, when node $x$ is at point $x_{\text{min}}$, $d(x, v)$ is the smallest. Therefore, we use the average value of the largest and smallest distances to formulate the triangle distance.

**Definition 7 (Triangle distance).** The triangle distance is employed to quickly measure the distance of an exclusive neighbor pair. It is defined as follows:

$$d_{\Delta}(x, v) = \frac{\|d(x, u) - d(u, v)\|}{2} + \frac{d(x, u) + d(u, v)}{2}$$

where $\langle x, v \rangle$ is an exclusive neighbor pair of edge $e(u, v)$. Using the triangle distance, we can accurately estimate the dynamic distance of an exclusive neighbor pair without any extra computation. This is very advantageous and can enhance the speed of the dynamic interaction process in a large network.

3.2.3 Validation of effectiveness

In this section, we validate the effectiveness of the triangle distance from both theoretical and experimental perspectives.

1. **Theoretical verification.** In the distance dynamics model, the Jaccard distance (Soergel distance)[25] is used as the metric for measuring the distance of an edge, as shown in Eqs. (2) or (3). Moreover, the Jaccard distance has the following properties[26]:

- The distance value must be zero or positive, and the distance from an object to itself must be zero:
  $$d(x, y) \geq 0, \quad d(x, x) = d(y, y) = 0.$$

- The distance value must be symmetric:
  $$d(x, y) = d(y, x).$$

- The distance between non-identical objects must be greater than zero:
  $$x \neq y \Rightarrow d(x, y) > 0.$$

- The distance value must obey triangular inequality:
  $$\|d(x, z) - d(y, z)\| \leq d(x, y) \leq d(x, z) + d(y, z),$$
  where $\| \cdot \|$ is the absolute value function.

In the above four properties, triangular inequality is very important and has been proven repeatedly...
In the traditional distance dynamics model, three interaction patterns (DI, CI, and EI) are identified to reveal the community structure of the network, as presented in detail in Section 2 above. In contrast to the traditional model, our improved model introduces the triangle distance to replace the Jaccard distance for measuring the distance of an exclusive neighbor pair. The triangle distance has two advantages. (1) It can accelerate the interaction process of the improved model because it is very fast and requires no extra calculation. (2) It is a dynamic distance for exclusive neighbor pairs. Along with the dynamic changes of two known distances, the triangle distance can dynamically change during the interaction process. This overcomes the shortage whereby “the distance of an exclusive neighbor pair is forever unchanged” in the traditional model. Because the traditional DI and CI patterns are independent of exclusive neighbor pairs, we continue to use these two patterns in our improved model, and simply update the EI pattern by our use of the triangle distance.

New pattern 3: Influence from exclusive neighbors. The distance \( d(u, v) \) is also influenced by the exclusive neighbors \( EN(u) = N(u) - (N(u) \cap N(v)) \) or \( EN(v) = N(v) - (N(u) \cap N(v)) \). Each exclusive neighbor links with only one end node (node \( u \) or node \( v \)), as shown in Fig. 1d. Through mutual interaction, each exclusive neighbor attracts only one end node (node \( u \) or node \( v \)) to move toward it. However, we do not know the direction of movement of the other end node, so we do not measure the influence of the exclusive neighbor on the distance \( d(u, v) \). In our improved model, we use the cohesion parameter \( \lambda \) to determine the direction of influence of the exclusive neighbor and use the triangle distance to measure the strength of influence of the exclusive neighbor. Hence,

| Table 2  | Test of the triangle distance. |
|----------|-------------------------------|
|          | Karate club                  | Books           | Football        |
|          | Exclusive neighbor pair | Jaccard distance | Triangle distance | Exclusive neighbor pair | Jaccard distance | Triangle distance |
| (23→28) | 0.86 | 0.85 | (6→41) | 0.95 | 0.88 | (33→63) | 0.95 | 0.9 |
| (30→32) | 0.80 | 0.82 | (60→100) | 0.75 | 0.75 | (0→55) | 0.95 | 0.91 |
| (28→30) | 0.75 | 0.74 | (0→24) | 0.93 | 0.85 | (42→94) | 0.88 | 0.88 |
| (13→32) | 0.89 | 0.90 | (78→85) | 0.90 | 0.88 | (5→69) | 0.86 | 0.82 |
| (1→33)  | 0.88 | 0.85 | (32→42) | 0.81 | 0.78 | (21→39) | 0.95 | 0.91 |
| (9→14)  | 0.66 | 0.67 | (20→50) | 0.84 | 0.83 | (86→96) | 0.90 | 0.90 |
| (6→12)  | 0.83 | 0.78 | (1→25) | 0.77 | 0.80 | (72→84) | 0.66 | 0.53 |
| (24→25) | 0.75 | 0.75 | (36→39) | 0.75 | 0.72 | (46→77) | 0.86 | 0.86 |
| (10→29) | 0.60 | 0.52 | (24→41) | 0.64 | 0.53 | (30→80) | 0.8 | 0.73 |
| (16→21) | 0.50 | 0.48 | (8→82) | 0.54 | 0.49 | (76→112) | 0.85 | 0.84 |

by different approaches\cite{27, 28}. Triangular inequality enables the determination of the minimum and maximum distances of two nodes \( x \) and \( y \) without the need for calculation if we know their pairwise distances \( d(x, z) \) and \( d(y, z) \) to a third node \( z \).

