A Novel Local Community Detection Method using Evolutionary Computation

Chao Lyu, Yuhui Shi, *Fellow, IEEE,* and Lijun Sun

***Abstract*—Local community detection is a significant branch of the community detection problems. It aims at finding the local community to which a given starting node belongs. Local community detection plays an important role in analyzing complex networks and recently has drawn much attention from researchers. In the past few years, several local community detection algorithms have been proposed. However, previous methods only make use of the limited local information of networks but overlook the other valuable information. In this paper, we propose an evolutionary computation based algorithm called evolutionary based local community detection (ELCD) algorithm to detect local communities in complex networks by taking advantages of the whole obtained information. The performance of the proposed algorithm is evaluated on both synthetic and real-world benchmark networks. Experimental results show that the proposed algorithm has a superior performance compared with the state-of-the-art local community detection methods. Furthermore, we test the proposed algorithm on incomplete real-world networks to show its effectiveness on networks whose global information can not be obtained.**

***Index Terms*—Community Detection, Evolutionary Computation, Local Community Detection.**

I. INTRODUCTION

THE study of complex networks attracts increasing attention from researchers in various domains since the network is naturally an effective formalism in representing relationships among objectives in real-world complex systems [1]. Out of so many directions in network science, the community structure detection has become a hot subject topic in recent years. In general, a network community is defined as a group of nodes with dense intra-connections and sparse interconnections with nodes outside of it, while the goal of the community detection is to divide a network into several communities [2]. As community structures play a vital role in the topology analysis of real-world complex networks, lots of community detection methods have been proposed [3]–[8]. Among them, the modularity maximization based methods [5] are becoming more and more popular. Because in this kind of methods, the community detection can be formalized as an optimization problem through the definition of the modularity and effectively solved by optimization algorithms, such as the powerful evolutionary computation inspired by the process of natural evolution [9]. Many evolutionary algorithms (EAs), such as genetic algorithm (GA) [10], particle swarm optimization (PSO) [11], ant colony optimization (ACO) [12], firefly algorithm (FA) [13], and bat algorithm (BA) [14], have been used to detect communities in complex networks and shown competitive performances.

The community detection mentioned above aims at finding the global community structures in networks by discovering all the hidden communities. However, in some cases, we only care about the community to which a specific node belongs. For instance, in the recommended systems of social softwares, if we want to recommend potential friends to a specific user, we only need to discover the community to which this person belongs. Even if traditional community detection is able to solve this problem by uncovering the global community structure of the social network and outputting a specific local community, it undoubtedly leads to a waste of computing resources. Moreover, sometimes the entire structure of a real-world network is difficult or no way to obtain, especially when its scale is extremely large. So, in this case, it is unrealistic to find all the communities of the network by using traditional community detection algorithms.

However, the local community detection, which aims at detecting the local communities in complex networks, is able to fill the above gaps and has become an important research topic. Local community is defined as the community to which a given starting node belongs, and where nodes in the local community are tightly connected while the connections between nodes in and out of the local community are sparse. Some works have been done on the local community detection and several methods have been proposed [15]–[26]. The basic strategy of the existing local community detection methods is generating the local community from the given starting node by successively adding appropriate nodes. However, although the local community detection requires to use the local information of networks, the existing methods only pay attention to very limited information around the starting node and overlook much valuable information that has been obtained, which frequently leads to a poor detection performance. To overcome this defect and improve the detection performance, it is necessary to utilize the whole obtained information of networks for local community detection, despite the fact that sometimes the global information of an entire network can't be obtained.

Based on the above idea, in this paper, we propose a novel method called evolutionary based local community detection

This work was partially supported by the National Key Research and Development Program of China under Grant NO. 2017YFC0804002, National Science Foundation of China under grant number 61761136008, Shenzhen Peacock Plan under Grant No. KQTD2016112514355531, Program for Guangdong Introducing Innovative and Entrepreneurial Teams under grant number 2017ZT07X386. ($Corresponding\ author:\ Yuhui\ Shi.$)

Chao Lyu, Yuhui Shi, and Lijun Sun are with Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, Guangdong, China. (E-mail: 11849557@mail.sustech.edu.cn, shiyh@sustech.edu.cn, sunlijun522@126.com).

(ELCD) algorithm to detect local communities in complex networks. ELCD takes full advantages of the obtained information of a network and detects local communities by evolutionary computation which has not been used in the local community detection problem to the best of our knowledge. ELCD is essentially a modularity optimization based method, in which the classic modularity concept is modified to adjust its application from the global community detection to the local community detection. As for the optimizer, ELCD adopts the binary particle swarm optimization (BPSO) which searches in the search space defined on networks and outputs the detected local communities.

The main contributions of this paper can be summarized as follows:

- This paper applies the evolutionary computation to the local community detection problem for the first time and thus expands EAs' applications in the community detection problems.
- This paper breaks the normal strategy adopted by previous methods and proposes a novel local community detection algorithm which overcomes the defects of the existing methods by utilizing the whole obtained information of networks.
- This paper demonstrates the effectiveness of the proposed algorithm by conducting experiments on both synthetic and real-world benchmark networks. Furthermore, the proposed algorithm is tested on the incomplete versions of real-world networks to show its effectiveness when the global information of an entire network can't be obtained.

The rest of this paper is organized as follows. In Section II, related work in the field of the local community detection is introduced. Then the proposed algorithm ELCD is presented in detail in Section III. In Section IV, ELCD is tested on benchmarks and compared with some existing methods. Section V summarizes this paper and gives the conclusion.

II. BACKGROUND

A. Related Work

Some local community detection methods have been proposed by researchers in the past few years. In this section, we review several typical algorithms.

Bagrow and Bollt proposed the L-shell method [26] to detect local communities in networks. Firstly, L-shell defines the concept of shell as the cluster of nodes which have specific distances to the given starting node. Then it successively adds nodes in the shells to the local community from the close to the distant, and calculates the following metric for each added shell:

$$C = \frac{s_{in}}{s_{out}} \tag{1}$$

where s_{in} and s_{out} are the number of edges connecting nodes in the shell with nodes inside and outside the current local community, respectively. The local community stops to expand if C exceeds a predefined threshold.

Bridge bounding is another local community detection algorithm proposed by Papadopoulos et al. [16] that is similar to L-shell, but the difference is that Bridge bounding method terminates the expansion of a local community when it meets the bridge bounding which can be defined by various metrics, such as the edge betweenness and edge-clustering coefficient.

In addition, the local modularity optimization is also a class of methods which detect local communities by optimizing the predefined local modularity. In this kind of methods, a local community begins with a starting node and grows by iteratively absorbing the adjacent nodes which make the local modularity increase most or decrease least. Clauset proposed a local modularity R [22] by dividing a local community into two parts: the core part and the boundary part, R is defined as follows:

$$R = \frac{e_{BB} + e_{Bcore}}{e_{BB} + e_{Bcore} + e_{out}} \tag{2}$$

where the numerator represents the number of internal edges that are associated with boundary nodes and the denominator represents the total number of edges associated with boundary nodes. We call this local community detection method as R method.

Besides, another typical local modularity called M is proposed by Luo et al. [20], which is defined as follows:

$$M = \frac{e_{in}}{e_{out}} \tag{3}$$

where e_{in} is the number of edges connecting two nodes within the local community and e_{out} is the number of edges connecting a node in the local community to an outside node. We call this local community detection method as M method.

In fact, Chen et al. indicate that the metrics M and R are equivalent if the local community has no core nodes [25]. However, the M and R are essentially greedy expansion-based methods, which have two main defects. One is that they are sensitive to the position of the starting node in the local community it belongs to. To be specific, the algorithms work well if the starting node locates in the central part of the local community. However, once the starting node locates in the boundary part, it is difficult for the algorithms to discover the entire local community. The other one is that the greedy optimization strategy they adopt may decrease the detection accuracy. As a result, some algorithms have been proposed to overcome these defects.

Chen et al. [17] proposed a local community detection method which is not sensitive to the position of the starting node. In this method, the local community is not discovered from the given starting node, but from the local degree central node that is associated with the starting node. Based on the above idea, three algorithms: LMD_M, LMD_R, and LMD_F have been proposed in [17]. Ding et al. [27] inherited this idea and proposed a two-stage local community detection algorithm called RTLCD. To solve the low accuracy problem caused by the greedy optimization strategy, Chen et al. [25] adopted the density metric L as the local modularity metric and proposed a novel method to optimize it instead of simply maximizing it in a greedy manner.

