IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received July 27, 2019, accepted August 27, 2019, date of publication September 5, 2019, date of current version September 20, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2939535*

# Local Higher-Order Community Detection Based on Fuzzy Membership Functions

**TAO MENG[1], LIJUN CAI[1], TINGQIN HE[1], LEI CHEN[2], AND ZIYUN DENG[3]**
[1]College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China
[2]College of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China
[3]Department of Economics and Trade, Changsha Commerce and Tourism College, Changsha 410082, China

Corresponding author: Lijun Cai (ljcai@hnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61472127 and Grant 61272395, in part by the Natural Science Foundation of Hunan Province under Grant 2017JJ5064, in part by the Social Science Foundation of Hunan Province under Grant 16ZDA07, and in part by the Open Project of State Key Laboratory of Advanced Design and Manufacturing for Vehicle Body under Grant 31715010.

**ABSTRACT** Local community detection, only considering the regional information of the large network, can be used to identify a densely connected community containing the seed node in a network, aiming to address the efficiency problem faced by global community detection. However, most existing studies in local community detection did not account for the higher-order structures crucial to the network, but rather have simply focused single nodes or edges. Moreover, existing higher-order solutions are not purely local methods, as they still use global search to find the best local community, which leads to a global search problem. Furthermore, the quality of the detected community depends on the location of the seed node, which leads to a seed-dependent problem. Thus, in this paper, we proposed a fuzzy agglomerative algorithm (FuzLhocd) for local higher-order community detection based on different fuzzy membership functions. To solve the global search problem, we introduce a novel, purely localized metric called local motif modularity. Based on this local metric, FuzLhocd only needs to visit a limited number of neighborhoods around the seed node. To solve the seed-dependent problem, we systematically studied the formation of the local community, divided the process of local community detection into three stages and employed various fuzzy membership functions at different stages. Our extensive experiments based on both real-world and synthetic networks demonstrated that FuzLhocd not only runs efficiently locally but also effectively solves the seed-dependent problem and achieves a high accuracy as well. We concluded that our local motif modularity metric and FuzLhocd algorithm is highly effective for local higher-order community detection.

**INDEX TERMS** Local community detection, higher-order structure, motif, fuzzy membership functions.

## I. INTRODUCTION

Community structures naturally exist in many real-world networks such as social networks, collaboration networks, biological networks and other types of complex networks [1]–[4]. The aim of community detection is to identify all communities in a global network, which has been widely studied in the literature and remains a fundamental problem in complex network analysis [5]–[7]. However, it is often expensive (even no way) to obtain the global information of the network in many real-world networks of increasing size [8], [9]. Thus, local community detection, which has recently drawn considerable research interest, is a related but distinct problem. The aim of local community detection is to identify a densely connected community containing the seed node by exploring a small region of the network in the vicinity of the seed node [10]. This technique has a wide range of applications for analyzing complex networks because we often only care about the local community to which the seed node belongs. For example, the goal of the friend recommendation feature of WeChat is to recommend candidate friends to a specific user, $u$. Intuitively, only those who are in the same community as $u$ are but who are not yet $u$'s friends are suggested [11]. Under such circumstances, local community detection is more appropriate.

Various methods have been proposed to detect the local community [12]–[16]. However, most existing local community detection algorithms are not designed to account

The associate editor coordinating the review of this manuscript and approving it for publication was Michael Lyu.

for the higher-order structures crucial to the networks; rather, they are simply based on individual nodes or edges and do not consider how these nodes connect to form small network substructures. The highly developed methods of local modularity $R$ [8], [17] and local modularity $M$ [18], [19] do not consider such higher-order connectivity structures as crucial to the organization of complex networks. However, increasing evidence shows that incorporating higher-order connectivity patterns not only improves the accuracy of assessing node membership but also improves the accuracy of identifying community members for a given domain [20]–[23]. For example, triangles are important higher-order structures for social networks, and feedforward loops are important higher-order structures for transcriptional regulation networks [21]. In this paper, we describe new algorithms for local higher-order community detection.

In recent research, Austin R. Benson [21], [24] used spectral clustering to complete higher-order community detection. Since this algorithm is globally based, Hao Yin [25] then proposed the MAPPR algorithm, a local community detection algorithm that sorts and uses motif conductance after using Approximate Personalized PageRank. However, two common problems are still unsolved in the MAPPR algorithm.

*1. Global Search Problem:* Before using the Approximate Personalized PageRank, MAPPR needs to precompute the number of motif instances that contain each pair of nodes. All nodes in the graph need to be visited, which is computationally very costly. Therefore, it is difficult to apply MAPPR to large-scale networks due to its high time complexity. In addition, MAPPR is often expensive and cannot be used to obtain the entire network in many real-world applications. Thus, purely local community detection based only on local information becomes essential.

*2. Seed-Dependent Problem:* Because MAPPR does not consider the dynamic formation process of the local community, the quality of the detected community critically depends on the location of the seed node, which leads to a seed-dependent problem. Specifically, the local community detection algorithm obtains a low-quality community from a noncenter seed node, while it obtains a high-quality community from a center seed node. In addition, MAPPR cannot effectively handle seed nodes with special roles, such as those that serve as hubs bridging different communities.

Considering these limitations, we propose a new local higher-order community detection algorithm (FuzLhocd) for finding the community of the seed node based on fuzzy membership functions. Our local method searches the target community, $C$, with maximal local motif modularity. Given the seed node, $s$, in a graph, $G$, and a motif, $M$, the concrete details of the local community detection procedure are as follows. First, in the core detecting stage, FuzLhocd detects core nodes of the target community. Second, in the community expansion stage, FuzLhocd expands the core nodes to obtain a rough target community of the seed node. Third, in the community optimization stage, FuzLhocd optimizes the rough target community by collecting nodes that should not be omitted. Additionally, FuzLhocd applies different fuzzy membership functions at different stages. Specifically, in the core detecting stage, the joining of a new node should improve the value of the local motif modularity and produce the largest gain for the internal motif. In the community expansion stage, the joining of the new node also needs to improve the local motif modularity value and yield the largest gain in the external motif value. In the community optimization stage, if the structural similarity between node, $s$, and the target community, $C$, is greater than the threshold parameter $\lambda$, FuzLhocd will consider that the relationship between $s$ and the target community $C$ is very close and that node $s$ should join $C$.

In summary, compared with MAPPR, our local higher-order community detection algorithm, FuzLhocd, has the following advantages:

*1. Purely Local Algorithm:* The FuzLhocd method is based on a purely local metric, and it only needs to visit a limited number of neighborhoods around the seed node; the method does not require the number of motif instances to be precomputed for all nodes in the entire network. When the neighborhood grows more slowly than the entire graph size, FuzLhocd is significantly faster than MAPPR. In fact, the average degree of nodes is often significantly smaller than the entire graph size.

*2. More Robust:* The FuzLhocd method divides the process of local community detection into three stages and employs different fuzzy membership functions at different stages to detect the local community. First, FuzLhocd finds the most relevant core nodes of the target community during the core detecting stage and then extends the community in the following two stages. This can solve the seed-dependent problem of the MAPPR method. Thus, FuzLhocd is more robust to the seed-invalid problem than MAPPR.

*3. Fewer Parameters:* Our method only needs to specify the threshold parameter, $\lambda$, at the community optimization stage. In fact, at the beginning of the community optimization stage, the target community is roughly formed. Thus, by using different $\lambda$ values for community optimization, the quality of the detected community is insensitive to parameter $\lambda$. Although MAPPR may have more leverage on the properties of the local community because it has more parameters, it is much more difficult to determine the proper values of the parameters for different networks.

The remaining sections of this paper are organized as follows. We review the related work in Section *II*, and design a purely local metric for local higher-order community detection in Section *III*. Section *IV* presents a naive algorithm based on a global fuzzy membership function. We further propose a more efficient algorithm based on dynamic fuzzy membership functions in Section *V*. Section *VI* evaluates all introduced algorithms using extensive experiments, and Section *VII* concludes the paper.

## II. RELATED WORKS

The work presented in this paper is closely related to community detection, local community detection, higher-order graph clustering and the fuzzy membership function.

### A. COMMUNITY DETECTION

The aim of community detection is to identify all communities in a global network, which has been extensively studied [26]–[30]. A representative method is the modularity measure [26], which identifies good communities in many real networks; however, modularity becomes unreasonable for large networks (resolution limit problem). Considering this limitation, label propagation has attracted widespread interest because it can generally detect communities in almost linear time [28]. However, this algorithm has poor stability due to the randomness of the label propagation process. In addition to extracting communities, finding nodes with special roles, such as hubs and outliers, is also beneficial for understanding the structure of the network. One of the most successful density-based algorithms is SCAN [27], which not only detects meaningful communities but also hubs and outliers. However, the quality of the detected communities is sensitive to the density threshold parameter. Recently, inspired by synchronization clustering [31], Shao *et al*. [30] considered the problem of community detection from a new point of view of distance dynamics. The proposed algorithm has several attractive benefits, such as ''intuitive community detection'' and ''anomaly detection''. However, the methods described above do not consider local community detection, which is important for many real life applications.

### B. LOCAL COMMUNITY DETECTION

The goal of local community detection is to detect a densely connected community that contains the seed node, which has attracted growing research interest [10], [12]–[14], [32]. In general, most existing methods take a seed as an initial community and then expand the community by running a greedy optimization process with a goodness metric. For example, local modularity $R$ [17] and local modularity $M$ [18] have been proposed to detect local communities. However, the metrics of local modularity $R$ and local modularity $M$ may cause a free rider effect issue [13]. Due to these shortcomings, Wu *et al*. [13] proposed a node weighting scheme based on random walk to solve the free-rider effect problem. Moreover, different semantics for local community detection have been studied, such as K-Core [10]. The K-Core community is based on the K-Core subgraph, which contains nodes with at least $k$ edges. However, this method ignores the diameter of the resulting community. In recent research, Huang *et al*. [12] proposed a K-Truss community model based on triangle adjacency to find a cohesive community that contains the seed node. The K-Truss community is based on the K-Truss subgraph, where every edge is contained in at least ($k$-2) triangles within the cohesive community. However, all these local community detection methods are simply based on individual nodes or edges and do not consider the higher-order structures crucial to the network, nor can they effectively handle directed networks.