If we compare triangular inequality with our triangle distance, it is not difficult to see that they are exactly the same. The basic idea of both is to use two known distances to measure a third distance. Therefore, the effectiveness of the triangle distance is proved and the detailed proof can be found in referenced works\cite{27, 28}.

(2) Simple experimental verification. To further validate the effectiveness of using the triangle distance, we randomly selected 30 exclusive neighbor pairs from three real-world networks (Karate club, Books, and Football), applied the Jaccard distance and triangle distance functions to calculate the distances of the exclusive neighbor pairs, and observed the disparities between the two.

Table 2 lists two distances of 30 exclusive neighbor pairs, in which we can see that the value of the triangle distance is less than the value of the Jaccard distance for most of the exclusive neighbor pairs. For the exclusive neighbor pairs (24→25) and (86→96), the value of the triangle distance is very close to that of the Jaccard distance. Based on our observations, the value of the triangle distance is equal to the value of the Jaccard distance between the two.

Table 2 Test of the triangle distance.
the influence of the exclusive neighbor is defined as follows:
\[
\sigma(x, u) = \begin{cases} 
1 - d_A(x, v), & 1 - d_A(x, v) \geq \lambda; \\
1 - d_A(x, v) - \lambda, & \text{otherwise}
\end{cases}
\] (13)

In the above equation, \(\sigma(x, u)\) indicates the influence from the exclusive neighbor \(x\) on the distance \(d(u, v)\), and \(d_A(x, v)\) represents the triangle distance of the exclusive neighbor pair \(x, v\). When the similarity between the two nodes of the exclusive neighbor pair \(x, v\) is more than or equal to the cohesion parameter \(\lambda\), \(1 - d_A(x, v) \geq \lambda\), then the exclusive neighbor \(x\) yields a positive influence on \(d(u, v)\) and decreases the value of \(d(u, v)\). In contrast, when the similarity between the two nodes of the exclusive neighbor pair \(x, v\) is less than \(\lambda\), \(1 - d_A(x, v) < \lambda\), then the exclusive neighbor \(x\) yields a negative influence on \(d(u, v)\) and increases the value of \(d(u, v)\).

Therefore, we use NEI to represent the new pattern of the exclusive neighbor, which is defined as follows.
\[
\text{NEI} = - \sum_{x \in \text{EN}(u)} \left( \frac{f(1 - d(x, u)) \cdot \sigma(x, u)}{\deg(u)} \right) - \sum_{y \in \text{EN}(v)} \left( \frac{f(1 - d(y, v)) \cdot \sigma(y, v)}{\deg(v)} \right)
\] (14)

Lastly, by considering the three interaction patterns together, the dynamics of the distance \(d(u, v)\) over time is governed by the following:
\[
d(u, v, t + 1) = d(u, v, t) + \text{DI}(t) + \text{CI}(t) + \text{NEI}(t)
\] (15)

where \(d(u, v, t + 1)\) is the renewed distance at time step \(t + 1\). \(\text{DI}(t)\), \(\text{CI}(t)\), and \(\text{NEI}(t)\) correspond to the influence from two directly linked nodes, common neighbors, and exclusive neighbors, respectively.

### 3.3.2 F-Attractor algorithm

Here, we present a comprehensive overview of the F-Attractor algorithm, which is very simple and contains three main sequential phases.

1. **Internal edge prejudgment phase.** The aim of this phase is to prejudge the internal edges of the network and reduce the number of fuzzy edges taking part in the distance dynamics model. In this process, we can scan each edge of the network and calculate the corresponding revised edge-cluster coefficient DECC of each edge. If DECC is more than or equal to parameter \(\varepsilon\), we consider this edge to be an internal edge and set its final distance to 0. If the contrary is true, we consider this edge to be fuzzy such that it must take part in the interaction process to yield its final distance. Moreover, we use the Jaccard distance function to calculate the initial distance for each fuzzy edge in preparation for the second phase.

2. **Acceleration of the dynamic interaction process phase.** The goal of the second phase is to calculate the final distances of all the fuzzy edges identified in the first phase using the improved distance dynamics model. To do so, we begin a dynamic interaction process for all fuzzy edges. In the process of dynamic interaction, we scan each fuzzy edge and calculate the influence \(\text{DI}\) from two directly linked nodes, the influence \(\text{CI}\) from common neighbors, and the influence \(\text{NEI}\) from exclusive neighbors. When we calculate the influence \(\text{NEI}\) from exclusive neighbors, we use the triangle distance to update the last distance of the exclusive neighbor pairs. Our goal is to ensure we identify the dynamics of influence from the exclusive neighbors. Over time, based on the \(\text{DI}\), \(\text{CI}\), and \(\text{NEI}\) interaction patterns, the distance of each fuzzy edge gradually decreases or increases. More specifically, the nodes with higher similarity synchronize faster and the distances between nodes decrease faster. Nodes with higher dissimilarity separate faster and the distances between nodes increase faster. Over multiple time steps, the distances of all fuzzy edges converge to either 0 or 1. When the dynamic interaction process is complete, we move to the third phase.