Furthermore, some researchers have introduced novel ideas into this problem. Based on the understanding that elements in the same community are more likely to share common links, Wu et al. [23] proposed a method to find local community

structures by analyzing the link similarity between the community and the node. Based on the random walk mechanism, Bian et al. [28] studied a multiwalker chain (MWC) model, which allows multiple walkers to explore the network to find local communities. Yao et al. [29] proposed an algorithm called VI which can discovery multiple levels and scales of local communities according to users' requirements. Based on the dynamic membership function, Luo et al. [24] proposed two local community detection algorithms: DMF_M and DMF_R by considering the characteristics of the local community during its formation process. These two algorithms divide the detection process into three stages and employ different dynamic membership functions for each stage to find local communities.

B. Preliminaries for Evolutionary Computation and Community Detection

Evolutionary computation (EC) is a powerful technique to solve optimization problems that can not be effectively addressed by classical optimization methods (e.g., steepest descent method and derivative method). In EC, an optimization problem is formalized as a vector x with d variables, that is, $x = [x_1, x_2, ..., x_d]$, where x is called a *candidate solution*, and $x_i,\ i \in \{1, 2, ...d\}$ is called a *decision variable*. A candidate solution represents a solution for the optimization problem and a decision variable represents a parameter whose value should be optimized by decision makers. d is called the *dimensionality* of this optimization problem and determines the complexity of the problem. The space constructed by all the feasible candidate solutions is called the *decision space* (or *search space*) of the optimization problem. Further, to evaluate the quantity of a solution, a *fitness function* should be defined and calculated on each candidate solution. The fitness function represents the optimization objective, the higher its value is, the better the candidate solution is. Since EC is inspired by the process of natural evolution and solves an optimization problem through the population-based search, in an EA, a *population* p is composed of n *individuals*, i.e., $p = \{p_1, p_2, ..., p_n\}$, where n is called the *population size*. Each individual represents a candidate solution and is responsible for searching in the decision space. Also, like classical optimization algorithms, EC adopts the iterative mechanism to simulate the evolution process in natural world, which means the population is updated in each iterative loop, until the desirable solution is found.

As for the community detection, it can be treated as a single objective optimization problem. Let $\mathcal{P} = \{\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_r\}$ be a community partition of a network, where each $\mathcal{C}_i,\ i \in \{1, 2, ..., r\}$ represents a community, r is the total number of communities in the network. The objective of the community detection is to find a best partition $\mathcal{P}^*$ for which

$$\mathcal{F}(\mathcal{P}^*) = max\ \mathcal{F}(\mathcal{P}),\ subject\ to\ \mathcal{P} \in \Omega \tag{4}$$

where Ω is the set of all the feasible community partitions of a network and $\mathcal{F} : \mathcal{P} \rightarrow \mathcal{R}$ determines the quantity of the community partition. If we use the EC framework to solve this optimization problem, the community partition $\mathcal{P}$ is represented by an individual and is encoded as a vector in the decision space Ω. The map $\mathcal{F}$ is generally defined by the modularity [30] of a network and employed as the fitness function. Through this formalization, an EA is able to solve this optimization problem by searching in Ω and outputting the desirable solution (a partition of communities).

Specially, for the local community detection problem, we want to divide a network into two parts: the cluster of nodes belonging to the community of the given starting node and the other nodes in the network, that is, $r = 2$ in local community detection. $\mathcal{C}_1$ and $\mathcal{C}_2$ represent the local community of the starting node and the other nodes in the network, respectively. If an EA is employed to solve this problem, $\mathcal{C}_1$ is represented by a candidate solution and the fitness function is defined by the proposed local modularity.

III. PROPOSED ALGORITHM

In this section, the proposed ELCD algorithm is introduced. Firstly we give the entire process of ELCD, then we give the details for its implementation.

A. General Introduction of ELCD

The entire process of ELCD is summarized in Algorithm 1 and the detailed description of which will be given in the following subsections.

Algorithm 1 ELCD Algorithm

```
Input Network G.
Input the given starting node v_i.
Calculate the diameter D of G.
Calculate the distances between each node in G and v_i.
Define the initial search space of v_i.
while the stopping criterion for local recursive partitioning (LRP) scheme is not met do
    Generate an initial population.
    Generate a detected local community to which v_i belongs by optimizing the local modularity by BPSO.
    Modify the local community by performing the supervised moving scheme (SMS) and the unsupervised moving scheme (UMS) operations.
    Update the search space.
end while
Output the detected local community to which v_i belongs.
```

B. Binary Particle Swarm Optimization

Particle swarm optimization (PSO) is a classical population-based EA [31], which is inspired by the flocking of birds around food resources. In PSO, the population consists of several particles, each of which is characterized by its position and velocity and moves in the search space to find the best solution. The current position of each particle is treated as a candidate solution which can be calculated based on its previous position and current velocity. The position and

velocity vectors of each particle are updated during iterations according to the following formula:

$$\begin{aligned} v_i(t) =& \omega v_i(t-1) + c_1\varphi_1[p_{ibest} - x_i(t-1)] \\ &+ c_2\varphi_2[p_{gbest} - x_i(t-1)] \\ x_i(t) =& x_i(t-1) + v_i(t) \end{aligned} \tag{5}$$

where $v_i(t)$ and $x_i(t)$ are the velocity and position vectors of the i-th particle at the t-th generation, respectively. p_{ibest} is the best position of the i-th particle found so far, p_{gbest} is the best position of the whole individuals found so far, ω is the inertia weight, φ_1 and φ_2 are two random values between 0 and 1, and c_1, c_2 are two learning factors. It should be noted that ω, $c1$, and $c2$ are three hyper-parameters which should be set in advance.

As it is known, this type of PSO can only address continuous optimization problems. However, discovering local communities in networks is an intrinsic discrete optimization problem with binary search spaces, which therefore requires a binary version of PSO (BPSO). Therefore, we employ a version of BPSO which was proposed in [32] as the core optimizer of ELCD, in which the position vectors of particles can only take values of 0 or 1. To be specific, after updating the velocity vector of a particle, each real element in the velocity vector will be converted to a probability in the rang of [0,1] by the following transfer function:

$$T(v_i^k(t)) = \begin{cases} 1 - \frac{2}{1+e^{-v_i^k(t)}} & \text{if } v_i^k(t) < 0 \\ \frac{2}{1+e^{-v_i^k(t)}} - 1 & \text{otherwise} \end{cases} \tag{6}$$

where $v_i^k(t)$ represents the k-th dimension of the velocity vector of the particle i at iteration t. After this conversion, each element in the position vector is updated according to the following formula:

$$x_i^k(t) = \begin{cases} 0 & \text{if } v_i^k(t) < 0 \text{ and } \mathrm{rand} < T(v_i^k(t)) \\ 1 & \text{if } v_i^k(t) \geq 0 \text{ and } \mathrm{rand} < T(v_i^k(t)) \\ x_i^k(t-1) & \text{otherwise} \end{cases} \tag{7}$$

where $x_i^k(t)$ represents the k-th dimension of the position vector of the particle i at iteration t and *rand* is a randomly generated number.

The process of the BPSO employed by ELCD is summarized in Algorithm 2.

C. Individual Representation

We propose a binary label-based representation scheme to represent the individuals of ELCD. Since the objective of the local community detection is to find the community to which a given starting node belongs, the whole nodes in the network should be divided into two groups: the nodes which belong to the local community of the starting node and the nodes which do not. Therefore, in our representation scheme, an individual x is a vector with d dimensions (i.e., d decision variables):

$$x = [x_1, x_2, ..., x_d], \tag{8}$$

where d is the number of nodes in the search space, and $x_i, i \in \{1, 2, ..., d\}$ represents the node's membership to the local community by its value being 1 or 0. To be specific, $x_i = 1$ means the i-th node in the search space belongs to the local community while $x_i = 0$ means it dose not. In this way, all the nodes in the search space are classified into two clusters and the local community can be detected in a straightforward way. An example is given in Fig. 1 to illustrate this representation scheme, from which the representation of the individual indicating the local community of node 1, 2, 3, and 4 can be written as:

$$x = [1, 1, 1, 1, 0, 0, 0, 0, 0, 0], \tag{9}$$

where the sequence of the decision variables in x is in accordance with the indexes of nodes in Fig. 1.