### C. HIGHER-ORDER GRAPH CLUSTERING

Incorporated higher-order connectivity patterns have the potential to improve clustering results and can be easily applied to community detection in directed networks. Recently, several higher-order graph clustering methods have been studied [20], [21]. Austin *et al*. [20] proposed a tensor spectral clustering (TSC) algorithm to model higher-order network structures. In TSC, higher-order network structures are represented using a tensor and then partition by developing a multilinear spectral method. In addition, Jure *et al*. [21] proposed a novel metric called motif conductance that is based on higher-order structures to measure the quality of the detected communities. The smaller the value of the motif conductance is, the better the detected communities are. Nevertheless, due to inherently different problems, none of these methods can be directly used for local community detection. A recent study by Austin *et al*. [25] proposed a novel approach for local higher-order community detection, namely, MAPPR. MAPPR begins by generalizing the Approximate Personalized PageRank (APPR) algorithm of Anderson *et al*. [24] to detect a local community containing a given seed node with minimal motif conductance. Unfortunately, this method has several problems, which were discussed in Section I.

### D. FUZZY MEMBERSHIP FUNCTION

The fuzzy membership function has been extensively studied [33], [34] and is widely used in fuzzy systems to describe the dynamics of systems [35]. Because the fuzzy membership function can measure the belongingness of nodes in different communities more accurately, the applications of the fuzzy membership function in community detection has attracted much attention in recent years [16], [36]–[39]. For example, Anupam *et al*. [39] proposed a fuzzy agglomerative community detection algorithm (FuzAg) based on the fuzzy membership function. In FuzAg, nodes with a higher self-membership degree are referred to as anchors, and they get a chance to expand their associated community. However, seed nodes are not considered since the focus of FuzAg is not on local community detection but on community detection. In addition, Luo *et al*. [16] provided a systematic analysis of the formation of the local community and proposed corresponding local community detection algorithms based on the dynamic membership function. However, higher-order structures were not considered since this method focuses on the individual node or edge, and it naturally is unable to handle directed networks. In this paper, we study another important, yet largely underexplored local higher-order community detection method, which is focused on the local community detection based on particular network motifs.

## III. LOCAL MOTIF MODULARITY

In this section, we define the problem of local higher-order community detection and then propose a purely local metric for local higher-order community detection.

### A. PROBLEM DEFINITION

Before providing the formal definition of local higher-order community detection, we first formally define the basic notations used throughout this paper. Consider an unweighted graph, $G = (V, E)$, where $V$ and $E$ denote the node set and the edge set, respectively. In this paper, graph $G$ may be a directed graph or an undirected graph. Other types of graphs, such as a weighted graph, can be handled with only slight modifications.

*Definition 1 (Node Neighbors):* The neighbors of a node are a node set composed of all its adjacent nodes. The neighbors of node $u$, denoted by $N(u)$, is defined as follows:

$$N(u) = \{v \in V \mid \{u, v\} \in E\} \cup \{u\}. \tag{1}$$

*Definition 2 (Community Neighbors):* The neighbors of a community are a node set in which all nodes are not in the community but have at least one neighbor in the community. The neighbors of community $C$, denoted by $N(C)$, is defined as follows:

$$N(C) = \{v \in V \mid v \notin C, u \in C, (u, v) \in E\}. \tag{2}$$

*Definition 3 (Network Motif):* The network motif (also called a higher-order structure) is a small connected subgraph and contains multiple nodes and edges, which represent the information interactions between multiple nodes. For example, triangles are important higher-order structures of social networks [21]. The common motifs are shown in Fig. 1.

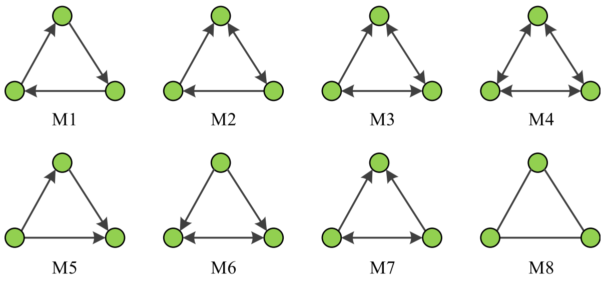


**FIGURE 1. Common motifs. (M1-M8 are triangular motifs and crucial for social networks.).**

*Definition 4 (Motif Degree):* Given graph $G$ and motif $M$, the motif degree of node $u$, denoted by $d_M(u)$, is the number of motif instances in which one of the end points is $u$, is defined as follows:

$$d_M(u) = |\{M \in G \mid u \in M\}|. \tag{3}$$

*Problem Definition (LHOCD):* The problem of local higher-order community detection studied here is defined as follows. Given a graph $G(V, E)$, a seed node $s \in V$ and a motif $M$, find the local community $C$ that contains $s$ and maximizes or minimizes a goodness metric based on motif $M$. The nodes in the local community $C$ are tightly connected, whereas the connection of the nodes between $C$ and $\overline{C}$ is sparse.

### B. MOTIF CONDUCTANCE

In a recent study, Benson *et al.* [21], [25] generalized the motif cut, motif volume, and motif conductance metrics to measure whether a subgraph forms a community. The corresponding concepts are illustrated in Fig. 2, and the formal definitions are as follows:

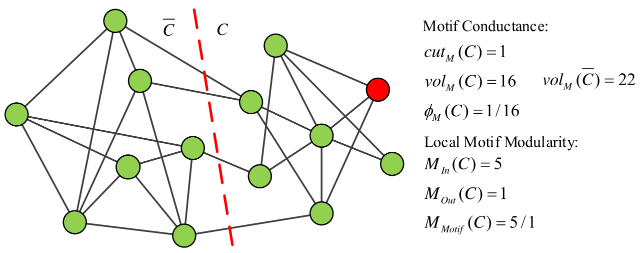


**FIGURE 2. Illustration of motif conductance and local motif modularity when M8 as the motif. (Red node denote query node. The dashed line shows the solution of the local higher-order community detection problem.).**

*Definition 5 (Motif Cut):* Given a motif $M$ and a local community $C$, the motif cut of the local community $C$ is the number of instances of $M$ that have at least one end point in $C$ and at least one end point in $\overline{C}$, which is denoted as $cut_M(C)$.

*Definition 6 (Motif Volume):* Given a motif $M$ and a local community $C$, the motif volume of the local community $C$ is the number of motif instance end points in $C$, i.e., counted over the number of times each node in $C$ participates in motif $M$, which is denoted as $vol_M(C)$.

*Definition 7 (Motif Conductance):* Given a motif $M$ and a local community $C$, the motif conductance of the local community $C$ is the ratio of the motif cut to motif volume, which is denoted as $\phi(C)$.

$$\phi(C) = \frac{cut_M(C)}{min\left\{vol_M(C), vol_M(\overline{C})\right\}}. \tag{4}$$

The motif conductance value is greater than or equal to zero, which indicates the quality of the detected community. When the motif conductance of a local community $C$ is small, the quality of the local community $C$ improves and vice versa.

However, the motif conductance is based on both global ($C$ and $\overline{C}$) and local ($C$) information; thus, it is difficult to apply motif conductance to large-scale graphs since it needs to traverse the entire graph.

### C. LOCAL MOTIF MODULARITY

Considering the above limitations, in this paper, we propose a new goodness metric, the local motif modularity, which is based on internal motifs and external motifs. The corresponding concepts are illustrated in Fig. 2, and the formal definitions are as follows.

*Definition 8 (Internal Motifs):* Given a motif $M$ and a local community $C$, the internal motifs of the local community $C$ are the number of instances of $M$ in which all nodes are in $C$, which is denoted as $M_{In}(C)$.

*Definition 9 (External Motifs):* Given a motif $M$ and a local community $C$, the external motifs of the local community $C$ are the number of instances of $M$ in which only some of the nodes are in $C$, which is denoted as $M_{Out}(C)$.

*Definition 10 (Local Motif Modularity):* Given a motif $M$ and a local community $C$, the local motif modularity of the local community $C$ is the ratio of internal motifs to external motifs, which is denoted as $M_{Motif}(C)$.

$$M_{Motif}(C) = \begin{cases} 0, M_{In}(C) = 0 \text{ } and \text{ } M_{Out}(C) = 0; \\ M_{In}(C), \qquad M_{Out}(C) = 0; \\ M_{In}(C) \big/ M_{Out}(C), \quad otherwise. \end{cases} \tag{5}$$

The local motif modularity is greater than or equal to zero, which indicates the quality of the detected community. A large local motif modularity indicates a good local community, and we will use this metric for evaluating community quality.

### D. WHY LOCAL MOTIF MODULARITY?

Compared with the latest motif conductance [21] metric, the local motif modularity metric has two significant advantages: a purely localized metric and has polynomial time complexity. Based on these beneficial properties, we can design a more efficient, scalable, and easier-to-use algorithm to detect the target community.

#### 1) PURELY LOCALIZED METRIC

Unlike motif conductance, our local motif modularity is a purely localized metric to assess the quality of the detected community and does not require the global information from the original network. Based on the local motif modularity, we can fully consider the characteristics of the local community during local community formation. Specifically, we can design different fuzzy membership functions based on the local motif modularity at different stages to more accurately measure the local community.

#### 2) POLYNOMIAL TIME COMPLEXITY

Because the local motif modularity is a purely localized metric, FuzLhocd can extend the community by running a greedy optimization process to reduce the complexity. It has an extremely fast running time, which is linear with respect to the size of the output community. In addition, the local motif modularity does not need to traverse the entire graph to precompute the number of motif instances that contain each pair of nodes. In general, when there are n nodes in the local community and when the average degree of the nodes in the local community is $d$, there should be at most $nd$ nodes that are traverse.

