IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received April 1, 2022, accepted April 13, 2022, date of publication April 18, 2022, date of current version April 28, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3168714

Research on Dynamic Community Detection Method Based on an Improved Pity Beetle Algorithm

YAN-JIAO WANG[1], **JIA-XU SONG**[1], **AND PENG SUN**[1,2]
[1]School of Electrical Engineering, Northeast Electric Power University, Jilin 132012, China
[2]Jilin Branch, China United Telecommunications Company Ltd., Changchun 130000, China

Corresponding author: Jia-Xu Song (songjiaxu1996@163.com)

This work was supported by the Project of Scientific and Technological Innovation Development of Jilin in China under Grant 20210103090.

ABSTRACT In the last decade, community detection in dynamic networks has received increasing attention, because it can not only uncover the community structure of the network at any time but also reveal the regularity of dynamic networks evolution. Although methods based on the framework of evolutionary clustering are promising for dynamic community detection, there is still room for further improvement in the snapshot quality and the temporal cost. In this study, a dynamic community detection algorithm based on optional pathway guide pity beetle algorithm (DYN-OPGPBA), which is a novel dynamic community detection method based on the framework of evolutionary clustering, is proposed. We propose an improved PBA for community detection of the network at the first time step, including a discrete search strategy based on adjacent nodes, a closeness-based community modification strategy and a crowded community split strategy. Compared with many representative static community detection methods, the proposed method has some superior detection accuracy. A neighbour vector competition-based individual update strategy and an external population size restriction mechanism are also proposed for community detection at subsequent time steps. Results show that DYN-OPGPBA has a better balance between snapshot quality and temporal cost than two representative dynamic community detection methods.

INDEX TERMS Community detection, dynamic network, evolutionary clustering, Pity Beetle algorithm.

I. INTRODUCTION

Complex networks exist widely in many fields, including biological science, computer science, social science, and computer science, etc [1]. Community structure is one of the most important characteristics of complex networks, which can be intuitively regarded as a group of nodes with dense intraconnections and sparse interconnections. Many networks derived from the real world are dynamic in nature. For instance, some members might join or leave a team, or a team could even dissolve. To further utilize, transform and predict networks, the dynamics of the community structure in dynamic networks, called dynamic community detection, need to be traced. In recent decades, we have experienced increasing demand for dynamic community detection from a wide range of real-world applications, such as analyzing and observing changes in news focus from news headlines, analyzing and discovering stock trends from the stock market, and analyzing and discovering community development trends from the blog space.

In general, dynamic networks can be represented as a sequence of snapshots of static networks at each time step. Dynamic community detection aims to accurately uncover the community structure of the snapshot at each time step and reveal the regularity of dynamic networks' evolution. Dynamic community detection algorithms can be roughly divided into two categories according to the methods used.

The first category is improved traditional community detection algorithms in static networks directly employed to dynamic community detection. For instance, Bilal S et al proposed a community detection algorithm based on evolutionary algorithm and modularity [2]. In this approach they use an evolutionary algorithm to find the first community structure that maximizes the modularity. Finally, find the

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

community structure with the highest value of modularity through merging communities. H.Papadakis et al proposed a standing for synthetic coordinate community detection (named SCCD) [3], which finds the entire community structure of a network on the basis of local interactions between neighboring nodes and an unsupervised distributed hierarchical clustering algorithm. Zhang et al proposed seed expansion with generative adversarial learning (SEAL) [4], a framework for learning heuristics for community detection. Marya et al proposed a community detection method for temporal multilayer networks [5]. Mahsa S et al introduced a new Louvain-based dynamic community detection algorithm that relied on the derived knowledge of the previous steps of network evolution [6]. Kamal B et al proposed a local community detection algorithm based on the detection and expansion of core nodes [7], text-associated DeepWalk-spectral clustering (TADW-SC) [8] and attributed spectral clustering (ASC) [9].