Algorithm 2 BPSO

```
Initialize the population p.
Initialize the global best position p_gbest and the individual best position p_ibest.
t = 1.
while t < t_max do
    for each particle p_i do
        Update the velocity v_i(t) according to (5).
        for each decision variable v_i^k(t) do
            Convert the v_i^k(t) into the probability T(v_i^k(t)) according to the transfer function (6).
            Update the x_i^k(t) according to (7).
        end for
    end for
    Calculate the fitness value for each particle.
    Update the individual best position p_ibest for each particle and the global best position p_gbest.
    t = t + 1.
end while
Output the global best solution p_gbest.
```

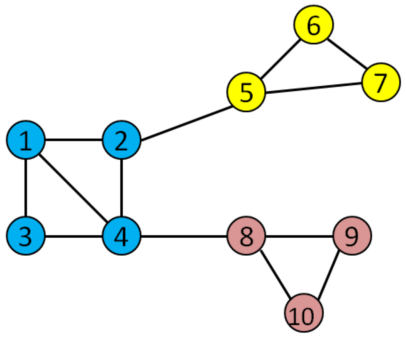


Fig. 1: The community partition of a network.

It should be noted that the previously proposed representation schemes used in EA-based community detection algorithms are all redundant [1]. However, this problem does not exist in the representation scheme proposed here, where one local community can be uniquely represented by one solution. That is to say, in ELCD, the correspondence between local community structures and candidate solutions is one-to-one.

D. Fitness Evaluation

Here we define a novel local modularity Q_l which comes from the concept of the modularity and is employed as the fitness function in ELCD. First, the modularity Q is proposed by Newman and Girvan [30] as the metric to measure the quality of community structures in complex networks, and Q is defined as follows [8]:

$$Q = \sum_{i=1}^{k} [\frac{e_i}{S} - (\frac{d_i}{2S})^2] \tag{10}$$

where i is the community index, k is the total number of communities in the network, S is the total number of edges in the network, e_i is the number of edges within community i, and d_i is the sum of the degrees of all nodes in community i.

The concept of modularity has been adopted by many EA-based community detection algorithms [5], [8], [33] and regarded as a very efficient quality metric for estimating the partitioning of a network into communities. It is not hard to find that Q is essentially a sum of the metrics evaluated on each community in the network, which means that the quantity of a single community can be measured by Q_l as follows:

$$Q_l = \frac{e_l}{S} - (\frac{d_l}{2S})^2 \tag{11}$$

where e_l is the number of edges within the local community and d_l is the sum of the degrees of all nodes in the local community. Therefore, in ELCD, we employ the proposed Q_l as the fitness function to measure the quantity of an individual.

Taking the network in Fig. 1 as an example, the Q_l for the local community with nodes 1, 2, 3, and 4 can be calculated as follows: First, in this community, there are 5 internal edges and 4 nodes with the degrees of 3, 3, 2, and 4, respectively, so we have $e_l = 5$ and $d_l = 12$. Then it can be seen that this network has 13 edges in total, so we have $S = 13$. Finally we can calculate $Q_l = 0.1716$ for this local community according to (11).

However, it should be noted that, because of the resolution limit (RL) problem [34], the modularity may bias towards network partitions with small communities merged into larger communities. As a result, the proposed local modularity Q_l will still suffer from this problem. In order to address this defect, a local recursive partitioning (LRP) scheme will be employed to increase the resolution of ELCD for detecting small scale local communities, which will be introduced later.

E. Definition of the Search Space

The search space of ELCD is a cluster of nodes corresponding to the decision variables of the EA which should be defined before local community detection process. For small scale networks, the search space can be the whole nodes in the network. However, for large scale networks with a great deal of nodes, detecting a local community by searching all the nodes will consume too much computing resource and is thus inefficient. Therefore, we propose the following method to define the search space of ELCD according to the nodes' distances to the given stating node.

In network G, the distance d_{ij} between node i and node j is defined as the shortest path from i to j, and the maximum distance between all pairs of nodes in G is defined as the diameter D of G. That is:

$$D = \max(d_{ij}),\ i, j \in \{1, 2, ..., m\}, \tag{12}$$

where m is the total number of nodes in the network.

Assume that j is the given starting node. Intuitively, nodes with relatively long distances to j will have little opportunities to join in the local community of node j. On the contrary, nodes with relatively short distances to j are more likely to be a member of the local community. Hence, we introduce a parameter λ which is set in advance to define the search space. To be specific, when detecting the local community of node j, the search space is defined as the cluster of nodes whose indexes i satisfy:

$$i \in \{1, 2, ..., m\},\ \text{ subject to } d_{ij} \leq \lambda D. \tag{13}$$

It can been seen that by using this definition, the search scope of ELCD can be greatly narrowed down when detecting local communities in large scale networks and thus the detection efficiency can be improved.

F. Population Initialization

In previous EA-based community detection algorithms, to keep the population's diversity, the randomization is widely used in the initialization process. That is, each initial individual is generated by assigning a random number to each dimension. However, this random initialization overlooks the fact that the nodes in a community should have dense intra-connections and thus it is likely to generate communities with disordered structures. To cope with this problem, we propose a new random initialization method that is more suitable for the detection of local communities.

The proposed initialization method is based on the following assumption: nodes belonging to the local community are more likely to have short distances to the starting node while the other nodes are more likely to have long distances to the starting node. Hence, we introduce a parameter P_i which is a probability that decides if the i-th dimension of an initial individual should be assigned to 1. P_i is defined as follows:

$$P_i = P_{\max} - \frac{P_{\max} - P_{\min}}{d_{\max}} d_{ij}, \tag{14}$$

where $P_{\max}$ and $P_{\min}$ are the maximum and the minimum values of this probability, respectively, and $d_{\max}$ is the longest distance from all nodes in the search space to the starting node. Since the starting node j must be a member of its own local community, the $P_{\max}$ is set to be 1. It can be seen that P_i is a linear decreasing function of d_{ij}. After obtaining P_i, the corresponding initial value $x_i(0)$ is probabilistically assigned the value of 0 or 1 according to the following formula:

$$x_i(0) = \begin{cases} 1 & \text{if rand} < P_i \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

where *rand* is a randomly generated number.
Through this initialization scheme, nodes around the starting

node will have a larger possibility to join in the local community than those located far away and the quality of the initial population can be improved.

G. Local Search Operation

BPSO is an efficient optimization tool to find the optimal solutions in the search space and to generate local communities with a high quality. However, it still suffers from the common failing of EAs. That is, the population can easily get trapped in local optima and the optimization process will consequently be stopped. To improve the quality of the best individual generated by BPSO, we propose two local search schemes and incorporate them in ELCD. One is called the supervised moving scheme (SMS) and the other is called the unsupervised moving scheme (UMS), both of which are successively performed on the optimal solution output by BPSO.

The SMS is proposed by modifying the local moving (LM) scheme [35] to make it suitable for local community detection. It works as follows: For a candidate solution, we change the value of a decision variable and thus generate a new solution, then we calculate its fitness value and compare it with the old solution. If the new solution is not better than the old one, this dimension will be restored to its original value. The above process is repeatedly performed on each dimension of the solution in a randomized order, until its fitness can no longer be improved. In other words, the SMS operation tries to modify the detected local community through removing the redundant nodes and adding the missing nodes under the supervision of the fitness function.

The UMS is operated on the solution modified by SMS which is performed on each dimension of the solution in a randomized order: For the decision variable x_i, we firstly find all the neighbour nodes of its corresponding node i, then we count the number of the neighbour nodes whose values are the same with x_i, denoted as q_i. Subsequently, the parameter η_i for x_i can be calculated by the following formula:

$$\eta_i = \frac{q_i}{\text{degree}(i)}, \tag{16}$$

where degree(i) is the degree of node i. Then we change the value of x_i according to the following formula:

$$x_i = \begin{cases} \bar{x_i} & \text{if } \eta_i < \delta \\ x_i & \text{otherwise} \end{cases} \tag{17}$$

where $\bar{x_i}$ means changing the value of x_i, and δ is a predefined threshold. In fact, what the UMS operation does is to further improve the quality of the detected local community by finding the exceptional nodes and changing their memberships to the local community.