3. **Community detection phase.** In this phase, we merge the results of the internal edge prejudgment and dynamic interaction process and collect edges with a distance of 1 as the set of external edges of the network. By deleting all of the external edges, we can naturally detect the community structure of the large network. More details regarding community detection can be found in Ref. [21].

### 3.3.3 Time complexity

The time complexity of the F-Attractor has two aspects. (1) To reduce the number of fuzzy edges participating in the distance dynamics model, internal edge prejudgment is made for each edge of the network. The time complexity for this process is denoted as \(O(|E|)\), where \(|E|\) is the number of edges. (2) We begin the dynamic interaction process for all fuzzy edges. Over \(T\) time steps, all the distances of the fuzzy edges converge and the interaction process is then complete. The time complexity for this process is denoted as \(O(T \cdot s \cdot |\text{FE}|)\), where \(|\text{FE}|\) is the number of fuzzy edges and \(s\) is the average number of neighbors for each fuzzy
edge. In summary, the time complexity of F-Attractor is $O(|E| + T \cdot s \cdot |FE|)$ and $3 \leq T \leq 30$.

4 Experiment

In this section, we present an experimental evaluation of the application of the F-Attractor on several synthetic and real-world networks.

4.1 Experimental setup

Comparison algorithms. To examine the performance of the F-Attractor algorithm, we compare it with the five representative community detection algorithms listed in Table 3. The InfoMap, FastGreedy, and Louvain algorithms are considered to be the best for disjoint community detection\footnote{1, 4}, the LPA algorithm offers high-speed community detection, and the Attractor algorithm is a native algorithm that builds on the distance dynamics model. For all these community detection algorithms, we used the default recommended parameters to obtain the best experimental results.

Evaluation measures. To extensively compare the effectiveness of different community detection algorithms, we selected three widely used metrics to evaluate their cluster quality. (1) The first metric is Normalized Mutual Information (NMI)\footnote{29}, which is defined in the context of classical clustering for comparing two different partitions of one data set by measuring the amount of information they have in common. (2) The second metric is the popular Adjusted Rand Index (ARI)\footnote{32}, which calculates the total number of pairs that belong to the same cluster, or to different clusters, at the same time in both true clusters and in the clustering results. (3) The third metric is an internal measure, known as Newman’s modularity\footnote{34}, which characterizes the overall quality of the community detection. All metrics scale between 0 and 1 to convey a random or perfect clustering result, respectively.

Experimental platform. To simulate a large network, we rented a high-performance server from the National Super Computing Center of Changsha, located in the Hunan Province of China. This high-performance server contains 2x Intel Xeon E5-2600 series processors and 176 GB of memory. Each processor has 16 cores and 32 threads to maximize the concurrent execution of multithreaded applications. Moreover, each processor has an L3 cache of 20 MB and runs at a frequency of 3.3 GHz, with two 8 GT/s QuickPath Interconnect (QPI) links between processors. We applied all algorithms on this high-performance server with the Windows server 2012 operating system and implemented the F-Attractor and Attractor algorithms in Python. For the other four algorithms, we downloaded their respective implementations in Python from the official websites of the corresponding authors.

4.2 Sensitivity of parameter $\varepsilon$

Our first objective in the experimental evaluation was to validate the sensitivity of parameter $\varepsilon$, which is a probability threshold used to determine whether an edge of the network is internal. For an edge $e(u, v)$, when the revised edge-clustering coefficient $DECC(u, v)$ is greater than or equal to $\varepsilon$, we consider this edge $e(u, v)$ to be internal, so it need not to take part in the distance dynamics model. In the opposite case, the edge $e(u, v)$ is a fuzzy edge and must take part in the distance dynamics model to obtain the final distance. Generally, for higher values of parameter $\varepsilon$, the F-Attractor algorithm yields fewer internal edges, more fuzzy edges take part in the distance dynamics model, and the time acceleration is lower, and vice versa. To obtain the best value of parameter $\varepsilon$, we selected two real-world networks (polblogs and Amazon) to test its sensitivity. By gradually modulating parameter $\varepsilon$, we observed any changes in performance by the Percentage of the Prejudged Internal Edges (PPIE), NMI, ARI, and modularity.

With $\varepsilon$ values ranging from 0 to 0.8 on the polblogs network, Figs. 9(a1)–9(a4) show plots of the PPIE, NMI, ARI, and modularity, respectively. From these four plots, we can state the following. (1) As parameter $\varepsilon$ increases from 0.05 to 0.8, the PPIE value reduced.
Fig. 9 Sensitivity of parameter $\varepsilon$.

from 95% to 0.4%, the NMI value increased from 0.19 to 0.49, the ARI value increased from 0.15 to 0.41, the modularity value increased from 0.14 to 0.56, (2) When parameter $\varepsilon$ is equal to the average value $\text{avgDECC} = 0.27$ of the revised edge-clustering coefficient (DECC), the PPIE value is 59%, the modularity value is 0.55, the NMI value is 0.48, and the ARI value is 0.39, as shown by the red dotted line in Figs. 9(a1) to 9(a4), respectively. (3) When parameter $\varepsilon$ is more than $\text{avgDECC} = 0.27$, the PPIE value continues to decrease significantly, but the modularity, NMI, and ARI values hardly increase.