The second category is the algorithms based on evolutionary algorithms. It starts from the concept of temporal smoothness of network evolution proposed by Chakrabarti *et al.*, which further revealed the characteristic regularity of network changes [10]. On the basis of the temporal smoothness, Pizzuti *et al.* proposed a dynamic community detection method based on evolutionary clustering, which not only improves the detection efficiency but is also more in line with actual network changes [11]. This method mainly contains two steps. Firstly, the snapshot quality of the static network is optimized at the first time step. Secondly, the snapshot quality and temporal cost are optimized simultaneously at the subsequent time steps. Mathematically, the above two steps can be essentially viewed as a single-objective optimization and multi-objective optimization problems, respectively, which are rated at NP-hard. In the past few decades, many experiments confirmed that swarm intelligence evolutionary algorithms are the most competitive and effective methods for solving optimization problems. Therefore, since then, a relevant number of dynamic community detection approaches based on the framework of evolutionary clustering have been proposed, which adopt EAs as optimization strategies, such as genetic algorithm (GA), particle swarm algorithm (PSO), and differential evolutionary algorithm (DE). For instant, a consensus community-based particle swarm optimization for dynamic community detection (CCPSO) proposed by Zeng et al [12], FaceNet [13], DYN-MOGA [14], DYN-DMLS [15], MOEA-SA [16], DYN-MODPSO [17], L-DMGAPSO [18] and ECD [19]. Up to now, a series of dynamic community detection approaches based on the frame of evolutionary clustering has become the most competitive and widely utilized approach of uncovering the community structure in dynamic networks.

However, dynamic community detection approaches based on the framework of evolutionary clustering have two disadvantages. First, the detection accuracy of the snapshot at the first time step has not received enough attention. If the community division of the snapshot at the first time step is not accurate, then the community structure division at the subsequent time steps will become increasingly inaccurate, that is, the problem of error accumulation occurs. Second, the modularity and normalized mutual information (NMI) obtained at each time step can be still improved further because EAs as optimization strategies of the dynamic community detection approaches based on the framework of evolutionary clustering are usually traditional, such as GA, PSO and DE. These EAs generally have many drawbacks, such as slow convergence speed and easily falling into the local optimum.

To address these issues, this article proposes an dynamic community detection method based on the framework of evolutionary clustering, which introduces a novel bio-inspired meta-heuristic algorithm called the pity beetle algorithm (PBA) [20] as evolutionary strategy. Because many experimental results on various complex continuous numerical optimization problems show that PBA outperforms many widely used meta-heuristics methods (such as GA, DE, PSO, and ABC, etc) in terms of accuracy, stability and speed. The proposed algorithm is hereafter called DYN-OPGPBA. The major innovations and contributions of this article can be summarized as follows:

1) To improve the community detection accuracy of the snapshot at the first time step and avoid the error accumulation problem, a static community detection algorithm based on the optional pathway guide pity beetle algorithm is proposed, hereafter called CD-OPGPBA. The main idea of CD-OPGPBA is to optimize the modularity function based on an improved PBA with excellent optimization performance. In CD-OPGPBA, a discrete search strategy uses adjacent nodes and network topological information as the crucial factors to enhance its global optimization performance. To guarantee that the number of divided communities is equal to that of real communities, a closeness-based community modification strategy and a crowded community split strategy are designed.

2) To improve modularity and NMI at each subsequent time step, a dynamic community detection algorithm based on the multi-objective optional pathway guide pity beetle algorithm is proposed. The initial community structure is obtained by choosing the optimal community division at the previous time step on the basis of our proposed novel identification method based on the module density, and an improved multi-objective pity beetle algorithm based on decomposition (MOEAD/PBA) with a neighbour vector competition-based individual updating strategy. An external population size restriction mechanism is introduced to optimize modularity and NMI simultaneously.

In experiments, two main scenarios were considered. 1) In the first scenario, the effectiveness of CD-OPGPBA in community quality is verified because the detection quality of the network at the first time step seriously affects the community detection at the subsequent time steps. Tested on static networks, including LFR and real-world networks, the superior community quality of CD-OPGPBA is

TABLE 1. Formal notations used in this study.

Symbol	Description
$G^t(V,E)$	a graph G^t with the node set V^t and the edge set E^t at t-th time step
V^t	a node set at t-th time step ($V^t=\{v_1^t,v_2^t,K,v_n^t\}$)
E^t	a edge set at t-th time step ($E^t=\{(v_i^t,v_j^t)\mid v_i^t\in V^t,v_j^t\in V^t, and\ i\neq j\}$)
n^t, m^t	the number of nodes in V^t and the number of edges in E^t
A^t	the adjacency matrix of G^t at t-th time step, where $A_{ij}{}^t\in\{0,1\}^{n^t\times n^t}$
d^t	the degree of the i-th node of G^t at t-th time step, where $d_i{}^t\in\sum_{i=1}^{n^t}A_{ij}{}^t$
C^t	a community set of G^t at t-th time step, where $C^t=\{C_1{}^t,C_2{}^t,\ldots,C_k{}^t\},C_1{}^t\cup\ldots\cup C_k{}^t=V^t$
k^t	the number of communities in C^t
U^t	Membership matrix
N_{pop}	Population size
$A=\{a_1,a_2,\ldots,a_k,\ldots,a_D\}$	the top bounds of the variable X
$B=\{b_1,b_2,\ldots,b_k,\ldots,b_D\}$	the bottom bounds of the variable X

further validated through comparison with other well-known community detection methods. 2) In the second scenario, DYN-OPGPBA is compare with two well-known algorithms over Infectious Social Patterns dynamic networks and dynamic LFR networks. At most time steps of the datasets, a high level of modularity and NMI was achieved by DYN-OPGPBA. Moreover, both CD-OPGPBA and DYN-OPGPBA work without prior knowledge of the total number of communities and require only a few specific optimization parameters.