H. Recursive Partitioning Scheme

As mentioned before, the fitness function Q_l adopted by ELCD still suffers from the RL problem. As a result, it is difficult for ELCD to discover the ground-truth local communities hiding in bigger local community structures. Therefore, to address this problem and improve the resolution of our local community detection algorithm, we propose the Local Recursive Partition (LRP) scheme and incorporate it in ELCD.

The LRP scheme is proposed by modifying the recursive partition (RP) scheme [36]. LRP works as follows: When a local community is detected, ELCD is performed again on the network defined by the detected local community to discover smaller local communities. The above process will be repeatedly performed. That is to say, we try to solve the RL problem by recursively detecting the local communities on different scales of search spaces.

To determine when to terminate the LRP scheme, we propose a stop strategy which works as follows: When a local community is detected, we compare its fitness value Q_l with a threshold Q_{lmin}. Once $Q_l < Q_{lmin}$, which means the current local community has a little possibility to contain strong subcommunity structures, the LRP scheme will be terminated and the current local community will be output as the final result. Otherwise, the LRP scheme will be performed for the next iteration.

I. Complexity Analysis of ELCD

In this subsection, we discuss about the algorithm complexity of ELCD and give the relationship between the computing cost and the scale of networks. The computation in ELCD consists of two parts: the search in the decision space and the fitness evaluation. However, compared with the search part, the calculation of fitness function Q_l only needs to count e_l and d_l, whose computation complexity has no significant growth with the increase of the number of nodes in networks. Therefore, we only discuss the computation complexity in the search part. The search in ELCD has two phases: the evolutionary search performed by BPSO and the local search (i.e., SMS and UMS). Since the former phase consumes the vast majority of the computation time due to its iteration scheme, we only consider the complexity in the evolutionary search phase (i.e., BPSO) for the estimation of the computation complexity of ELCD.

Due to the LRP scheme, we first consider the complexity in one recursion. According to Algorithm 2, evolutionary operators should be performed on each decision variable in each individual for each generation. So the complexity for the k-th recursion $T_k(n)$ can be written as: $T_k(n) = r_k nml$, where n is the total number of nodes in the network, m is the population size, l is the maximum number of generations, $r_k \leq 1$ is the ratio of the number of nodes in the search space of the k-th recursion with n and we have $r_k < r_{k-1}$. Adding the complexity in each recursion together, we can estimate the computation complexity $T(n)$ for ELCD as follows:

$$T(n) = \sum_{k=1}^{p} r_k \cdot mln \tag{18}$$

where p is the total number of recursions performed by LRP. Due to the stopping criterion, p can not be very large (i.e., $p \ll n$). In fact, for large scale networks, there is no necessity to add nodes far away from the starting node into the search space, so $r_1 < 1$ and $T(n) < mln$ are satisfied in most cases.

Moreover, in next section, we will give the experimental results about the time cost of ELCD with increase of number of nodes in the search space.

IV. EXPERIMENTS

A. Experimental Settings

In this section, the performance of ELCD is evaluated on six synthetic LFR networks and five well-known real-world networks. The parameters in ECLD are set as follows: in the process of BPSO, the population size is 100, the maximum number of generations is 40, the maximum speed for a particle is 9, and the parameters ω, $c1$, and $c2$ are set to 0.729, 1.414, and 1.414, respectively. Furthermore, the parameter λ in the definition of search space is set to 1.0 and 0.6 for the local community detection in the small scale networks and the large scale networks, respectively, the p_{min} for the population initialization process is 0.1, the δ for the UMS operation is 0.8, and the Q_{lmin} in the LRP scheme is 0.3. All the experiments are performed on MATLAB R2016b.

B. Benchmark Datasets

The synthetic benchmark networks LFR are adopted by us to evaluate the performance of ELCD. LFR are introduced by Lancichinetti et al. [33] which have been widely used by researchers to test the performance of community detection algorithms. Generating networks by LFR needs 10 parameters, which are shown in Table IV.

In order to compare ELCD with the state-of-the-art DMF algorithms, in this paper, we adopt the same parameters as that in [24] for generating synthetic networks. To be specific, four small scale networks: LS1-LS4 and two large scale networks: LB1 and LB2 are generated, whose parameters are shown in Table V. LFR will generate benchmark networks with specific community structures according to the corresponding parameters.

What's more, to test the performance of ELCD on real-world networks, the following five real-world datasets, which have been widely adopted as benchmarks in complex network researches, are used in the experiments.

The Karate Club network [37] is proposed by Zachary, which is built by observing 34 members of a karate club. Each node in the Karate Club represents a member and each edge represents a close relationship between two of which.

The Dolphins network [38] is a network of dolphins which is proposed by studying 62 bottlenose dolphins inhabiting in New Zealand. An edge between two nodes means that these two dolphins interact frequently.

The US Political Books network [39] is a network of books about US politics and each edge between two books means they are frequently bought together by readers.

The Football network [2] is a well-known network representing the relationships of 115 football college teams in America. If two college teams play a football match, there is an edge between them.

The Amazon network [40] is a large-scale benchmark network collected from the Amazon website, each node in the network represents a commodity and each edge between two nodes means they are frequently bought together by consumers.

The basic information about these synthetic and real-world benchmark networks can be seen in Table VI.

C. Performance Metrics

In this paper, three widely used metrics: $recall$, $precision$ and $fscore$ are adopted to evaluate results output by local community detection algorithms, which are defined as follows [17]:

$$recall = \frac{|C_{Found} \cap C_{True}|}{|C_{True}|}, \tag{19}$$

$$precision = \frac{|C_{Found} \cap C_{True}|}{|C_{Found}|}, \tag{20}$$

$$fscore = \frac{2 * precision * recall}{precision + recall}, \tag{21}$$

where C_{Found} is the local community detected by algorithms and C_{True} is the ground-truth local community to which the given starting node belongs.

It can be seen from the above definitions that the metric $recall$ is the proportion of nodes which have been detected by algorithms in the real local community, which reflects the detection breadth of the algorithm. The metric $precision$ is the proportion of nodes that belong to the real local community in the detected local community, which reflects the detection accuracy of the algorithm. The $fscore$ is a combination of the above two metrics, which reflects the overall performance of the local community detection algorithm. The values of these three metrics are all in the range of $[0, 1]$, and the higher the values are, the better the algorithm is.

D. Results and Discussion

1) Results and comparisons on benchmark networks: Firstly, the proposed ELCD algorithm is tested on six synthetic networks generated by LFR, four of which: LS1, LS2, LS3, and LS4 are small scale networks with no more than 1000 nodes, and two of which: LB1 and LB2 are large scale networks with more than 10000 nodes. For each benchmark network, we successively select each node as the starting node and apply our ECLD to detect the local community to which it belongs, then the metrics $recall$, $precision$, and $fscore$ are calculated for this starting node. After testing all the nodes in a network, the average value and the standard deviation value of the above three metrics are calculated and shown in Table I. For comparison, we list the results of two classical algorithms: M and R and two state-of-the-art algorithms: DMF_M and DMF_R, which are implemented on the same benchmarks.

Then we do the same experiments on the five real-world networks and the results are shown in Table II. For comparison, in addition to the above four competitor algorithms, we list the results of three typical algorithms: LMD_R, LMD_M, and LMD_F, the local-based overlapping community detection algorithm F [18] and two recently proposed algorithms: VI and RTLCD in this table, which are all implemented on

TABLE I: Comparison of results of different algorithms on the synthetic networks.