Similar to the polblogs network, Figs. 9(b1)–9(b4) demonstrate the sensitivity of parameter $\varepsilon$ on the Amazon network, with regard to the PPIE, NMI, ARI, and modularity, respectively. From these figures, we can make the following three observations. (1) As parameter $\varepsilon$ increases from 0.05 to 0.8, the PPIE value is reduced from 80% to 1%, the modularity value increases from 0.41 to 0.75, the NMI value increases from 0.5 to 0.74, and the ARI value progresses from 0.19 to 0.47. (2) When parameter $\varepsilon$ is equal to the average value $\text{avgDECC} = 0.38$ of the revised edge-cluster coefficient, the PPIE value is 54%, the modularity value is 0.74, the NMI value is 0.73, and the ARI value is 0.46. (3) When parameter $\varepsilon$ is greater than $\text{avgDECC} = 0.38$, the PPIE value continues to decrease significantly, but the modularity, NMI, and ARI values...
hardly increase.

According to the above observations with respect to polblogs and Amazon networks, we find that the best parameter \( \varepsilon \) value differs for different networks. Moreover, when the parameter \( \varepsilon \) is equal to the avgDECC of the network, the performance of the F-Attractor algorithm is good. Hence, we find that avgDECC is the best parameter \( \varepsilon \) value for different networks. In the following experiments, we set parameter \( \varepsilon = \text{avgDECC} \) as the default value.

4.3 Synthetic networks

4.3.1 Network generation

To compare the performance of various community detection algorithms, we used the LFR benchmark to generate several synthetic networks that feature distinct characteristics. We defined the generation model of the LFR benchmark as LFR(C#, Cs, \( K_{\text{max}} \), \( \mu \)), in which C# indicates the number of communities, Cs represents the number of nodes in one community, \( K_{\text{max}} \) is the maximum degree of the node, and \( \mu \) is the mixing parameter for indicating the fraction of links of each node that are outside its community, which is used to manage the difficulty associated with community separation.

Based on these four parameters, we generated six synthetic networks with ground truth, as listed in Table 4. For a fair comparison, the six synthetic networks have different network scales, numbers of communities, average node degrees, and noise edges. By modulating parameters C# and Cs in each synthetic network, we imposed different network scales (nodes and edges) and numbers of communities. By modulating parameter \( K_{\text{max}} \), each synthetic network featured different average node degrees. By modulating parameter \( \mu \), each synthetic network was given a different number of noise edges in each community.

4.3.2 Community detection performance

The second objective of the experimental evaluation was to test the community detection performance for various algorithms on the LFR synthetic networks and to observe the difference between our F-Attractor algorithm and the native Attractor algorithm with respect to NMI, ARI, modularity, and computation time.

Figure 10 shows the community detection performances of the various algorithms on the LFR synthetic networks, wherein Fig. 10a shows the NMI results, Fig. 10b shows the ARI results, and Fig. 10c shows the modularity results. From these results, we can make the following observations. (1) With respect to NMI, all six community detection algorithms generated good results and the average NMI value is greater than 0.6. Overall, we can see that the Attractor, F-Attractor, and Louvain algorithms generated the best results and stability, with the InfoMap and LPA algorithms next, and the FastGreedy algorithm having the worst performance. (2) With respect to ARI, the trends of the six algorithms are very uneven, which implies that the performances of the six algorithms were very unstable on the LFR networks. For instance, the ARI values for the six algorithms are good on the LFR-1 and LFR-4 networks, whereas they are poor on the LFR-2 and LFR-3 networks. (3) With respect to the modularity, the six algorithms show very obvious differences, with the performances of the Attractor, F-Attractor, Louvain, and InfoMap algorithms being better than those of the LPA and FastGreedy algorithms. (4) If we merge the NMI, ARI, and modularity results, we find that the Attractor, F-Attractor, and Louvain algorithms perform better and have better stability than the other three algorithms on the high-density networks (LFR-1, LFR-4, and LFR-6) and the sparse network (LFR-3). (5) If we focus on the Attractor and F-Attractor algorithms, we find that the NMI, ARI, and modularity results of the F-Attractor algorithm are very close to those of the native Attractor algorithm on most LFR networks. However, the performance of the F-Attractor is better than that of the Attractor on the LFR-1 and LFR-4 networks.

| Network | Number of nodes | Number of edges | Average degree | \( \mu \) | \( K_{\text{max}} \) | Cs | C# | AvgDECC |
|---------|----------------|----------------|----------------|---------|----------------|----|----|---------|
| LFR1    | 25 322         | 562 103        | 44.40          | 0.2     | 55             | 150| 200| 0.1     |
| LFR2    | 51 162         | 305 519        | 11.94          | 0.25    | 15             | 150| 400| 0.1     |
| LFR3    | 134 333        | 1 256 456      | 5.61           | 0.3     | 8              | 200| 800| 0.1     |
| LFR4    | 202 554        | 4 907 551      | 48.46          | 0.25    | 55             | 200| 1200| 0.4     |
| LFR5    | 381 671        | 2 589 060      | 18.81          | 0.3     | 22             | 300| 1500| 0.18    |
| LFR6    | 505 846        | 10 880 874     | 43.02          | 0.25    | 55             | 300| 2000| 0.29    |
Table 5 lists the computation times of the six algorithms on the LFR networks, from which we can see that the computation time of the LPA algorithm is the lowest, the Louvain algorithm is next, the F-Attractor algorithm is the third, the InfoMap, Attractor, and FastGreedy algorithms have the longest computation times. If we compare the LPA and InfoMap algorithms, we find that the computation time of InfoMap is close to 30 times that of LPA. Moreover, with respect to the Attractor and F-Attractor algorithms, the computation time of Attractor is close to eight times that of the F-Attractor. We provide more details on the time acceleration of the F-Attractor in the next section.