## IV. GLOBAL FUZZY METHOD

In this section, we devise a global fuzzy membership function based solution for LHOCD. The global fuzzy membership function based approach first take the seed node as an initial community, and then extend the community by running a greedy optimization process with a global fuzzy membership function.

### A. GLOBAL FUZZY MEMBERSHIP FUNCTION

Before presenting the global fuzzy membership function based solution for LHOCD, it is necessary to introduce the main process of the local community formation.

For a network $G$, we can divide it into three parts, i.e., the local community $C$ that the seed node belongs to, region $B$ in which all nodes have at least one neighbor in $C$, and region $U$, which is the unknown part of the network, in which all of its neighbors are not in local community $C$, as Fig. 3 shows. The concrete process of the local community formation is as follows: Initially, the local community $C$ includes only a seed node. Then, one by one node in $B$ will be added into $C$ each time. At the same time, $B$ is updated by adding the border nodes in $U$ into $B$, and $U$ is also updated. This process breaks until the stopping criterion is met. Actually, a goodness metric is usually used to measure whether the stopping criterion is met.

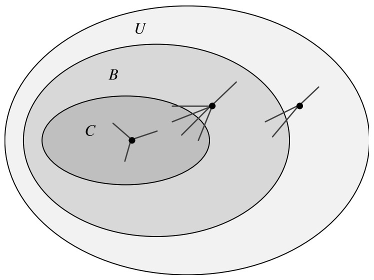


**FIGURE 3. Local community structure. (*C* is the local community. The region *B* in which all nodes have neighbors in *C*. *U* represent the unknown region.).**

*Definition 11 (Motif Gain):* At time step $t$, the local motif modularity value about local community $C$ can be denoted as $M_{Motif}(C)$. At time step $t + 1$, assume node $x$ in region $B$ is chosen and add it into local community $C$. The definition and calculation of the motif gain of metric $M_{Motif}(C)$ is provided as follows.

$$\begin{aligned} \Delta M\left(C\right) &= M_{Motif}^{t+1}\left(C\right) - M_{Motif}^{t}\left(C\right) \\ &= \frac{M_{In}^{t+1}\left(C\right)}{M_{Out}^{t+1}\left(C\right)} - \frac{M_{In}^{t}\left(C\right)}{M_{Out}^{t}\left(C\right)} \\ &= \frac{M_{In}^{t}\left(C\right) + m_{In}^{u}}{M_{Out}^{t}\left(C\right) - m_{In}^{u} + m_{Out}^{u}} - \frac{M_{In}^{t}\left(C\right)}{M_{Out}^{t}\left(C\right)}, \end{aligned} \tag{6}$$

where $m_{In}^{u}$ is the number of internal motifs of node $u$ in $C$, and $m_{Out}^{u}$ is the number of external motifs of node $u$ that are not in $C$. Clearly, $\Delta M\left(C\right)$ may be above 1. Therefore, the global fuzzy membership function, $\delta(u)$, is defined as follows.

$$\delta(u)=\begin{cases}1-\dfrac{1}{1+\Delta M(C)}, & \Delta M(C)\geq 0;\\ 0, & \Delta M(C)<0.\end{cases} \tag{7}$$

In fact, $\Delta M(C)$ is equivalent to $\delta(u)$. If we add node $u$ with the largest value of $\delta(u)$ into the local community $C$, we can also obtain the best motif gain of metric $M_{Motif}(C)$. Moreover, adding a node into $C$ changes the values of $m_{In}^{u}$ and $m_{Out}^{u}$. Thus, $\delta(u)$ is a fuzzy membership function.

## B. THE GLOLHOCD ALGORITHM

Based on obtaining the best motif gain of metric $M_{Motif}(C)$, a straightforward strategy for local higher-order community detection involves only adding the node with the largest value of $\delta(u)$ into the local community $C$ at each time step. The details of the greedy optimization procedure are as follows. First, we take the seed as initial community $C$. Then, the nodes in $B$ that have the largest value of $\delta(u)$ will be chosen and added into $C$ at each time step. Afterwards, region $B$ is updated by putting the boundary nodes in $U$ into $B$. As a result, region $U$ is naturally updated. Finally, the local detection process stops when all nodes in $B$ decrease the value of $M_{Motif}(C)$. Algorithm 1 outlines the procedure to process local higher-order community detection based on the global fuzzy membership function, $\delta(u)$.

**Algorithm 1** : GloLhocd

```
1:  Input: Graph $G = (V, E)$, motif $M$, seed node $s$;
2:  Output: Motif-based local community $C$;
3:  Procedure: GloLhocd($G, M, s$);
4:  Set $C = \{s\}$;
5:  while true do
6:      get $N(C)$ based on definition 2;
7:      $Candidate = \emptyset$;
8:      compute $N = M_{Motif}(C)$ via Eq. (5);
9:      for each node $u \in N(C)$ do
10:         compute $N' = M_{Motif}(C)$ via Eq. (5) if node $u$
            is added into $C$;
11:         if $N - N' \geq 0$ then
12:             $Candidate = Candidate \cup \{u\}$;
13:         end if
14:     end for
15:     if $Candidate \neq \phi$ then
16:         traverse $Candidate$ and select the node that
            maximizes $\delta(u)$, denoted as $vBest$;
17:         $C = C \cup \{vBest\}$;
18:     else
19:         break;
20:     end if
21: end while
22: Return: $C$;
```

## C. LIMITATIONS OF GLOLHOCD

GloLhocd has two drawbacks in its detection processing mechanism of using the global fuzzy membership function.

GloLhocd prefers to add the nodes with a smaller $m_{Out}^{u}$ to the local community first. As a result, the core nodes of the local community may be missed because the $m_{Out}^{u}$ of the core nodes are usually larger at the beginning.

The local community detected by running the greedy optimization process with the global membership function cannot always accurately describe the dynamic structures of the local community. This phenomenon is because the structure of the local community is constantly changing due to joining of new nodes.

We can solve the problems based on the idea of making the local community $C$ cohesive. In fact, with an increasing value of $M_{Motif}(C)$, the local community $C$ becomes cohesive because the nodes with a high density are connected in local community $C$ and because there are relatively low density links between the nodes in region $B$.

As shown from Eq. (6), obtaining a more cohesive local community corresponds to two conditions, a larger $m_{In}^{u}$ and a smaller $m_{Out}^{u}$. The main reason that GloLhocd may miss core nodes initially is that the core nodes of the local community may have a smaller $m_{In}^{u}$ or larger $m_{Out}^{u}$ at the beginning. Hence, it is difficult for GloLhocd to add nodes with a smaller $m_{In}^{u}$ or larger $m_{Out}^{u}$ initially. Therefore, a more reasonable way to identify a cohesive local community is as follows.

First, we choose and add the core nodes with the largest potential value of $m_{In}^{u}$ because the neighbors of node $u$ have the greatest possibility to join local community $C$ in the following time steps. Then, we expand the local community to obtain the border nodes with the smallest potential value of $m_{Out}^{u}$ because the neighbors of node $x$ have little possibility to join local community $C$ in the following time steps. Finally, the local community can be further optimized by adding some fuzzy nodes that have a larger similarity with it.

## V. DYNAMIC FUZZY METHOD

In this section, we devise a more efficient algorithm for LHOCD, which can overcome the challenges introduced in Section IV-C. Specifically, we divide the local higher-order community detection process into three stages, and design the different fuzzy membership functions to detect the local community at different stages.

## A. THE STAGES OF FUZLHOCD

According to the above analysis, we divide the process of local higher-order community detection into three stages to accurately describe the dynamic structures of the local community. Specifically, the stages of the local higher-order community detection are decided by the values of $M_{Motif}(C)$ and $B$', where $B$' is the subset of $N(C)$ containing all nodes that satisfy $\Delta M(C) \geq 0$. The specific division standard is provided as follows.

### 1) CORE DETECTING STAGE

The process of local higher-order community detection is at the core detecting stage when $M_{Motif}(C) < 1$. At this stage, $M_{In}(C)$ is less than $M_{Out}(C)$, and the skeleton of the local community will be basically formed.

### 2) COMMUNITY EXPANSION STAGE

The process of local higher-order community detection is at the community expansion stage when $M_{Motif}(C) \geq 1$ and $B' \neq \phi$. At this stage, $M_{In}(C)$ is larger than $M_{Out}(C)$, and the members of the local community will be roughly identified.

### 3) COMMUNITY OPTIMIZATION STAGE

The process of local higher-order community detection is at the community optimization stage when $B' = \phi$. At this stage, $M_{Motif}(C)$ no longer increases, and the fuzzy nodes of the local community will be fully considered.

## B. CORE DETECTING

Core nodes have a considerable influence on the formation and stability of the local community. Core detection aims to detect the core members of the local community based on the fuzzy membership function. At this stage, the skeleton of the local community will be basically formed. Thus, the fuzzy membership function and core detecting method determine the effectiveness of this local higher-order community detection algorithm.

At the core detecting stage, the number of internal motifs is smaller than the number of external motifs, and the degree of node $u$ belonging to the local community $C$ is measured by the fuzzy membership function, $\delta_c(u)$. The definition of $\delta_c(u)$ is given as follows.

$$\delta_c(u) = \begin{cases} 1 - \dfrac{1}{1 + \Delta m_{In}}, & \Delta M(C) \geq 0; \\ 0, & \Delta M(C) < 0. \end{cases} \tag{8}$$

where $\Delta m_{In} = m_{In}^{N(u)} - m_{In}^{u}$, $m_{In}^{N(u)}$ is the number of internal motifs of node $u$ in $C$ if the neighbors of node $u$ are added into the local community $C$, and $m_{In}^{u}$ is the number of internal motifs of node $u$ in $C$ if node $u$ is added into the local community $C$.