The rest of this article is organized as follows. Section II introduces definitions and concepts about the dynamic community detection problem and related work. Section III describes the details of the proposed DYN-OPGPBA. Section IV shows the empirical results to illustrate the effectiveness of the proposed method. Finally, Section V concludes this article.

II. CONCEPTS AND RELATED WORK

A. SYMBOL DECLARATION

Table 1 gives the main formal notations involved in this article, including graphs, membership matrix, and PBA.

B. DESCRIPTION OF DYNAMIC NETWORK

In dynamic networks, the nodes and the edges generally change with time. For instance, some nodes and edges might be added, and others could be deleted. Fig. 1 gives an example of a dynamic network which changes from step t = 1 to t = 4. Compared with the network at t = 1, node 9 is added edges (9, 5) and (9, 6) are newly added at t = 2. At t = 3, edge (9, 5) is removed, and edge (9, 3) is newly added. At t = 4, node 7 is removed, node 10 is newly added, edge (3, 5) is removed, and edges (10, 3) and (10, 4) are newly added.

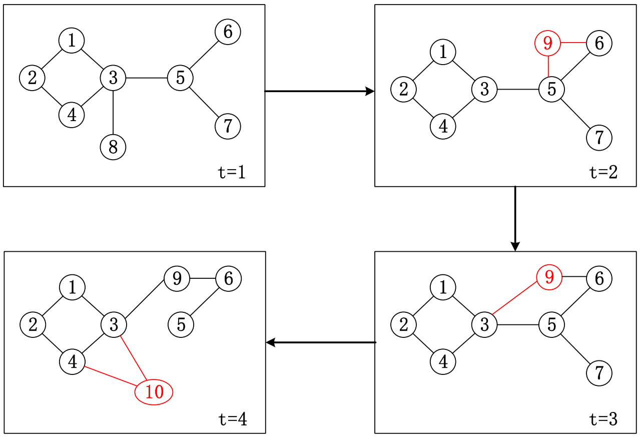


FIGURE 1. Example of dynamic network.

Generally, a dynamic network can be described as a sequence $G = \left\{G^1, G^2, \cdots, G^t, \cdots, G^T\right\}$, where $t \in \{1, 2, \cdots, T\}$ represents the t-th time step, G^1 represents the original network, and G^t is a snapshot of nodes and connections among these nodes at time step t, denoted as $G^t = \left\{V^t, E^t\right\}$, where $V^t = \left\{v_1^t, v_1^t, \cdots, v_n^t\right\}$ and $E^t = \left\{(v_i^t, v_j^t)|v_i^t \in V^t, v_j^t \in V^t, and\ i \neq j\right\}$ are the nodes set and the edges set of G^t, respectively. Generally, the topology information of G^t can also be represented by the adjacency matrix $A^t = \left(A_{ij}\right)_{n^t\times n^t}^t$, where A_{ij}^t represents the node type of G^t and whether there is an edge between nodes i and j at t-th time step. In this paper, we merely consider unweighted and undirected networks, i.e., $A_{ij}^t = A_{ji}^t$ and $A_{ij}^t \in \{0, 1\}$.

C. DESCRIPTION OF DYNAMIC COMMUNITY DETECTION

Dynamic community detection often involves two objectives: to optimize snapshot quality at each time step, thus enabling us to gain insights into network topological features as much as possible; and to guarantee slow changes between the network structures at consecutive time steps. After dynamic community detection, we finally obtain a sequence of community structures, denoted as $C = \{C^1, C^2, \ldots, C^t, \ldots C^T\}$, where C^t is the community division in G^t at the t-th time step, and $C^t = \{C_1^t, C_2^t, \ldots, C_{k^t}^t\}$ has k^{t} divided subgraphs. This article focuses only on nonoverlapping community structures, i.e. C^t needs to satisfy the following conditions:

$$\begin{cases} C^t = \bigcup_{i=1}^{k^t} C_i^t & where\ C_i^t \neq \phi \\ C_i^t \cap C_j^t = \phi & \forall i \neq j\ and\ i, j \in \left\{1, 2, \ldots, k^t\right\} \end{cases} \tag{1}$$