4.3.3 Time acceleration performance
The third objective of our experimental evaluation was to validate the time acceleration of the F-Attractor algorithm on LFR synthetic networks with regard to internal edge prejudgment, triangle distance, and overall time acceleration.

(1) Internal edge prejudgment.
The goal in internal edge prejudgment is to reduce the number of fuzzy edges taking part in the distance dynamics model. Generally, when the number of prejudged internal edges was greater, the number of fuzzy edges taking part in the distance dynamics model was lower, and the computation time of the F-Attractor was lower, and vice versa. For internal edge prejudgment, we propose two prejudgment rules: the node-based and edge-based prejudgment rules. Figure 11 shows the internal edge prejudgment results for the LFR networks and the relative proportions of the two different rules. In the figure, the height of each column indicates the percentage of prejudged internal edges, with the blue region representing the proportion

| Algorithm    | LFR-1 | LFR-2 | LFR-3 | LFR-4 | LFR-5 | LFR-6 |
|--------------|-------|-------|-------|-------|-------|-------|
| FastGreedy   | 236   | 91    | 1421  | 14722 | 9243  | 24036 |
| InfoMap      | 122   | 108   | 679   | 5645  | 5328  | 15478 |
| Louvain      | 11    | 7     | 36    | 297   | 242   | 563   |
| LPA          | 5     | 2     | 27    | 146   | 117   | 292   |
| Attractor    | 187   | 83    | 789   | 8532  | 6431  | 18346 |
| F-Attractor  | 47    | 35    | 185   | 1427  | 911   | 2738  |

Fig. 11 Internal edge prejudgment on the LFR networks.
of the node-based prejudgment rule and the light red region representing that of the edge-based prejudgment rule. We can see that, the percentage of prejudged internal edges is large and exceeds 40% on the three high-density networks (LFR-1, LFR-4, and LFR-6). However, on the three sparse networks (LFR-2, LFR-3, and LFR-5), the percentage of prejudged internal edges is small, corresponding to 16%, 19%, and 14%, respectively. Thus, if we contrast the two prejudgment rules, we find that the node-based prejudgment rule works poorly on LFR networks and the edge-based prejudgment rule works well.

(2) Time acceleration of triangle distance.

The idea behind the triangle distance is to use the known distances of two real edges to measure the distance of one virtual edge represented by an exclusive neighbor pair. Thus, the aim of triangle distance is to reduce the calculation time of the dynamic distance for all exclusive neighbors in the dynamic interaction process. Figure 12 shows the time acceleration of the triangle distance on the LFR networks, in which the “before” column indicates the computation time of the F-Attractor without the help of the triangle distance, the “after” column represents the computation time of the F-Attractor with the help of the triangle distance, and the red number is the time acceleration of the triangle distance. As shown in the figure, on the three high-density networks (LFR-1, LFR-4, and LFR-6), the time acceleration of the triangle distance is small, corresponding to 19%, 22%, and 24%, respectively. This is because the percentage of prejudged internal edges is very large and the number of fuzzy edges taking part in the distance dynamics model is small. However, on the three sparse networks (LFR-2, LFR-3, and LFR-5), the time acceleration of the triangle distance is large, corresponding to 36%, 32%, and 22%, respectively.

(3) Overall time acceleration of the F-Attractor algorithm.

Table 6 lists the overall time acceleration of the F-Attractor algorithm in contrast to that of the native Attractor algorithm on the LFR networks. From the table, we can see that when the scale of the network is small (LFR-1, LFR-2, or LFR-3), the time acceleration of the F-Attractor algorithm is small; with the increase of the scale of the network, the time acceleration of the F-Attractor algorithm increases gradually; and when the number of edges in the network exceeds 10 million (LFR-6), the computation time of the native Attractor algorithm is 7 times that of the F-Attractor algorithm. Specifically, the computation time is reduced from 18 346 s to 2738 s, thus demonstrating the significant time acceleration of the F-Attractor. Furthermore, Fig. 13 shows the composition of the computation time of the F-Attractor, in which the green region represents the time proportion of the internal edge prejudgment and the light red region represents the time proportion of the dynamic interaction process. As shown in this figure, we can see that the time proportion of internal edge prejudgment is smaller than that of the dynamic interaction process on high-density and sparse networks.

Table 6 Overall time acceleration of the F-Attractor algorithm on the LFR networks.

| Network | Attractor (s) | F-Attractor (s) | Saving (s) | Acceleration (x) |
|---------|--------------|----------------|------------|-----------------|
| LFR1    | 187          | 47             | 140        | 4.00            |
| LFR2    | 83           | 35             | 48         | 2.37            |
| LFR3    | 789          | 185            | 604        | 4.26            |
| LFR4    | 8532         | 1427           | 7105       | 6.00            |
| LFR5    | 6431         | 911            | 5520       | 7.06            |
| LFR6    | 18 346       | 2738           | 15 608     | 6.70            |

Fig. 12 Time acceleration of the triangle distance on the LFR networks.