Equation (8) describes the potential internal motif gain of node $u$. At this stage, we add the node with the largest value of $\delta_c(u)$ that also satisfies $\Delta M(C) \geq 0$ into local community $C$. We use an example to explain the potential internal motif gain of a node and core detecting stage, which is shown in Fig. 4.

In Fig. 4, $M_{In}(C) = 0$, $M_{Out}(C) = 3$, and $M_{Motif}(C) = 0$ according to Eq. (5); thus, the detection is at the core detecting stage. From Fig. 4, we observe that only seed node 1 is in local community $C$ and nodes 2, 3, and 4 are the neighbor nodes of community $C$. We calculate $\Delta M(C)$ according to Eq. (6) if nodes 2, 3, and 4 are added into local community $C$, and nodes 2, 3, and 4 satisfy $\Delta M(C) \geq 0$. Then, we calculate $\delta_c(2)$, $\delta_c(3)$, and $\delta_c(4)$ according to Eq. (8). Because $m_{In}^{2} = 0$, $m_{In}^{N(2)} = 4$, $m_{In}^{3} = 0$, $m_{In}^{N(3)} = 4$, $m_{In}^{4} = 0$ and $m_{In}^{N(4)} = 6$, $\delta_c(2) = 0.8$, $\delta_c(3) = 0.8$ and $\delta_c(4) = 0.857$. Therefore, node 4 will be added into local community $C$ at this time step.

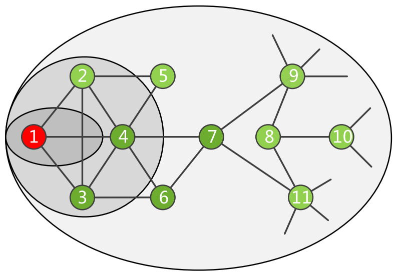


**FIGURE 4.** The core detecting stage.

Moreover, adding a node into $C$ changes the values of $m_{In}^{u}$ and $m_{In}^{N(u)}$. Thus, $\delta_c(u)$ is a fuzzy membership function. Algorithm 2 outlines the procedure to process core detecting based on the fuzzy membership function, $\delta_c(u)$.

**Algorithm 2** : CoreDetecting

```
1:  Input: Graph $G = (V, E)$, motif $M$, seed node $s$;
2:  Output: The core members of the target community;
3:  Procedure: CoreDetecting($G, M, s$);
4:  Set $C_c = \{s\}$;
5:  while true do
6:      get $N(C_c)$ based on definition 2;
7:      $Candidate = \emptyset$;
8:      compute $N = M_{Motif}(C_c)$ via Eq. (5);
9:      if $N \geq 1$ then
10:         break;
11:     end if
12:     for each node $u \in N(C_c)$ do
13:         compute $N' = M_{Motif}(C_c)$ via Eq. (5) if node $u$
            is added into $C_c$;
14:         if $N - N' \geq 0$ then
15:             $Candidate = Candidate \cup \{u\}$;
16:         end if
17:     end for
18:     if $Candidate \neq \phi$ then
19:         traverse $Candidate$ and select the node that
            maximizes $\delta_c(v)$, denoted as $vBest$;
20:         $C_c = C_c \cup \{vBest\}$;
21:     else
22:         break;
23:     end if
24: end while
25: Return: $C_c$;
```

## C. COMMUNITY EXPANSION

The input of the community expansion method is the output of the core detecting method. Community expansion aims to expand the community with another fuzzy membership function. At this stage, the members of the local community will be roughly identified. Thus, the fuzzy membership function and community expansion method determine the efficiency of our local higher-order community detection algorithm.

At the community expansion stage, the number of internal motifs is greater than the number of external motifs, and the degree of node u belonging to the local community $C$ is measured by the fuzzy membership function, $\delta_e(u)$. The definition of $\delta_e(u)$ is given as follows.

$$\delta_e(u) = \begin{cases} \frac{\Delta m_{Out}}{d_M(u)}, & \Delta M(C) \geq 0; \\ 0, & \Delta M(C) < 0. \end{cases} \tag{9}$$

where $\Delta m_{Out} = m_{Out}^{\neg u} - m_{Out}^{u}$, $m_{Out}^{\neg u}$ is the number of external motifs of node $u$ in $C$ if node $u$ is not added into the local community $C$, and $m_{In}^{u}$ is the number of external motifs of node $u$ in $C$ if node $u$ is added into the local community $C$, whereas $d_M(u)$ is the motif degree of node $u$.

Equation (9) describes the potential external motif gain of node $u$. At this stage, we add the node with the largest value of $\delta_e(u)$ that also satisfies $\Delta M(C) \geq 0$ into local community $C$. We use an example to explain the potential external motif gain of a node and community expansion stage, which is shown in Fig. 5.

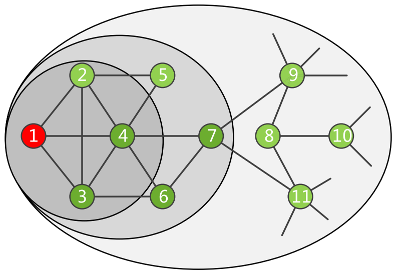


FIGURE 5. The community expansion stage.

In Fig. 5, $M_{In}(C) = 4$, $M_{Out}(C) = 3$, and $M_{Motif}(C) = 4/3$ according to Eq. (5); thus, the detection is at the community expansion stage. From Fig. 5, we observe that nodes 1, 2, 3 and 4 are in local community $C$ and nodes 5, 6, and 7 are the neighbor nodes of community $C$. We calculate $\Delta M(C)$ according to Eq. (6) if nodes 5, 6, and 7 are added into local community $C$ and if nodes 5 and 6 satisfy $\Delta M(C) \geq 0$. Then, we calculate $\delta_e(5)$ and $\delta_e(6)$ according to Eq. (9). Because $m_{Out}^{\neg 5} = 1$, $m_{Out}^{5} = 0$, $d_M(5) = 1$, $m_{Out}^{\neg 6} = 2$, $m_{Out}^{6} = 1$ and $d_M(6) = 2$, $\delta_e(5) = 1.0$ and $\delta_e(6) = 0.5$. Therefore, node 5 will be added into local community $C$ at this time step.

Moreover, adding a node into $C$ changes the values of $m_{Out}^{\neg u}$ and $m_{Out}^{u}$. Thus, $\delta_e(u)$ is also a fuzzy membership function. Algorithm 3 outlines the procedure to process community expansion based on the fuzzy membership function, $\delta_e(u)$.

**Algorithm 3** : CommunityExpansion

```
1:  Input: Graph $G = (V, E)$, motif $M$, $C_c$;
2:  Output: The roughly members of the target community;
3:  Procedure: CommunityExpansion($G, M, C_c$);
4:  Set $C_r = C_c$;
5:  while true do
6:      get $N(C_r)$ based on definition 2;
7:      $Candidate = \emptyset$;
8:      compute $N = M_{Motif}(C_r)$ via Eq. (5);
9:      for each node $u \in N(C_r)$ do
10:         compute $N' = M_{Motif}(C_r)$ via Eq. (5) if node $u$
            is added into $C_r$;
11:         if $N - N' \geq 0$ then
12:             $Candidate = Candidate \cup \{u\}$;
13:         end if
14:     end for
15:     if $Candidate \neq \phi$ then
16:         traverse $Candidate$ and select the node that
            maximizes $\delta_e(v)$, denoted as $vBest$;
17:         $C_r = C_r \cup \{vBest\}$;
18:     else
19:         break;
20:     end if
21: end while
22: Return: $C_r$;
```

## D. COMMUNITY OPTIMIZATION

The input of the community optimization method is the output of the community expansion method. Community optimization aims to add some fuzzy nodes to the local community with the third fuzzy membership function. At this stage, the fuzzy nodes of the local community will be fully considered. Thus, the fuzzy membership function and community optimization method determine the accuracy of our local higher-order community detection algorithm.

At this stage, the local community $C$ has been roughly identified, and metric $M_{Motif}(C)$ cannot be increased by adding any node into $C$. For fuzzy node $u$, we calculate the value of $\delta_o(u)$. If $\delta_o(u) \geq \lambda$ ($\lambda$ is the threshold of the structural similarity, $0 < \lambda < 1$), then the relationship between node $u$ and the local community $C$ is likely very close. The definition of $\delta_o(u)$ is given as follows:

$$\delta_o(u) = \frac{|N(u) \cap C|}{|N(u)|}. \tag{10}$$

Equation (10) describes the structural similarity between node $u$ and local community $C$. When they share more common nodes, their structural similarity is large. At this stage, we add the node with largest value of $\delta_o(u)$ into local community $C$. We use an example to explain the similarity between a node and a local community and the community optimization stage, which is shown in Fig. 6. We set the threshold of the structural similarity to $\lambda = 0.5$.

In Fig. 6, $M_{In}(C) = 6$, $M_{Out}(C) = 1$, and $M_{Motif}(C) = 6$ according to Eq. (5), and metric $M_{Motif}(C)$ cannot be increased by adding any node into $C$; thus, the detection is at the community optimization stage. From Fig. 6, we observe that nodes 1, 2, 3, 4, 5 and 6 are in local community $C$ and node 7 is the neighbor node of $C$. Then, we calculate $\delta_o(7)$ according to Eq. (10). Because $|N(7)| = 6$ and $|N(7) \cap C| = 3$, then, $\delta_o(7) = 0.5$. Therefore, node 7 will be added into local community $C$ at this time step.

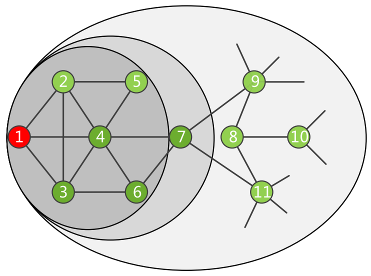


**FIGURE 6.** **The community optimization stage.**

Moreover, adding a node into $C$ changes the membership of the local community. Thus, $\delta_o(u)$ is also a fuzzy membership function. Algorithm 4 outlines the procedure to process community optimization based on the fuzzy membership function, $\delta_o(u)$.