Algorithms	Metrics	LS1	LS2	LS3	LS4	LB1	LB2
M	$recall$	0.7605±0.4064	0.2044±0.2278	0.4706±0.4719	0.1459±0.2750	0.5533±0.4456	0.7112±0.4395
	$precision$	0.7500±0.4017	0.8270±0.2312	0.5010±0.4254	0.7949±0.2548	0.8345±0.2879	0.7078±0.4331
	$fscore$	0.7549±0.4039	0.2894±0.2273	0.4722±0.4560	0.1859±0.2718	0.5880±0.4117	0.7073±0.4378
R	$recall$	0.7605±0.4064	0.1922±0.2076	0.4706±0.4719	0.1436±0.2711	0.5514±0.4451	0.7112±0.4395
	$precision$	0.7500±0.4017	0.8286±0.2330	0.5010±0.4254	0.7973±0.2547	0.8345±0.2878	0.7078±0.4331
	$fscore$	0.7549±0.4039	0.2187±0.2165	0.4722±0.4560	0.1841±0.2684	0.5866±0.4111	0.7073±0.4378
DMF_M	$recall$	0.9814±0.1307	0.8453±0.3228	0.9980±0.0438	0.6925±0.4311	0.9644±0.1731	0.9992±0.0254
	$precision$	0.9813±0.1316	0.9331±0.1892	0.9941±0.0464	0.8275±0.3082	0.9845±0.1091	1.0000±0.0000
	$fscore$	0.9813±0.1312	0.8595±0.2845	0.9960±0.0446	0.7089±0.4116	0.9673±0.1592	**0.9993±0.0211**
DMF_R	$recall$	0.9814±0.1307	0.8517±0.3197	0.9980±0.0438	0.7084±0.4260	0.9640±0.1733	0.9992±0.0254
	$precision$	0.9813±0.1316	0.9374±0.1811	0.9941±0.0464	0.8397±0.3045	0.9846±0.1090	1.0000±0.0000
	$fscore$	0.9813±0.1312	0.8651±0.2823	0.9960±0.0446	0.7240±0.4075	0.9671±0.1593	**0.9993±0.0211**
ELCD	$recall$	0.9927±0.0387	0.8802±0.1348	0.9953±0.0110	0.8956±0.1681	0.9830±0.0476	0.9895±0.0235
	$precision$	0.9876±0.0397	0.9658±0.0492	0.9995±0.0156	0.7948±0.2731	0.9875±0.0608	1.0000±0.0000
	$fscore$	**0.9892±0.0316**	**0.9146±0.0818**	**0.9973±0.0120**	**0.8209±0.2188**	**0.9836±0.0465**	0.9946±0.0121

TABLE II: Comparison of results of different algorithms on the real-world networks.

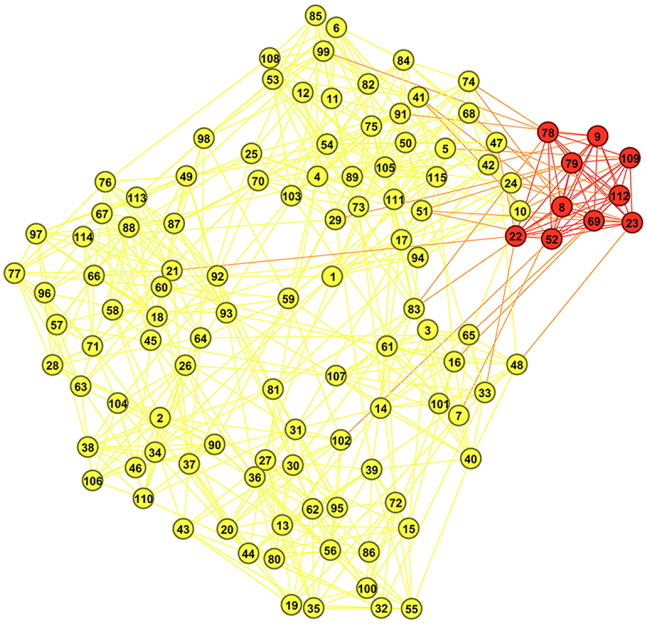
Algorithms	Metrics	Karate Club	Dolphins	Football	Political Books	Amazon
M	$recall$	0.6442±0.3039	0.3464±0.2322	0.7478±0.3704	0.4837±0.3782	0.6398±0.3763
	$precision$	0.8905±0.1447	0.9667±0.1118	0.6958±0.3658	0.7664±0.2604	0.9350±0.1569
	$fscore$	0.7007±0.2351	0.4685±0.2404	0.7146±0.3656	0.5257±0.3384	0.6881±0.3263
R	$recall$	0.5527±0.2521	0.3230±0.1999	0.7428±0.3724	0.4401±0.3459	0.6073±0.3697
	$precision$	0.9088±0.1406	0.9648±0.1125	0.6929±0.3717	0.7752±0.2657	0.9486±0.1490
	$fscore$	0.6474±0.2141	0.4505±0.2191	0.7101±0.3703	0.5030±0.3256	0.6703±0.3293
DMF_M	$recall$	0.9471±0.1736	0.6387±0.3089	0.8971±0.2180	0.7321±0.3149	0.7363±0.3310
	$precision$	0.7089±0.1772	0.9731±0.1212	0.8863±0.2456	0.7818±0.2485	0.9524±0.0979
	$fscore$	0.7794±0.1279	0.7255±0.2569	0.8881±0.2358	0.7295±0.3009	0.7744±0.2728
DMF_R	$recall$	0.9471±0.1736	0.6322±0.3149	0.8954±0.2227	0.7238±0.3105	0.7264±0.3324
	$precision$	0.7227±0.1811	0.9846±0.1210	0.8895±0.2475	0.7870±0.2508	0.9622±0.0887
	$fscore$	0.7884±0.1324	0.7242±0.2691	0.8893±0.2388	0.7286±0.3008	0.7721±0.2775
F	$recall$	0.5899	0.2476	0.6548	0.4352	NA
	$precision$	0.9045	0.8280	0.6562	0.7041	NA
	$fscore$	0.6771	0.3523	0.6543	0.4741	NA
LMD_M	$recall$	0.8542	0.2685	0.8025	0.4022	NA
	$precision$	1.0000	0.9581	0.8701	0.8155	NA
	$fscore$	0.9052	0.4011	0.8281	0.5133	NA
LMD_R	$recall$	0.6556	0.4373	0.9061	0.7059	NA
	$precision$	0.9227	0.9511	0.8093	0.7787	NA
	$fscore$	0.7368	0.5866	0.8489	0.7183	NA
LMD_F	$recall$	0.6148	0.4197	0.8986	0.7828	NA
	$precision$	0.9307	0.9538	0.8898	0.7710	NA
	$fscore$	0.7104	0.5610	0.8916	0.7571	NA
VI	$fscore$	0.7300	0.4900	0.8900	0.6300	0.7100
RTLCD	$fscore$	**1.0000**	0.7300	0.6600	0.7600	0.7400
ELCD	$recall$	0.9449±0.1412	0.8623±0.1410	0.9225±0.1941	0.8756±0.2034	0.7526±0.3108
	$precision$	0.9347±0.1423	0.9186±0.1345	0.8963±0.1933	0.7211±0.2603	0.9671±0.1288
	$fscore$	0.9392±0.1400	**0.8802±0.1084**	**0.9086±0.1946**	**0.7779±0.2548**	**0.7824±0.2041**

TABLE III: Comparison of results on the complete and incomplete real-world networks.