Fig. 13 Composition of the computation time of the F-Attractor algorithm on the LFR networks.
4.4 Real-world networks

4.4.1 Network selection

To further evaluate the performance of the various community detection algorithms, we chose six typical real-world networks with ground truth for our experiments, as listed in Table 7. All these real-world networks are publicly available from the UCI network data repository (https://networkdata.ics.uci.edu/index.php) and the Stanford large network dataset collection (http://snap.stanford.edu/data/). The six real-world networks represent different network types: Youtube and LiveJournal are social networks, polblogs is a blog network, Football is a football network, DBLP is a collaboration network, and Amazon is an Amazon product network. Moreover, these six real-world networks are characterized by different network densities, with Amazon, Youtube, and DBLP being sparse networks, and Football, LiveJournal, and polblogs being dense networks.

4.4.2 Community detection performance

The fourth objective of our experimental evaluation was to test the community detection performance of the various algorithms on real-world networks and to observe the difference between those of our F-Attractor algorithm and the native Attractor algorithm with respect to NMI, ARI, modularity, and computation time.

Figure 14 shows the community detection performance of the six algorithms on the real-world networks, wherein Fig. 14a plots the NMI results, Fig. 14b plots the ARI results, and Fig. 14c plots the modularity results. From Fig. 14, we can make the following remarks. (1) With respect to NMI, the six algorithms exhibit different advantages. On Football and polblogs, two high-density networks, the Attractor, F-Attractor, and Louvain algorithms have the best NMI values, with the InfoMap algorithm next, and the LPA and FastrGreedy algorithms being the worst. Moreover, the Attractor, F-Attractor, and Louvain algorithms also exhibit better stability than the other three. (2) With respect to ARI, the differences of the six algorithms are significant, with the average ARI values for the Attractor, F-Attractor, and Louvain algorithms being better than those of the other three. (3) With respect to modularity, the average modularity values
for the InfoMap, LPA, and FastGreedy algorithms are significantly poorer than those of the other three (Attractor, F-Attractor, and Louvain). (4) If we consider NMI, ARI, and Modularity together, we find that the overall performances of the Attractor, F-Attractor, and Louvain algorithms are better than those of the InfoMap, LPA, and FastGreedy algorithms on both high-density real-world networks (Football, polblogs, and LiveJournal) and sparse real-world networks (Amazon, DBLP, and Youtube). (5) If we compare the F-Attractor and Attractor algorithms, the NMI, ARI, and Modularity results for the F-Attractor algorithm are very close to those of the native Attractor algorithm on most real-world networks. In addition, on some real-world networks, the performance of the F-Attractor algorithm even exceeds that of the native Attractor algorithm.

Table 8 shows the computation time of the six algorithms on the real-world networks, in which we can see that the computation time of the LPA algorithm is the shortest, the Louvain algorithm is next, the F-Attractor algorithm is the third, and the Attractor, InfoMap, and FastGreedy algorithms have the longest computation times. If we compare LPA and FastGreedy algorithms, when the number of edges in the network exceeds 30 million (LiveJournal), the computation time of FastGreedy is close to 90 times that of the LPA, i.e., the difference in computation time of the two algorithms is very large. If we compare the F-Attractor and Attractor algorithms, we find that the computation time of the F-Attractor algorithm is far shorter than that of the native Attractor algorithm. We discuss the time acceleration of the F-Attractor in greater detail in the next section.

### 4.4.3 Time acceleration performance

The fifth objective of our experimental evaluation was to validate the time acceleration of the F-Attractor algorithm on real-world networks with regard to internal edge prejudgment, triangle distance, and overall time acceleration.

**1) Internal edge prejudgment.**

As with the LFR synthetic networks, here, we discuss the internal edge prejudgment performance on real-world networks. In Fig. 15, the height of each column indicates the percentage of prejudged internal edges, the blue region represents the proportion of internal edges identified by the node-based prejudgment rule, and the light red region represents the proportion of internal edges identified by the edge-based prejudgment rule. From the figure, we can see that, on the Football and polblogs networks, the number of internal edges identified by the node-based rule is far smaller than the number identified by the edge-based rule. However, on the DBLP and Youtube networks, the number of internal edges identified by the node-based rule is far greater than the number identified by the edge-based rule. On the Amazon and LiveJournal networks, the numbers of internal edges identified by the two different rules are almost equal. Considering all of the real-world networks, we can easily see that internal edge prejudgment can significantly reduce the number of fuzzy edges and thereby accelerate the process of community detection.