**Algorithm 4** : CommunityOptimization

```
1:  Input: Graph $G = (V, E)$, motif $M$, $C_r$, $\lambda$;
2:  Output: The detected local community $C$;
3:  Procedure: CommunityOptimization($G, M, C_r, \lambda$);
4:  Set $C = C_r$;
5:  while true do
6:      get $N(C)$ based on definition 2;
7:      if $N(C) \neq \phi$ then
8:          traverse $N(C)$ and select the node that
            maximizes $\delta_o(v)$, denoted as $vBest$;
9:          if $\delta_o(vBest) \geq \lambda$ then
10:             $C = C \cup \{vBest\}$;
11:         else
12:             break;
13:         end if
14:     else
15:         break;
16:     end if
17: end while
18: Return: $C$;
```

## E. THE FUZLHOCD ALGORITHM

As mentioned above, the local higher-order community detection process is divided into three stages, and we use different fuzzy membership functions at different stages to detect the local community. Combining core detecting, community expansion and community optimization together, we propose a more effective approach, which is denoted as FuzLhocd, for local higher-order community detection. Algorithm 5 describes the process of these three stages. The first is the core detecting stage, which detects the core members of the local community based on the fuzzy membership function. Second, the community expansion stage uses another fuzzy membership function to expand which nodes should join into the local community. Third, the community optimization stage occurs in which the third fuzzy membership function is provided to add some fuzzy nodes to further improve the quality of the local community.

**Algorithm 5** : FuzLhocd

```
1:  Input: Graph $G = (V, E)$, motif $M$, seed node $s$, $\lambda$;
2:  Output: The detected local community $C$;
3:  Procedure: FuzLhocd($G, M, s, \lambda$);
4:  Set $C_c = \phi$, $C_r = \phi$, $C = \phi$;
5:  // Stage 1: Core Detecting
6:  $C_c$ = CoreDetecting($G, M, s$); Algorithm 2
7:  // Stage 2: Community Expansion
8:  $C_r$ = CommunityExpansion($G, M, C_c$); Algorithm 3
9:  // Stage 3: Community Optimization
10: $C$ = CommunityOptimization($G, M, C_r, \lambda$); Algorithm 4
11: Return: $C$;
```

## F. COMPLEXITY ANALYSIS

In this section, we analyze the computational complexity of the GloLhocd and FuzLhocd algorithms. These two methods are greed algorithms. GloLhocd use a global membership function during all the stages, and FuzLhocd use different fuzzy membership functions at different stages. Therefore, the time complexity of these algorithms is basically the same. Assuming that the identified local community has $n$ nodes and $m$ edges, the average degree of the nodes in the local community is $d$. To facilitate the analysis, we take the triangle as the motif. At time step 0, there is only the seed node in the local community, and these two algorithms traverse $d$ nodes to determine which node should be added into the local community. At time step $t$, there are $k$ nodes in the local community, and these two methods should traverse at most $kd$ nodes to determine which node should be added into the local community. In general, there are at most $nd$ nodes that should be traversed during the entire process. For each node, we need $O(d \log d)$ to compute its membership, and we need $O(m^{1.5})$ to compute the local motif modularity. Thus, the time complexity of our methods is $O\left(n^2d^2 \log d + dn^2m^{1.5}\right)$.

Currently, in local higher-order community detection, almost all methods need precomputes the number of motif instances containing each pair of nodes, which is a possibly large upfront time consuming, such as K-Truss [12] and MAPPR [25]. Assuming that the network has $N$ nodes and $M$ edges, the average degree of the nodes in network is $D$. At present the most effective solutions such as Latapy [25] requires $O(M^{1.5})$ to find triangle motifs in the network. So when the motif is a triangle, the time complexity of local higher-order community detection method is at least $O\left(NM^{1.5} \log D\right)$. In contrast, our methods only needs to search the limited motifs around the seed node. So when the local motifs grows more slowly than the whole network size, our method will definitely be more efficient than other methods.

## VI. EXPERIMENTS

In this section, we first give details of the strategy for evaluating the performance of the proposed methods. Then, we performed experiments to evaluate the effectiveness of the proposed methods using a variety of real-world networks. We also conducted experiments on different synthetic networks to show its scalability and the sensitivity to community structure. In the end, the parameter-tuning result and the case studies are analyzed.

## A. EXPERIMENTAL SETUP

### 1) EXPERIMENTAL PLATFORM

To simulate the performances of all algorithms on both real and synthetic graphs, all experiments are performed on a high performance server with 176G memory, 2x Intel Xeon 3.2GHz CPU, and Ubuntu OS.

### 2) COMPARISON ALGORITHMS

To evaluate the performance of our proposed algorithms, we select two local higher-order community detection algorithms in the state of the art to compare with our algorithms, as shown in Table 1. In which, K-Truss [12] and MAPPR [25] are considered as the standard algorithms for local higher-order community detection, we can download the implementations of C++ version from the website for the algorithms. In addition, our algorithms GloLhocd and FuzLhocd were implemented in Python. For all local higher-order community detection algorithms, unless otherwise stated, the recommended default parameter values were used to obtain the best experimental results. Specifically, K-Truss uses the trussness value ($k = 5$) as suggested by the authors. MAPPR uses the default teleportation parameter ($\alpha = 0.98$) and tolerance parameter ($\varepsilon = 10^{-4}$) as suggested by the authors. We set the threshold parameter to ($\lambda = 0.6$) for FuzLhocd as the default parameter.

**TABLE 1.** Comparison algorithms.

| Algorithm | Type | Implementation |
|---|---|---|
| K-Truss [12] | Higher-Order Algorithm | C++ |
| MAPPR [25] | Higher-Order Algorithm | C++ |
| GloLhocd | Higher-Order Algorithm | Python |
| FuzLhocd | Higher-Order Algorithm | Python |

### 3) EVALUATION METRICS

To extensively compare the community detection algorithms in terms of effectiveness, we adopt the following two quality measures:

Relative Density: The first metric is the relative density. The density is a popular local community detection fitness measure that takes both the inter and intra edges of a target community into consideration [11]–[13]. Intuitively, a target community with a high density indicates that the connectivity of the target community is high. The relative density of the target community is formally defined as follows, where $C'$ is the detected community, $|*|$ indicates the number of $|*|$ in the set, $E\left|C'\right|$ is the number of inter edges, and $E'\left|C'\right|$ is the number of intra edges.

$$density\left(C'\right) = \frac{\left|E\left(C'\right)\right|}{\left|E\left(C'\right)\right| + \left|E'\left(C'\right)\right|}. \tag{11}$$

F-Score: We also use the F-Score to quantify the performances of the identified community. F-Score is a commonly used criterion for local community detection algorithms when the community ground truth is known [16], [24], [32]. $C$ denotes the real community where the seed is located. $C'$ denotes the identified community. Recall is the number of correctly classified nodes divided by the number of nodes in $C$.

$$Recall\left(C'\right) = \frac{\left|C \cap C'\right|}{|C|}. \tag{12}$$

The precision is the number of correctly classified nodes divided by the number of the nodes in $C$'.

$$Precision\left(C'\right) = \frac{\left|C \cap C'\right|}{|C'|}. \tag{13}$$

The F-Score provides a real number that is between zero and one and combines recall and precision. A poorly performing community detection algorithm should be associated with a low F-Score. A higher F-Score value represents a better performing algorithm.

$$F - Score\left(C'\right) = \frac{2 * Recall(C') * Precision(C')}{Recall(C') + Precision(C')}. \tag{14}$$

## B. EXPERIMENTS ON REAL-WORLD NETWORKS

### 1) NETWORK DESCRIPTION

To evaluate the performance and effectiveness of various community detection algorithms, it is necessary to investigate them in real-world networks. Six commonly used large-scale real-world networks were used in the experiments, and the characteristics of the networks are listed in Table 2. These networks were selected since they are very well-known and contain the real structure of communities, which can be used to discuss the results of each algorithm with a desired accuracy. All chosen real-world networks are publicly available from the Stanford large network dataset collection (http://snap.stanford.edu/data/).

**TABLE 2.** The characteristic of commonly used real-world networks.

| Networks | Node | Edge | Communities |
|---|---|---|---|
| com-dblp | 317K | 1.04M | 13477 |
| com-amazon | 335K | 926K | 75149 |
| com-youtube | 1.13M | 2.98M | 8385 |
| com-livejournal | 3.99M | 34.68M | 287512 |
| com-orkut | 3.07M | 117.18M | 6288363 |
| com-friendster | 65.6M | 1806.06M | 957154 |

### 2) EFFECTIVENESS EVALUATION

We first evaluated the effectiveness of the selected methods on real networks. For each network, we use a triangle as a motif and randomly select nodes as seeds to examine 100 communities with more than 100 nodes. The details of the procedure of the seed selection strategy are as follows.

We randomly picked 2 sets of seeds with a size of 100. The first set of seeds was picked from a random ground-truth community with a lower node degree. Another set of seeds was picked from a random ground-truth community with a higher node degree. Then, we calculated the average values for the density and F-score for the two seed sets. The experimental results are presented in Fig. 7 and Fig. 8. The experimental results demonstrate that our algorithm performs better in terms of density and F-score precision with the ground truth community in these networks than earlier state-of-the-art algorithms, and its running time is also competitive.