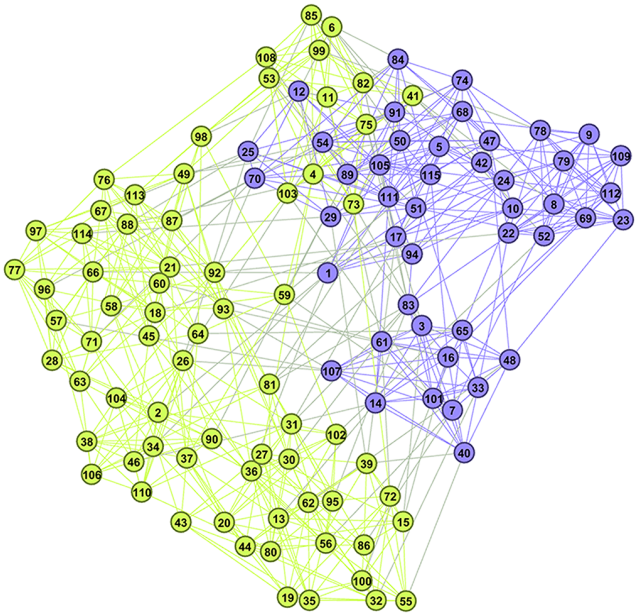
Algorithms	Metrics	Karate Club	Dolphins	Football	Political Books	Amazon
ELCD(incomplete)	$recall$	0.8989±0.1331	0.7448±0.1935	0.9352±0.1416	0.8119±0.1503	0.7683±0.2617
	$precision$	0.9699±0.0993	0.9370±0.0823	0.8929±0.1409	0.8134±0.2424	0.9452±0.0975
	$fscore$	0.9310±0.1120	0.8178±0.1322	0.9082±0.1360	**0.7964±0.2117**	**0.7861±0.1993**
ELCD(complete)	$recall$	0.9449±0.1412	0.8623±0.1410	0.9225±0.1941	0.8756±0.2034	0.7526±0.3108
	$precision$	0.9347±0.1423	0.9186±0.1345	0.8963±0.1933	0.7211±0.2603	0.9671±0.1288
	$fscore$	0.9392±0.1400	**0.8802±0.1084**	**0.9086±0.1946**	0.7779±0.2548	0.7824±0.2041
state-of-the-art	$fscore$	**1.0000** (RTLCD)	0.7300 (RTLCD)	0.8916 (LMD_F)	0.7600 (RTLCD)	0.7744±0.2728 (DMF_M)

the same benchmark networks. It should be explained that the standard deviations of some algorithms have not been published in related literatures and the *NA* in Table II means the corresponding tests have not been done by proposers.

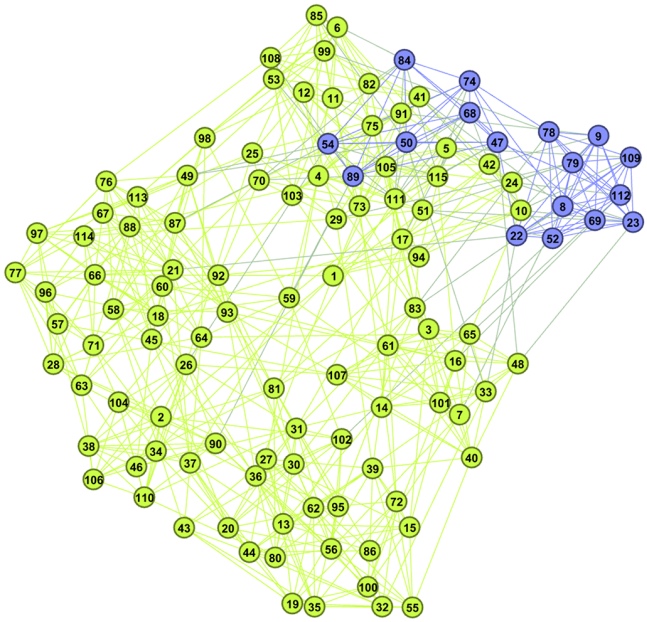
It can be seen from these two tables that the average values of $fscore$ of our ELCD algorithm are higher than the other methods on most benchmark networks even if the $recall$ and $precision$ are a little bit lower in some cases, which means that our method can effectively balance the detection breadth with the detection accuracy by utilizing more information of the networks and thus has superior overall performance than existing algorithms on the local community detection problem.



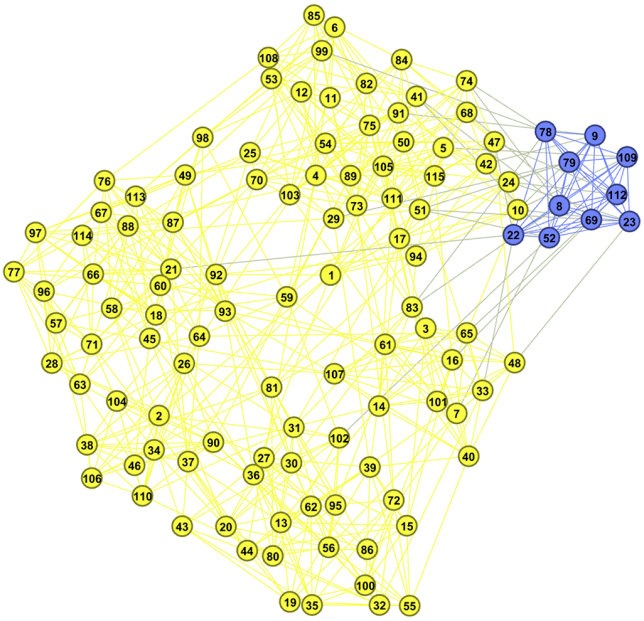
(a) The ground-truth local community



(b) The detected local community in the first stage



(c) The detected local community in the second stage



(d) The detected local community in the third stage

Fig. 2: The local community detection process for the 69th node of the Football network by ELCD. In this figure, the red nodes represent the ground-truth local community members, the blue nodes represent the local community members detected by ELCD, and the yellow nodes represent the other nodes in the network.

2) Results on a single node: To intuitively show the effectiveness of ELCD, we choose the Football network as an example and use ELCD to detect the local community to which the 69th node belongs. The local community detection process is shown in Fig. 2 and the optimization curves are shown in Fig. 3. From these two figures, it can be seen that under the control of the proposed LRP scheme, the ground-truth local community to which the 69th node belongs is recursively detected by ELCD through three stages, and the detected local community continuously shrinks during the entire process while the fitness value keeps increasing during each stage, which indicates that the RL problem caused by the fitness function Q_l is solved in ELCD.

Moreover, to show the robustness of ELCD to the position of the given starting node in its local community, in Table VII, we give the local community detection results by choosing each member in the local community showed in Fig. 2(a) as the starting node. It can be seen that the position of the

TABLE IV: Parameters in LFR.

Parameters	Explaination
N	number of nodes
k	average degree of nodes
$maxk$	maximum degree of nodes
μ	mixing parameter
$t1$	minus exponent for the degree sequence
$t2$	minus exponent for the community size distribution
$minc$	minimum of the community sizes
$maxc$	maximum of the community sizes
on	number of overlapping nodes
om	number of memberships of the overlapping nodes

TABLE V: Parameter settings in the synthetic datasets.

	S1	S2	S3	S4	B1	B2
N	100	100	1000	1000	10000	20000
k	10	5	15	7	7	15
$maxk$	20	15	50	25	20	50
μ	0.3	0.1	0.4	0.1	0.1	0.4
$t1$	2	2	2	2	2	2
$t2$	1	1	1	1	1	1
$minc$	10	10	10	10	10	10
$maxc$	20	50	50	300	100	50
on	0	0	0	0	0	0
om	0	0	0	0	0	0

TABLE VI: Basic information about benchmarks.

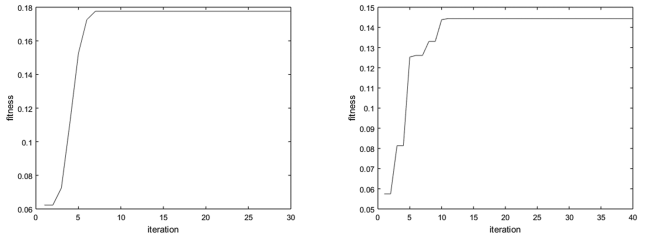
Network	♯Nodes	♯Edges	♯Communities	Type
LS1	100	518	6	synthetic
LS2	100	261	4	synthetic
LS3	1000	7677	36	synthetic
LS4	1000	3176	7	synthetic
LB1	10000	37526	269	synthetic
LB2	20000	153347	809	synthetic
Karate	34	78	2	social
Dolphins	62	159	2	social
Books	105	441	3	social
Football	115	615	12	social
Amazon	16716	97478	1163	social

given starting node has no significant influence on the local community detection results.

TABLE VII: Local community detection results of the Football network by choosing each node in a specific community as the starting node.