**2) Time acceleration of triangle distance.**

Figure 16 further shows the time acceleration of the triangle distance on real-world networks. In the figure, the “before” column shows the computation time of the F-Attractor without the help of the triangle distance, the “after” column presents the computation time of the F-Attractor with the help of the triangle distance, and the red number represents the time acceleration of the triangle distance. As shown in Fig. 16, the time accelerations of the triangle distances on the Football, polblogs, and Youtube networks are less than that on the other three networks, corresponding to 27%, 31%, and 34%, respectively. Considering all of the real-world networks combined, the average time acceleration for

### Table 8 Computation time of different algorithms on real-world networks.

| Algorithm    | Football | polblogs | Amazon | DBLP  | Youtube | LiveJournal |
|--------------|----------|----------|--------|-------|---------|-------------|
| FastGreedy   | 1        | 31       | 4809   | 6666  | 11024   | 93746       |
| InfoMap      | 1        | 21       | 1057   | 3766  | 7476    | 61645       |
| Louvain      | 1        | 6        | 57     | 225   | 237     | 1867        |
| LPA          | 1        | 3        | 36     | 110   | 162     | 1125        |
| Attractor    | 1        | 22       | 5507   | 7910  | 9271    | 87546       |
| F-Attractor  | 1        | 11       | 565    | 771   | 1341    | 8222        |
(3) Overall time acceleration of the F-Attractor algorithm.

Table 9 shows the overall time acceleration of the F-Attractor algorithm in comparison with that of the native Attractor algorithm on real-world networks. From the table, we see that when the network is small (Football and polblogs), the time acceleration of the F-Attractor is small, with the increase of network scale, the time acceleration of F-Attractor gradually increases, and when the number of edges in the network exceeds 30 million (LiveJournal), the computation time of the native Attractor algorithm is close to 11 times that of the F-Attractor algorithm, which corresponds to 87,546 s and 8,222 s, respectively. Figure 17 shows a plot of the composition of the computation time for the F-Attractor algorithm on the real-world networks, in which, on the Football and polblogs networks, the time proportion of the internal edge prejudgment is far smaller than that of the dynamic interaction process. However, on the other four real networks, the time proportion of the internal edge prejudgment is close to that of the dynamic interaction process.

Table 9 Overall time acceleration of the F-Attractor algorithm on real-world networks.

| Network     | Attractor (s) | F-Attractor (s) | Saving time (s) | Acceleration (×) |
|-------------|---------------|-----------------|-----------------|------------------|
| Football    | 1             | 1               | 0               | 1.0              |
| polblogs    | 22            | 11              | 11              | 2.0              |
| Amazon      | 5507          | 565             | 4942            | 9.7              |
| DBLP        | 7910          | 771             | 7139            | 10.3             |
| Youtube     | 9271          | 1341            | 7930            | 6.9              |
| LiveJournal | 87,546        | 8222            | 79,324          | 10.6             |
the distance dynamics model. The triangle distance concept uses two known distances to measure the distance of an exclusive neighbor without any extra computation. Based on these two strategies, we improve the performance of the distance dynamics model and describe the F-Attractor algorithm process. We performed a series of extensive experiments on multiple synthetic and real-world networks, the results of which demonstrate the advantages of our proposed algorithm.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Nos. 61573299, 61174140, 61472127, and 61272395); the Social Science Foundation of Hunan Province (No. 16ZDA07); China Postdoctoral Science Foundation (Nos. 2013M540628 and 2014T70767); the Natural Science Foundation of Hunan Province (Nos. 14JJ3107 and 2017JJ5064); and the Excellent Young Scholars Project of Hunan Province (No. 15B087).