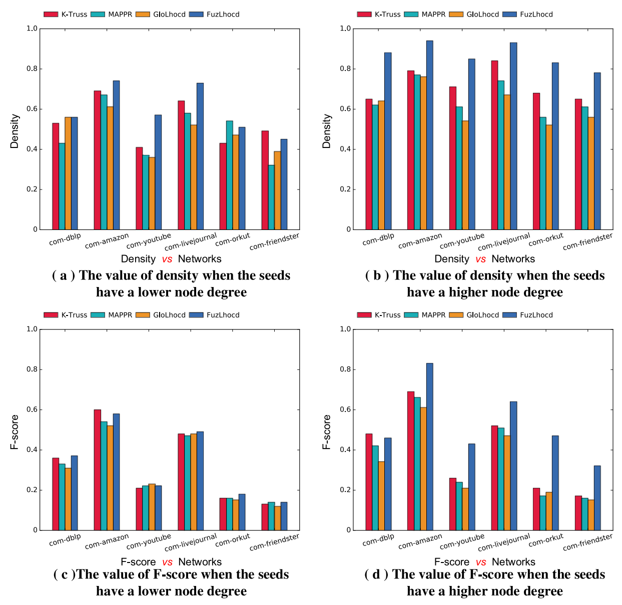


**FIGURE 7.** **Local community detection performances of various algorithms on real-world networks.**

Fig. 7 shows the two goodness metrics, density and F-score, are the selected methods on real-world networks. These metrics are normalized so that their maximum values equal to 1. Fig. 7(a) and Fig. 7(b) shows the density indices of the algorithms on real-world networks. Our method FuzLhocd achieves the highest density on most networks. K-Truss has the second best performance, which outperforms FuzLhocd on com-friendster network when the degree of seed was smaller. MAPPR and GloLhocd are does not perform well due to the seed-dependent problem. Fig. 7(c) and Fig. 7(d) shows the F-score of the detected communities using different methods. The FuzLhocd method has the best overall performance. We can see that the F-score value of FuzLhocd is about 20% to 50% higher than those of other methods. The seed-dependent problem causes the low F-score of other methods. This is why their F-score value is very low. Similar results can be observed in other datasets.

Fig. 8 shows the overall running time averaged over 100 random queries. Fig. 8 shows that FuzLhocd runs much faster than MAPPR and K-Truss, and is close to GloLhocd. The FuzLhocd method can process large graphs with millions of nodes in tens of seconds. Note that even though the heuristic local search method GloLhocd runs faster than FuzLhocd, but its accuracies is low. In addition, The MAPPR and K-Truss methods takes long running time, because both of them needs to precomputes the number of motif instances containing each pair of nodes, and this precomputation is time consuming.. The running time for FuzLocd almost monotonically increases with the growth of the network. This implies FuzLocd is quite effective when the network is large.

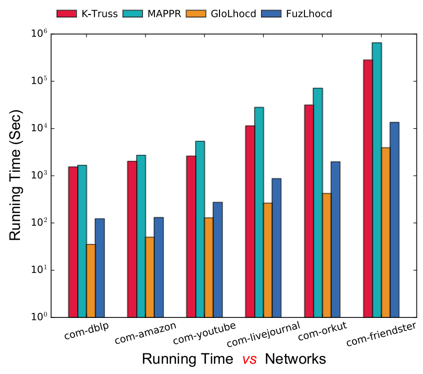


**FIGURE 8.** **The overall running time of different methods.**

## C. EXPERIMENTS ON SYNTHETIC NETWORKS

### 1) NETWORK GENERATION

In order to test the sensitivity-to-community-structure and scalability of the selected algorithms, we investigated the results on synthetic networks generated by Lancichinetti Fortunato Radicchi (LFR) benchmark [40]. The network generating model LFR($N$, $C$, $k$, $k_{\max}$, $\mu$, $\ldots$) has five important parameters, where $N$ is the number of nodes in the network, $C$ is the number of communities, $k$ is the average degree of the nodes, $k_{\max}$ is the maximum degree of nodes, $\mu$ is the mixing parameter indicating the proportion of a node's neighbors that reside in other communities. Generally, the higher the mixing parameter of a network is, the more difficult it is to identify the intrinsic communities. By varying the parameters of the LFR benchmark, we can analyze the behavior of the algorithms in detail. In this experiments, we generate eight large-scale synthetic networks with properties of ground-truth. The values of the parameters for generated networks are given in Table 3.

### 2) COMMUNITY STRUCTURE SENSITIVITY EVALUATION

Local community detection is sensitive to the clarity of community structures in the network. In general, the fuzzier the community structure, the more difficult local community detection. To verify this conjecture, we evaluated the local community detection performances of various algorithms on LFR synthetic networks. Clearly, by tuning $\mu$, we can vary the clearness of the community structure of the generated synthetic network.

**TABLE 3. Synthetic networks and parameters for the LFR benchmark.**

| Networks | $N$ | $C$ | $k$ | $k_{\max}$ | $\mu$ |
|---|---|---|---|---|---|
| LFR1 | 2M | 1000 | 10 | 20 | 0.2 |
| LFR2 | 4M | 1000 | 10 | 20 | 0.2 |
| LFR3 | 6M | 1000 | 10 | 20 | 0.2 |
| LFR4 | 8M | 1000 | 10 | 20 | 0.2 |
| LFR5 | 10M | 1000 | 10 | 20 | 0.2 |
| LFR6 | 12M | 1000 | 10 | 20 | 0.2 |
| LFR7 | 16M | 1000 | 10 | 20 | 0.2 |
| LFR8 | 20M | 1000 | 10 | 20 | 0.2 |

We first evaluate the sensitivity-to-community-structure when varying the mixing parameter $\mu$ from 0.1 to 0.8. The number of nodes in the network is 10M, the number of communities $C = 1000$, the average degree of nodes $k = 10$, and the maximum degree of nodes $k_{\max} = 20$. The seed selection strategy is the same as the experiments on real-world networks. Then, the average values for the density and F-score were calculated. Fig. 9 display density and F-score indices of the algorithms on artificial networks.

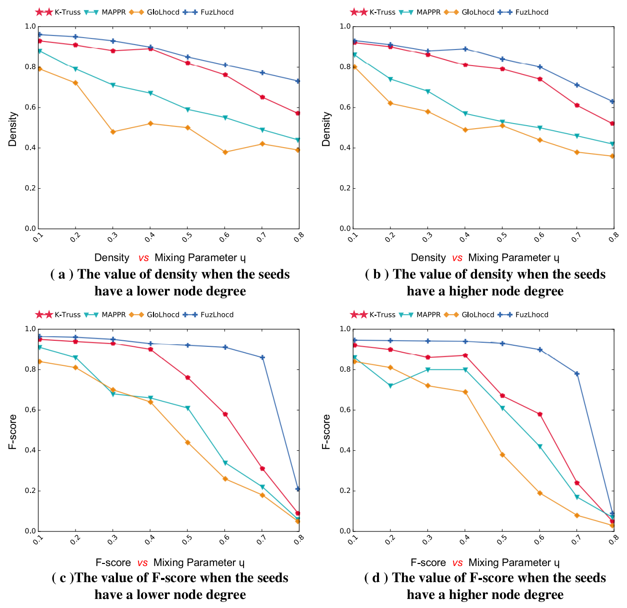


**FIGURE 9. Local community detection performances of various algorithms on LFR networks.**

Fig. 9(a) and Fig. 9(b) shows the density indices of the algorithms on artificial networks. As we can see, the density decreases when increasing $\mu$, i.e., lowering the clearness of the community structure. Moreover, it is clear that FuzLhocd and K-Truss are consistently and significantly better than GloLhocd and MAPPR for different $\mu$. FuzLhocd exhibits a similar performance as K-Truss, but is still better than K-Truss. Fig. 9(c) and Fig. 9(d) shows the F-score of the detected communities using different methods. We can see that the accuracies of all existing methods drop significantly after certain threshold. Because when $\mu$ is large, the boundary of the communities become vague. In addition, FuzLhocd performs better than other methods no matter taking lower degree nodes or higher degree nodes as seeds, causing by our method FuzLhocd is more robust to the seed-dependent problem than other algorithms. In summary, for synthetic networks, FuzLhocd performs well for detecting a local community with ground truth and is more robust to fuzzy structure networks than the other algorithms.

### 3) SCALABILITY EVALUATION

We then evaluate the scalability when varying the network size from 2M to 20M. Using the LFR model, we generated an ensemble of synthetic networks with various numbers of nodes from 2M to 20M. The other parameters, including community number, average degree, maximum degree and mixing ratio, were fixed at 1000, 10, 20 and 0.2, respectively. For each network we use triangle as a motif and randomly select nodes as seeds to examine 100 communities with more than 100 nodes; then, the average running time of these 100 seeds were calculated and normalized. The scalability results of all methods are shown in Fig. 10.

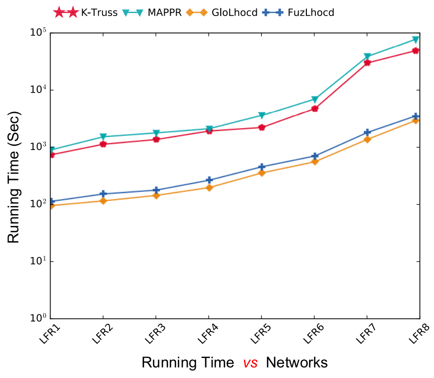


**FIGURE 10. Scalability of local community detection.**

As shown in Fig. 10, we found that FuzLhocd (or GloLhocd) are consistently more efficient than global search methods even on networks with millions of nodes. Interestingly, we observed that the running time of FuzLhocd is an average of 1∼10 times shorter than that of MAPPR. Moreover, we found that the running time gap between MAPPR (or K-Truss) and FuzLhocd (or GloLhocd) becomes larger as the nodes size increases. This is because MAPPR (or K-Truss) is a global method and needs to visit all nodes in the graph. In contrast, FuzLhocd (or GloLhocd) is a local method and only needs to visit limited neighborhoods around the seed. These observations verify that a local method is certainly more efficient than a global method. As the node size grows to a million, the running time of FuzLhocd may be a few orders of magnitude faster than MAPPR. Thus the FuzLhocd method runs efficiently.