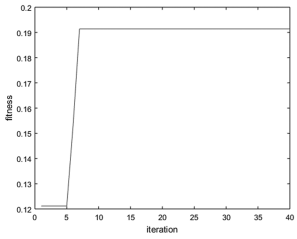
Node ID	$recall$	$precision$	$fscore$
78	1.0000	0.9091	0.9524
9	1.0000	0.9091	0.9524
109	1.0000	0.9091	0.9524
79	1.0000	1.0000	1.0000
112	1.0000	1.0000	1.0000
8	1.0000	0.9091	0.9524
22	1.0000	1.0000	1.0000
52	1.0000	0.8333	0.9091
69	1.0000	0.9091	0.9524
23	1.0000	1.0000	1.0000

3) Discussion about the parameter setting in ELCD: There are several parameters in our proposed algorithm. Among them, the parameter λ in the definition of search space is the most vital one, which determines the number of nodes considered in the local community detection process and the computing complexity of ELCD. To intuitively show the influence of parameter λ on the performance of our proposed



(a) The optimization curve during the first stage.

(b) The optimization curve during the second stage.



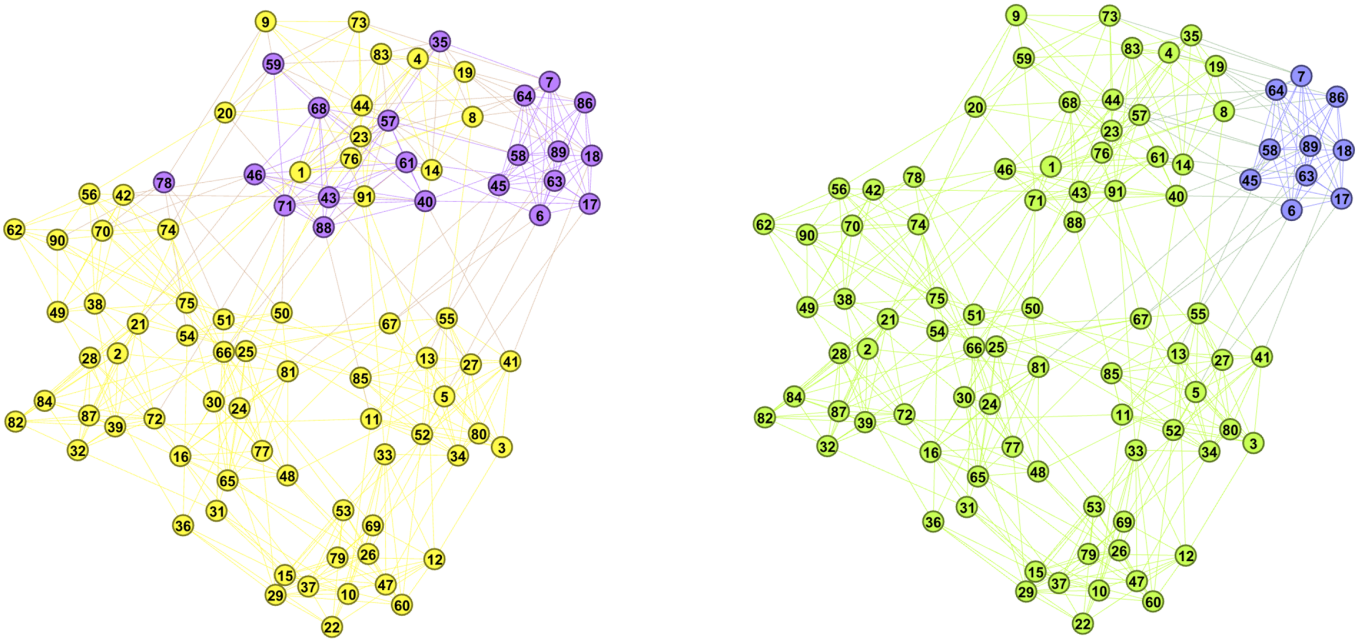
(c) The optimization curve during the third stage.

Fig. 3: The optimization curves of ELCD in different stages of the local community detection process for the 69th node of the Football network.

algorithm, we use LS4 as the test network, choose the 19th node as the starting node, and perform the local community detection experiment on it by setting λ as different values. For each value of λ, we perform 10 independent experiments, calculate the average values of the three performance metrics, and record the number of nodes in the search space and the average run time of ELCD. Experimental results are shown in Fig. 6.

It can be seen that the parameter λ has a great influence on both the performance and the computing complexity of ELCD. With increase of λ, more nodes in the network are considered by ELCD for finding the local community. As a result, the $precision$ metric shows a slight descending trend while the other two metrics show sharp increasing trends. On the other hand, the growth of the number of nodes in the search space also results in the increase of run time, however, the relationship between them is approximately linear, which accords with our complexity analysis about ELCD. Moreover, it should be noted that once λ beyond a certain value, the overall metric $fscore$ tends to be stable, but the time cost continues to increase, which means an appropriate value of λ should be carefully setted both to guarantee the algorithm's performance and to save computing time. In this paper, λ are setted by us for the benchmark networks through some trial and error experiments.

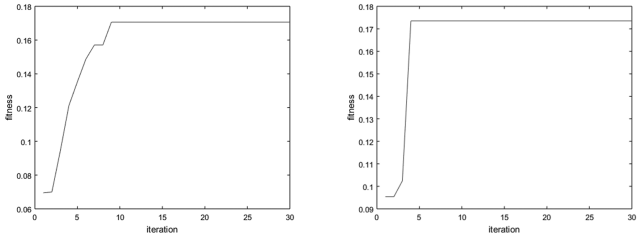
4) Results and comparisons on incomplete networks: In the above experiments, we use the complete networks as benchmarks to test the performance of ELCD. However, in some cases, it is expensive or impossible to obtain the global information of the real-world networks. To demonstrate the effectiveness of ELCD on incomplete networks, we cut the above five real-world benchmark networks into incomplete



(a) The detected local community in the first stage.

(b) The detected local community in the second stage.

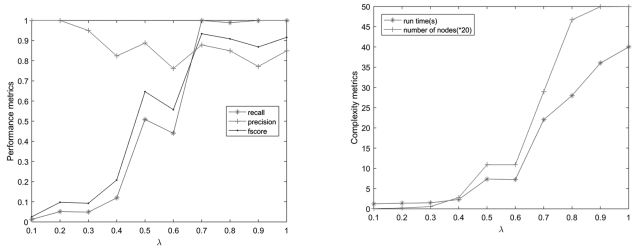
Fig. 4: The local community detection process for the 63th node of the incomplete Football network by ELCD. In this figure, the blue nodes represent the local community members detected by ELCD, and the yellow nodes represent the other nodes in the network.



(a) The optimization curve during the first stage.

(b) The optimization curve during the second stage.

Fig. 5: The optimization curves of ELCD in different stages of the local community detection process for the 63th node of the incomplete Football network.



(a) Average trends of performance metrics with different values of λ.

(b) Number of nodes in the search space and average run time with different values of λ.

Fig. 6: Experimental results of detecting the local community of the 19th node in LS4 network by setting different values of λ.

networks by removing some of their nodes and perform the same experiments on them.

To be specific, before detecting the local community to which a given starting node belongs, we randomly remove 30% nodes of the other community if the network has only two ground-truth communities and randomly remove all nodes of one of the other communities if the network has three ground-truth communities. For the Football network with many ground-truth communities, we randomly remove all nodes of two of the other communities. And for the large-scale Amazons network, we randomly remove all nodes of 30% of the other ground-truth communities in the network. The results of this experiment are shown in Table III. For comparison, we also list the results of ELCD performing on complete networks and the state-of-the-art algorithms.

It can be seen that for the Political Books and the Amazon networks, ELCD works better on their incomplete versions than on the complete versions. For the other three networks, even if removing some of their nodes influences the performance of ELCD to some degrees, ELCD still outperforms the state-of-the-art algorithms on most real-world benchmarks, which demonstrates that ELCD is effective regardless of the completeness of networks.

To intuitively show the effectiveness of ELCD on incomplete networks, we show the local community detection process for the 63th node (whose ID is 69 in the complete network) of the Football network in Fig. 4, and the optimization curves are shown in Fig. 5. It can be seen that after removing two communities from the network, the ground-truth local

community to which the node belongs can still be accurately discovered through two stages of recursive detection, which means that the network completeness has no influence on the local community detection by ELCD.