References

[1] S. Fortunato, Community detection in graphs, *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.
[2] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, and P. Spyridonos, Community detection in social media, *Data Mining and Knowledge Discovery*, vol. 24, no. 3, pp. 515–554, 2012.
[3] X. Zhang and W. Cheng, Pattern mining in linked data by edge-labeling, *Tsinghua Sci. Technol.*, vol. 21, no. 2, pp. 168–175, 2016.
[4] N. Barbieri, F. Bonchi, and G. Manco, Cascade-based community detection, in *Proc. 6th ACM International Conference on Web Search and Data Mining*, Rome, Italy, 2013, pp. 33–42.
[5] Y. Yang, H. Guo, T. Tian, and H. Li, Link prediction in brain networks based on a hierarchical random graph model, *Tsinghua Sci. Technol.*, vol. 20, no. 3, pp. 306–315, 2015.
[6] Y. Li, K. He, D. Bindel, and J. E. Hopcroft, Uncovering the small community structure in large networks: A local spectral approach, in *Proc.24th International Conference on World Wide Web*, Florence, Italy, 2015, pp. 658–668.
[7] Y. van Gennip, B. Hunter, R. Ahn, P. Elliott, K. Luh, M. Halvorson, S. Reid, M. Valasik, J. Wo, G. E. Tita, et al., Community detection using spectral clustering on sparse geosocial data, *SIAM Journal on Applied Mathematics*, vol. 73, no. 1, pp. 67–83, 2013.
[8] A. P. Traganitis, K. Slavakis, and B. G. Giannakis, Spectral clustering of large-scale communities via random sketching and validation, in *the 49th Annual Conference on Information Sciences and Systems (CISS)*, New York, NY, USA, 2015, pp. 1–6.
[9] M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks, *Physical Review E*, vol. 69, no. 2, pp. 26–33, 2004.
[10] F. M. D. Marquitti, P. R. Guimares, M. M. Pires, and L. F. Bittencourt, MODULAR: Software for the autonomous computation of modularity in large network sets, *Ecography*, vol. 37, no. 3, pp. 221–224, 2014.
[11] R. Aktunc, I. H. Toroslu, M. Ozer, and H. Davulcu, A dynamic modularity based community detection algorithm for large-scale networks, in *IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, Paris, France, 2015, pp. 1177–1183.
[12] C. Schulke and F. Ricci-Tersenghi, Multiple phases in modularity-based community detection, *Physical Review E*, vol. 92, no. 4, pp. 42–48, 2015.
[13] J. Xiang, T. Hu, Y. Zhang, K. Hu, J. M. Li, X. K. Xu, C. C. Liu, and S. Chen, Local modularity for community detection in complex networks, *Physica A: Statistical Mechanics and its Applications*, vol. 443, no. 1, pp. 451–459, 2016.
[14] S. Fortunato and M. Barthelemy, Resolution limit in community detection, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 1, pp. 36–41, 2007.
[15] U. N. Raghavan, R. Albert, and S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, *Physical Review E*, vol. 76, no. 3, pp. 96–106, 2007.
[16] H. Sun, J. Liu, J. Huang, G. T. Wang, Z. Yang, Q. B. Song, and X. L. Jia, CenLP: A centrality-based label propagation algorithm for community detection in networks, *Physica A: Statistical Mechanics and its Applications*, vol. 436, no. 12, pp. 767–780, 2015.
[17] C. L. Staudt and H. Meyerhenke, Engineering parallel algorithms for community detection in massive networks, *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 1, pp. 171–184, 2016.
[18] A. Arenas and A. Díaz-Guilera, Synchronization and modularity in complex networks, *The European Physical Journal Special Topics*, vol. 143, no. 1, pp. 19–25, 2007.
[19] J. E. M. Oliveira and M. G. Quiles, Community detection in complex networks using coupled kuramoto oscillators, in *the 14th International Conference on Computational Science and Its Applications (ICCSA)*, Washington DC, USA, 2014, pp. 85–90.
[20] J. E. M. Oliveira, M. G. Quiles, M. D. N. Maia, and E. E. N. Macau, Community detection with lower time complexity using coupled Kuramoto oscillators, in *Proc. of 30th Annual ACM Symposium on Applied Computing*, Salamanca, Spain, 2015, pp. 1160–1166.
[21] J. Shao, Z. Han, Q. Yang, and T. Zhou, Community detection based on distance dynamics, in *Proc. of 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Sydney, Australia, 2015, pp. 1075–1084.
[22] J. Wang, M. Li, H. Wang, and Y. Pan, Identification of essential proteins based on edge clustering coefficient, *IEEE/ACM Transactions on Computational Biology & Bioinformatics*, vol. 9, no. 4, pp. 1070–1080, 2012.
[23] M. Girvan and M. E. J. Newman, Community structure in social and biological networks, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, pp. 7821–7826, 2002.

[24] A. W. Rives and T. Galitski, Modular organization of cellular networks, *Proceedings of the National Academy of Sciences of the United States of America*, vol. 100, no. 3, pp. 1128–1133, 2003.

[25] J. C. Gower, Measures of similarity dissimilarity and distance, *Encyclopedia of Statistical Sciences*, vol. 5, no. 3, pp. 397–405, 1985.

[26] P. Willett, J. M. Barnard, and G. M. Downs, Chemical similarity searching, *Journal of Chemical Information and Computer Sciences*, vol. 38, no. 6, pp. 983–996, 1998.

[27] H. Spath, *Cluster Analysis Algorithms for Data Reduction and Classification of Objects*. Chichester, UK: Ellis Horwood, 1980.

[28] A. H. Lipkus, A proof of the triangle inequality for the Tanimoto distance, *Journal of Mathematical Chemistry*, vol. 26, no. 3, pp. 263–265, 1999.

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

[30] A. Clauset, E. M. Newman, and C. Moore, Finding community structure in very large networks, *Physical Review E*, vol. 70, no. 6, pp. 264–277, 2004.

[31] D. V. Blondel, L. J. Guillaume, R. Lambiotte, and E. Lefebvre, Fast unfolding of communities in large networks, *Journal of Statistical Mechanics: Theory & Experiment*, vol. 2008, no. 10, pp. 155–168, 2008.

[32] A. Strehl and J. Ghosh, Cluster ensembles a knowledge reuse framework for combining multiple partitions, *The Journal of Machine Learning Research*, vol. 3, no. 12, pp. 583–617, 2003.

[33] W. M. Rand, Objective criteria for the evaluation of clustering methods, *Journal of the American Statistical Association*, vol. 66, no. 336, pp. 846–850, 1971.

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

Lei Chen received the PhD degree from Hunan University, China in 2017. He is currently a lecturer at Hunan University of Science and Technology. His research interests include date mining, web mining, graph mining, cloud computing, and big data scheduling and analysis.

Jing Zhang received the PhD degree from Hunan University in 1997. He is currently a professor at Hunan University. His research interests include complex process control and optimization, big data scheduling and analysis, and data mining.

Lijun Cai received the PhD degree from Hunan University in 2007. He is currently a professor at Hunan University. His research interests include bioinformatics, cloud computing, big data scheduling and management, data mining, web mining, graph mining, and pattern recognition.

Ziyun Deng received the PhD degree from Hunan University in 2016. He is currently a professor at Changsha Commerce&Tourism College. His research interests include high performance computing, logistics information technology, and big data.