## D. PARAMETER SENSITIVITY EVALUATION

The third objective of the experimental evaluation was to observe and validate the sensitivity of parameter $\lambda$ in the

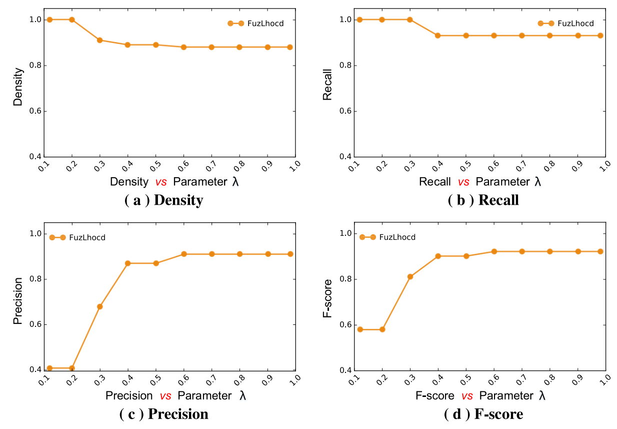


**FIGURE 11. Sensitivity of parameter λ for local community detection.**

FuzLhocd algorithm. Parameter λ was defined to determine which node should be added into the local community at the community optimization stage. We were interested in the changes in accuracies and community size with different values of λ. For this evaluation, we used the Polbooks network as the experimental dataset and observed the change in community structure upon gradually modifying the value of λ. Similar results were obtained on other networks.

We first analyzed the influence of λ on the accuracies. In general, it is expected that the accuracies depend on small changes to λ. The results are shown in Fig. 11, and they verify our conjecture. As shown in Fig. 11(a), the density is very large when $0 < \lambda \leq 0.2$. When $\lambda = 0.3$ and is larger, the density is almost stable. As shown in Fig. 11(b), the recall is very large when $0 < \lambda \leq 0.3$, When $\lambda = 0.4$ and is larger, the recall is almost stable. As shown in Fig. 11(c), $\lambda = 0.4$ is the critical point upon which stable precision is found. Afterwards, the precision is almost stable. As shown in Fig. 11(d), $\lambda = 0.4$ is the critical point upon which a stable F-score is found. Afterwards, the F-score is almost stable. Therefore, FuzLhocd yields a perfect partitioning with parameter λ over a long stable range (0.4∼1.0).

Then, we analyzed the influence of λ on community size. In general, it is expected that the community size monotonically decreases with λ. When λ is small, it is easy to find a large community. When λ grows, the community decreases. The results are shown in Fig. 12, and they verify our conjecture. As shown in Fig. 12(a), the community size monotonically decreases with λ. Moreover, $\lambda = 0.4$ is the critical point upon which a stable value for the community size was found. Before this, some border nodes could be added to the local community when λ is small. After this, irrelevant nodes quickly keep far away from the local community due to the strong constraint on the closeness of a community. The resulting community structures of the Polbooks network with respect to distinct parameters are further illustrated in Fig. 12(b) to Fig. 12(e), where all nodes with the same color belong to the same community. The red nodes denote the seed, and the green nodes denote the membership of the finding community.

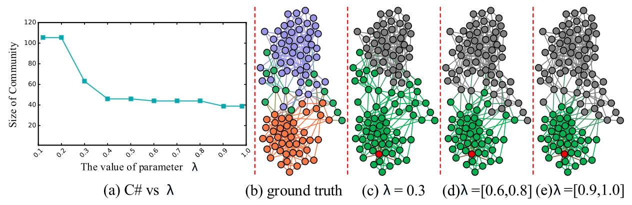


**FIGURE 12. Sensitivity of parameter λ for community size.**

Besides the above cases, we also investigate the influence of λ on the result of running time. In general, it is expected that the running time monotonically decreases with λ. When λ grows, the search space of local search becomes smaller. As a result, the running time of local search decreases. To verify this conjecture, we apply FuzLhocd with $\lambda = 0.1, 0.2, \ldots, 1.0$ on the synthetic networks and real networks. For each λ, we randomly selected 10 nodes as the query node from the network. Then, we averaged the running time of these 10 query nodes. The results are shown in Fig. 13 and they verify our conjecture. From Fig. 13, We can clearly see that with the increase of parameter λ, the running time gradually decreases. Moreover, when $\lambda = 0.5$ and larger, the running time is almost stable.

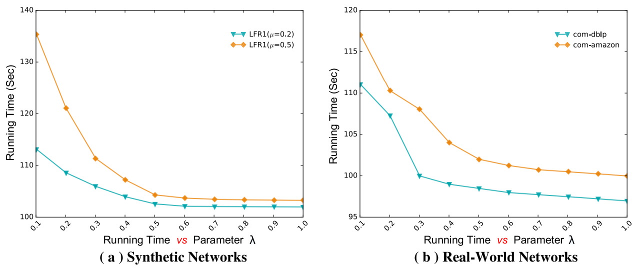


**FIGURE 13. The running time change with parameter λ in synthetic networks and real networks.**

From the result, we found that FuzLhocd is not sensitive to the search results by using different parameter λ parameters. In general, a λ value between 0.4 and 1.0 is sufficient to achieve a good result. We recommend a value of 0.6 for λ.

## E. CASE STUDIES

To evaluate the effectiveness of our local higher-order community detection methods, we selected three well-known real-world networks with ground truth, namely, Strike, Karate and Dolphins, for case studies. These real-world networks are publicly available from the UCI data repository at https://networkdata.ics.uci.edu/index.php. Additionally, we selected different special role nodes as seeds at different networks for case study. Specifically, we selected an outlier (sparsely connected nodes) node as the seed in the Strike network, a hub (bridging different communities) node as the seed in the Karate network, a core (higher centrality) node as the seed in the Dolphins network.

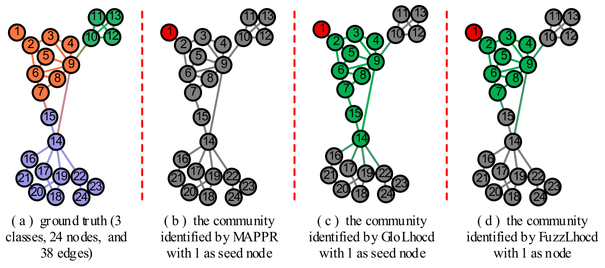

( a ) ground truth (3 classes, 24 nodes, and 38 edges)
( b ) the community identified by MAPPR with 1 as seed node
( c ) the community identified by GloLhocd with 1 as seed node
( d ) the community identified by FuzzLhocd with 1 as node

**FIGURE 14.** Case study on the Strike network.

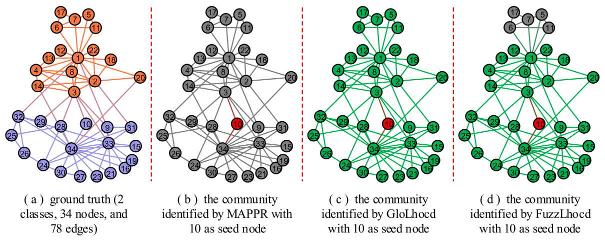

( a ) ground truth (2 classes, 34 nodes, and 78 edges)
( b ) the community identified by MAPPR with 10 as seed node
( c ) the community identified by GloLhocd with 10 as seed node
( d ) the community identified by FuzzLhocd with 10 as seed node

**FIGURE 15.** Case study on the Karate network.

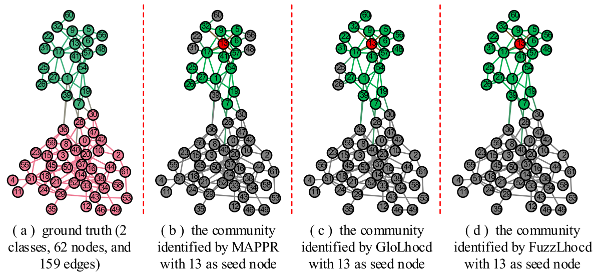

( a ) ground truth (2 classes, 62 nodes, and 159 edges)
( b ) the community identified by MAPPR with 13 as seed node
( c ) the community identified by GloLhocd with 13 as seed node
( d ) the community identified by FuzzLhocd with 13 as seed node

**FIGURE 16.** Case study on the Dolphins network.

The first network was the Strike network. In this case study, we used outlier node "1" as the seed and obtained the community result shown in Fig. 14. Fig. 14(a) shows the ground truth of the networks, where all nodes with the same color belong to the same community. Fig. 14(b) shows the detection results that were obtained by the MAPPR algorithm, which identified only the seed node in the local community. Fig. 14(c) shows the detection results that were obtained by the GloLhocd algorithm, which are denoted by green nodes. Fig. 14(d) shows the detection results that were obtained by the FuzLhocd algorithm, which are denoted by green nodes. Comparing Fig. 14(b) to Fig. 14(a), Fig. 14(c) to Fig. 14(a) and Fig. 14(d) to Fig. 13(a), FuzLhocd was found to perform better for identifying a ground-truth community when the outlier was the seed.

The second network was the Zachary's Karate club network. Here, we used hub node "10" as the seed and obtained the community result shown in Fig. 15. Fig. 15(a) shows the ground truth of the networks, where all nodes with the same color belong to the same community. Fig. 15(b) shows the detection results that were obtained by the MAPPR algorithm, which identified only the seed node in the local community. Fig. 15(c) shows the detection results that were obtained by the GloLhocd algorithm, which are denoted by green nodes. Fig. 15(d) shows the detection results that were obtained by the FuzLhocd algorithm, which are denoted by green nodes. Comparing Fig. 15(b) to Fig. 15(a), Fig. 15(c) to Fig. 15(a) and Fig. 15(d) to Fig. 15(a), we found that FuzLhocd performed better in identifying the ground-truth community when the hub node was used as the seed.