V. CONCLUSIONS

To summarize, this paper has proposed an EA-based method: ELCD to detect the local community to which a given starting node belongs by using the whole obtained information of the network. In detail, we used the BPSO as the basic optimization algorithm, combining with some proposed strategies used in the individual representation, fitness evaluation, and the local search operation, to effectively detect the local communities. Furthermore, the LRP scheme was also adopted to address the RL problem caused by the fitness function.

To show the effectiveness of our algorithm, we used both synthetic and real-world benchmark networks to test the performance of ELCD. The results showed that ELCD algorithm has superior overall detection performance than the classical algorithms and the state-of-the art algorithms do. Furthermore, we tested ELCD on incomplete real-world networks to demonstrate its effectiveness when the global information of an entire network can't be obtained, and the results showed that the completeness of networks has no significant influence on the performance of ELCD.

Finally, it is worth noting that even if the EA-based local community detection method has better performance, its efficiency may decrease a lot when coping with large scale networks due to its population iteration strategy and the LRP scheme. Consequently, in the future study, parallel computing can be incorporated in EAs to improve their optimization efficiency. Moreover, new fitness functions could be designed to solve the RL problem instead of employing the LRP scheme.

REFERENCES

[1] C. Pizzuti, "Evolutionary computation for community detection in networks: a review," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 464–483, 2018.

[2] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the national academy of sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[3] M. Gong, Q. Cai, X. Chen, and L. Ma, "Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 1, pp. 82–97, 2014.

[4] A. Bailey, M. Ventresca, and B. Ombuki-Berman, "Genetic programming for the automatic inference of graph models for complex networks," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 405–419, 2014.

[5] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Physical review E*, vol. 70, no. 6, p. 066111, 2004.

[6] H. Shen and X. Cheng, "Spectral methods for the detection of network community structure: a comparative analysis," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2010, no. 10, p. P10020, 2010.

[7] W. Lin, X. Kong, P. S. Yu, Q. Wu, Y. Jia, and C. Li, "Community detection in incomplete information networks," in *Proceedings of the 21st international conference on World Wide Web*. ACM, 2012, pp. 341–350.

[8] M. Tasgin, A. Herdagdelen, and H. Bingol, "Community detection in complex networks using genetic algorithms," *arXiv preprint arXiv:0711.0491*, 2007.

[9] J. H. Holland *et al.*, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[10] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.

[11] Y. Shi *et al.*, "Particle swarm optimization: developments, applications and resources," in *evolutionary computation, 2001. Proceedings of the 2001 Congress on*, vol. 1. IEEE, 2001, pp. 81–86.

[12] M. Dorigo and G. Di Caro, "Ant colony optimization: a new meta-heuristic," in *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*, vol. 2. IEEE, 1999, pp. 1470–1477.

[13] X. Yang, "Firefly algorithms for multimodal optimization," in *International symposium on stochastic algorithms*. Springer, 2009, pp. 169–178.

[14] ——, "A new metaheuristic bat-inspired algorithm," in *Nature inspired cooperative strategies for optimization (NICSO 2010)*. Springer, 2010, pp. 65–74.

[15] Q. Chen and T. Wu, "A method for local community detection by finding maximal-degree nodes," in *Machine Learning and Cybernetics (ICMLC), 2010 International Conference on*, vol. 1. IEEE, 2010, pp. 8–13.

[16] S. Papadopoulos, A. Skusa, A. Vakali, Y. Kompatsiaris, and N. Wagner, "Bridge bounding: A local approach for efficient community discovery in complex networks," *arXiv preprint arXiv:0902.0871*, 2009.

[17] Q. Chen, T. Wu, and M. Fang, "Detecting local community structures in complex networks based on local degree central nodes," *Physica A: Statistical Mechanics and its Applications*, vol. 392, no. 3, pp. 529–537, 2013.

[18] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New Journal of Physics*, vol. 11, no. 3, p. 033015, 2009.

[19] J. P. Bagrow, "Evaluating local community methods in networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 05, p. P05001, 2008.

[20] F. Luo, J. Z. Wang, and E. Promislow, "Exploring local community structures in large networks," *Web Intelligence and Agent Systems: An International Journal*, vol. 6, no. 4, pp. 387–400, 2008.

[21] X. Zhang, L. Wang, Y. Li, and W. Liang, "Extracting local community structure from local cores," in *International Conference on Database Systems for Advanced Applications*. Springer, 2011, pp. 287–298.

[22] A. Clauset, "Finding local community structure in networks," *Physical review E*, vol. 72, no. 2, p. 026132, 2005.

[23] Y. Wu, H. Huang, Z. Hao, and F. Chen, "Local community detection using link similarity," *Journal of computer science and technology*, vol. 27, no. 6, pp. 1261–1268, 2012.

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

[25] J. Chen, O. Zaïane, and R. Goebel, "Local community identification in social networks," in *Social Network Analysis and Mining, 2009. ASONAM'09. International Conference on Advances in*. IEEE, 2009, pp. 237–242.

[26] J. P. Bagrow and E. M. Bollt, "Local method for detecting communities," *Physical Review E*, vol. 72, no. 4, p. 046108, 2005.

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

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

[29] Y. Yao, W. Wu, M. Lei, and X. Zhang, "Community detection based on variable vertex influence," in *2016 IEEE First International Conference on Data Science in Cyberspace (DSC)*. IEEE, 2016, pp. 418–423.

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

[31] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*. IEEE, 1995, pp. 39–43.

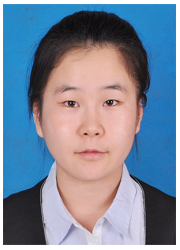
[32] J. Liu, R. Yang, and S. Sun, "The analysis of binary particle swarm optimization," *Journal of Nanjing University (Natural Sciences)*, vol. 47, no. 5, pp. 504–514, 2011.

[33] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical review E*, vol. 78, no. 4, p. 046110, 2008.

[34] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," *Proceedings of the National Academy of Sciences*, vol. 104, no. 1, pp. 36–41, 2007.
[35] R. Rotta and A. Noack, "Multilevel local search algorithms for modularity clustering," *Journal of Experimental Algorithmics (JEA)*, vol. 16, pp. 2–3, 2011.
[36] J. Ruan and W. Zhang, "Identifying network communities with a high resolution," *Physical Review E*, vol. 77, no. 1, p. 016104, 2008.
[37] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of anthropological research*, vol. 33, no. 4, pp. 452–473, 1977.
[38] D. Lusseau, "The emergent properties of a dolphin social network," *Proceedings of the Royal Society of London B: Biological Sciences*, vol. 270, no. Suppl 2, pp. S186–S188, 2003.
[39] V. Krebs, "Political books network." [Online]. Available: http://www.orgnet.com
[40] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowledge and Information Systems*, vol. 42, no. 1, pp. 181–213, Jan 2015. [Online]. Available: https://doi.org/10.1007/s10115-013-0693-z

Chao Lyu received the B.S. degree from China University of Mining and Technology, Xuzhou, Jiangsu, China, in 2014 and the M.Eng. degree from Lanzhou University, Lanzhou, Gansu, China, in 2017. He is currently pursuing the Ph.D. degree at Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, Guangdong, China. His research interests include complex network and evolutionary computation.

Yuhui Shi received the Ph.D. degree in electronic engineering from Southeast University, Nanjing, China, in 1992. He is a Chair Professor in the Department of Computer Science and Technology at the Southern University of Science and Technology (SUSTech), Shenzhen, Guangdong, China since September 2016. From 1998 to 2007, he was with Electronic Data Systems Corporation, Indianapolis, IN, USA as an applied specialist. From 2008 to August 2016, he was a Professor with the Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou, Jiangsu, China. He is an IEEE fellow. He coauthored a book on swarm intelligence together with Dr. James Kennedy and Prof. Russell Eberhart, and another book entitled Computational Intelligence: Concept to Implementation together with Prof. Russell Eberhart. He has extensive knowledge on innovation and creative problem-solving skills. His main research interests include computational intelligence techniques (including swarm intelligence) and their applications.



Lijun Sun received the B.S. and M.Eng. degrees from Lanzhou University, Lanzhou, Gansu, China, in 2013 and 2016, respectively. She is currently a research assistant at the Southern University of Science and Technology since April 2017. Her research interests include swarm intelligence and machine learning.