The last network was the Dolphins social network. Here, we used core node "13" as the seed and obtained the community result shown in Fig. 16. Fig. 16(a) shows the ground truth of the networks, where all nodes with the same color belong to the same community. Fig. 16(b) shows the detection results that were obtained by the MAPPR algorithm, which are denoted by green nodes. Fig. 16(c) shows the detection results that were obtained by the GloLhocd algorithm, which are denoted by green nodes. Fig. 16(d) shows the detection results that were obtained by the FuzLhocd algorithm, which are denoted by green nodes. Comparing Fig. 16(b) to Fig. 16(a), Fig. 16(c) to Fig. 16(a) and Fig. 16(d) to Fig. 16(a), the performances of three algorithms were found to be similar when the core node was used as the seed.

Based on the above three case studies, we make the following remarks. (1) The quality of the detected community by our FuzLhocd method does not depend on the location of the seed. (2) Our local FuzLhocd method is the most effective of the different networks.

## VII. CONCLUSION

Local community detection is a fundamental problem in complex network analysis and has attracted intensive research interest. However, most existing local community detection methods are based on a single node or edge, thereby ignoring the higher-order structures that are important for networks of a given domain. This paper proposes a local higher-order community detection algorithm (FuzLhocd) based on fuzzy membership functions. FuzLhocd is proposed to solve two common problems of existing local higher-order community detection algorithms: the global search problem and the seed-dependent problem. To solve the global search problem, this paper introduces a new local metric called local motif modularity. FuzLhocd only needs to visit limited neighborhoods around the seed based on this local metric. To solve the seed-dependent problem, we divide the process of the local community detection into three stages and employ the different fuzzy membership functions at different stages to detect the local community. We conducted extensive experiments on both real-world and synthetic networks, and the results demonstrate that FuzLhocd not only runs efficiently locally but also effectively solves the seed-dependent problem and achieves a high accuracy.

The methods in this paper are effective for static networks. However, complex networks in the real world change dynamically over time, and their community structures are dynamically updated. In the face of dynamic networks with complex changes, new fuzzy membership functions should be designed. In future work, the local community detection algorithms of dynamic networks will be further studied.

## ACKNOWLEDGMENT

The authors would like to thank the experimental equipments provided by National Super Computing Center of Changsha, located in Hunan province of China.

## REFERENCES

[1] A. Lancichinetti and S. Fortunato, "Community detection algorithms: A comparative analysis," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 80, pp. 056–117, Nov. 2009.

[2] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, nos. 3–5, pp. 75–174, Feb. 2010.

[3] J. Luo, P. Ding, L. Cheng, and X. Chen, "Semi-supervised prediction of human miRNA-disease association based on graph regularization framework in heterogeneous networks," *Neurocomputing*, vol. 294, pp. 29–38, Jun. 2018.

[4] L. Qin, G. Wang, L. Feng, S. Yang, and W. Jie, "Preserving privacy with probabilistic indistinguishability in weighted social networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 5, pp. 1417–1429, May 2017.

[5] M. E. Newman, "Fast algorithm for detecting community structure in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 69, Jun. 2004, Art. no. 066133.

[6] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 76, pp. 36–106, Sep. 2007.

[7] J. Chen, K. Li, K. Bilal, A. A. Metwally, K. Li, and P. Yu, "Parallel protein community detection in large-scale PPI networks based on multi-source learning," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, to be published.

[8] A. Clauset, "Finding local community structure in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 72, Aug. 2005, Art. no. 026132.

[9] M. Sozio and A. Gionis, "The community-search problem and how to plan a successful cocktail party," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2010, pp. 939–948.

[10] W. Cui, Y. Xiao, H. Wang, and W. Wang, "Local search of communities in large graphs," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2014, pp. 991–1002.

[11] W. Cui, Y. Xiao, H. Wang, Y. Lu, and W. Wang, "Online search of overlapping communities," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2013, pp. 277–288.

[12] X. Huang, H. Cheng, L. Qin, W. Tian, and J. X. Yu, "Querying k-truss community in large and dynamic graphs," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Jun. 2014, pp. 1311–1322.

[13] Y. Wu, R. Jin, J. Li, and X. Zhang, "Robust local community detection: On free rider effect and its elimination," *Proc. VLDB Endowment*, vol. 8, no. 7, pp. 798–809, Feb. 2015.

[14] Y. Fang, R. Cheng, S. Luo, and J. Hu, "Effective community search for large attributed graphs," *Proc. VLDB Endowment*, vol. 9, no. 12, pp. 1233–1244, Aug. 2016.

[15] Y. Bian, J. Ni, W. Cheng, and X. Zhang, "Many heads are better than one: Local community detection by the multi-walker chain," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2017, pp. 21–30.

[16] W. Luo, D. Zhang, H. Jiang, L. Ni, and Y. Hu, "Local community detection with the dynamic membership function," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 5, pp. 3136–3150, Oct. 2018.

[17] Q. Chen, T.-T. Wu, and M. Fang, "Detecting local community structures in complex networks based on local degree central nodes," *Phys. A, Stat. Mech. Appl.*, vol. 392, no. 3, pp. 529–537, Feb. 2013.

[18] F. Luo, J. Z. Wang, and E. Promislow, "Exploring local community structures in large networks," *Web Intell. Agent Syst., Int. J.*, vol. 6, no. 4, pp. 387–400, 2008.

[19] F. Luo, Y. Yang, C.-F. Chen, R. Chang, J. Zhou, and R. H. Scheuermann, "Modular organization of protein interaction networks," *Bioinformatics*, vol. 23, pp. 207–214, Jan. 2006.

[20] A. R. Benson, D. F. Gleich, and J. Leskovec, "Tensor spectral clustering for partitioning higher-order network structures," in *Proc. SIAM Int. Conf. Data Mining*, Jun. 2015, pp. 118–126.

[21] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Science*, vol. 353, no. 6295, pp. 163–166, 2016.

[22] J. Luo, L. Ding, C. Liang, and N. H. Tu, "An efficient network motif discovery approach for co-regulatory networks," *IEEE Access*, vol. 6, pp. 14151–14158, 2018.

[23] C. Liang, Y. Li, J. Luo, and Z. Zhang, "A novel motif-discovery algorithm to identify co-regulatory motifs in large transcription factor and microrna co-regulatory networks in human," *Bioinformatics*, vol. 31, no. 14, pp. 2348–2355, Jul. 2015.

[24] R. Andersen, F. Chung, and K. Lang, "Local graph partitioning using pagerank vectors," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci.*, Oct. 2006, pp. 475–486.

[25] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 555–564.

[26] M. E. J. Newman, "Modularity and community structure in networks," *Proc. Nat. Acad. Sci. USA*, vol. 103, no. 23, pp. 8577–8582, Jun. 2006.

[27] X. Xu, N. Yuruk, Z. Feng, and T. A. Schweiger, "Scan: A structural clustering algorithm for networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2007, pp. 824–833.

[28] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Exp.*, vol. 10, Oct. 2008, Art. no. P10008.

[29] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proc. Nat. Acad. Sci. USA*, vol. 105, no. 2, pp. 1118–1123, Jan. 2008.

[30] J. Shao, Z. Han, Q. Yang, and T. Zhou, "Community detection based on distance dynamics," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2015, pp. 1075–1084.

[31] C. Böhm, C. Plant, J. Shao, Q. Yang, "Clustering by synchronization," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2010, pp. 583–592.

[32] X. Ding, J. Zhang, and J. Yang, "A robust two-stage algorithm for local community detection," *Knowl.-Based Syst.*, vol. 152, pp. 188–199, Jul. 2018.

[33] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller. II," *IEEE Trans. Syst., Man, Cybern.*, vol. 20, no. 2, pp. 404–418, Mar./Apr. 1990.

[34] S. Wu and M. J. Er, "Dynamic fuzzy neural networks-a novel approach to function approximation," *IEEE Trans. Syst., Man, Cybern. B. Cybern.*, vol. 30, no. 2, pp. 358–364, Apr. 2000.

[35] M. Cerrada, J. Aguilar, E. Colina, and A. Titli, "Dynamical membership functions: An approach for adaptive fuzzy modelling," *Fuzzy Sets Syst.*, vol. 152, no. 3, pp. 513–533, Jun. 2005.

[36] P. G. Sun, "Community detection by fuzzy clustering," *Phys. A, Stat. Mech. Appl.*, vol. 419, pp. 408–416, Feb. 2015.

[37] L. Hu and K. C. Chan, "Fuzzy clustering in a complex network based on content relevance and link structures," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 2, pp. 456–470, Apr. 2016.

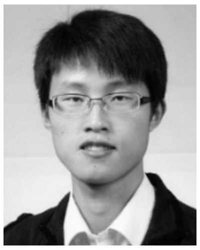
[38] H. Zhang, X. Chen, J. Li, and B. Zhou, "Fuzzy community detection via modularity guided membership-degree propagation," *Pattern Recognit. Lett.*, vol. 70, pp. 66–72, Jan. 2016.

[39] A. Biswas and B. Biswas, "FuzAg: Fuzzy agglomerative community detection by exploring the notion of self-membership," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 5, pp. 2568–2577, Oct. 2018.

[40] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 78, Oct. 2008, Art. no. 046110.

**TAO MENG** is currently pursuing the Ph.D. degree with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. His research interests include date mining and network analysis.

**LIJUN CAI** received the Ph.D. degree from the College of Computer Science and Electronic Engineering, Hunan University, in 2007, where he is currently a Professor. His research interests include bioinformatics, cloud computing, and big data scheduling and management.



**TINGQIN HE** received the M.S. degree from the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China, where he is currently pursuing the Ph.D. degree with the College of Information Science and Engineering. His research interests include date mining, cloud computing, and big data analysis.

**LEI CHEN** received the Ph.D. degree from Hunan University, in 2017. He is currently a Lecturer with the Hunan University of Science and Technology. His research interests include date mining, Web mining, graph mining, cloud computing, and big data scheduling and analysis.

**ZIYUN DENG** received the Ph.D. degree from the College of Electrical and Information Engineering, Hunan University, in 2016. He is currently a Professor with the Changsha Commerce and Tourism College. His research interests include high performance computing and logistics information technology.