Briefly, dynamic community detection needs to optimize the snapshot quality and the temporal cost at each time step simultaneously. Generally, we adopt the normalized mutual information (NMI) [19] to estimate the temporal cost, while the snapshot quality can be measured by the following fitness functions: community score (CS) [22], modularity density (D) [23] and modularity (Q) [24]:

(1) Normalized Mutual Information (NMI)

$$NMI\left(C^t, C^{t-1}\right) = \frac{-2\sum_{i=1}^{m^t}\sum_{j=1}^{m^{t-1}} L_{ij}\log\left(L_{ij}N/L_{i.}L_{j.}\right)}{\sum_{i=1}^{m^t} L_{i.}\log\left(L_{i.}/Num^t\right)+\sum_{j=1}^{m^t} L_{.j}\log\left(L_{.j}/Num^{t-1}\right)} \quad (2)$$

where $C^t = \left\{c_1^t, c_2^t, \ldots, c_{m^t}^t\right\}(C^{t-1} = \left\{c_1^{t-1}, c_2^{t-1}, \ldots, c_{m^{t-1}}^{t-1}\right\})$ represents the community structure of the snapshot at the t-th time step (the previous time step), m^t (m^{t-1}) represents the number of the divided subgraphs in G^t (G^{t-1}), $Num^t(Num^{t-1})$ is the number of nodes in G^t (G^{t-1}), L is the confusion matrix whose element L_{ij} represents the number of nodes both in the community $C_i^t \in C^t$ and the community $C_j^{t-1} \in C^{t-1}$, and $L_{i.}$ and $L_{.j}$ represent the sum of the elements of L in row i and column j, respectively. $NMI\left(C^t, C^{t-1}\right) \in [0, 1]$, and a high value of $NMI\left(C^t, C^{t-1}\right)$ indicates greater similarity between C^t and C^{t-1}, i.e. $NMI\left(C^t, C^{t-1}\right) = 0$ means C^t and C^{t-1} are different.

(2) Modularity (Q)

$$Q = \frac{1}{2M}\sum_{i,j}\left(A_{ij} - \frac{k_i k_j}{2M}\right)\delta(i,j) \quad (3)$$

where M is the number of edges in the network, and k_i and k_j are the degrees of nodes i and j, respectively. $\delta(i, j)$ represents the community relationship between nodes i and j, i.e. if nodes i and j belong to the same community, $\delta(i, j) = 1$; otherwise, $\delta(i, j) = 0$. The higher the value of Q, the better the quality of community divisions.

(3) Modularity density(D)

$$D = \sum_{i=1}^{m} d(G_i) = \sum_{i=1}^{m}\frac{L(V_i, V_i) - L\left(V_i, \overline{V_i}\right)}{|V_i|} \quad (4)$$

where m represents the number of communities in the network, $|V_i|$ represents the number of nodes in the i-th community, and $L(V_i, V_i)$ and $L\left(V_i, \overline{V_i}\right)$ represent the number of internal edges of the i-th community and the number of edges with other communities, respectively.

(4) Community score(CS)

$$CS = \sum_{k=1}^{m}\frac{\sum_{i\in C_k}(\mu_i)^r}{|C_k|} \times \sum\nolimits_{i,j\in C_k} A_{ij} \quad (5)$$

where r is the adjustment parameter (usually, $r = 2$), $\sum_{i,j\in C_k} A_{ij}$ represents the number of edges in k-th community C_k and μ_i is conducted as (6).

$$\mu_i = \frac{1}{|C_k|}\sum\nolimits_{j\in C_k} A_{ij} \quad (6)$$

D. EVOLUTIONARY CLUSTERING-BASED METHODS FOR DYNAMIC COMMUNITY DETECTION

As shown in Fig.2, evolutionary clustering-based methods for the dynamic community detection contains two steps. At the first time step $t = 1$, to obtain excellent snapshot quality, the criterion function (i.e. Q, D, CS, etc.) is optimized by using a single-objective evolutionary algorithm. Afterwards, optimize the snapshot quality and the temporal cost at subsequent time steps simultaneously by a multi-objective evolutionary algorithm. $C_{set}^{t-1} = \left\{C_1^{t-1}, C_2^{t-1}, \ldots, C_k^{t-1}\right\}$ is assumed as the selected best community division of the snapshot at the previous time step. The snapshot quality at the current time step can be optimized by maximizing Q, D, CS, etc., and the temporal cost can be optimized by minimizing $NMI^t\left(C^t, C_{set}^{t-1}\right)$. Finally, a sequence of community divisions $C = \{C^1, C^2, \ldots, C^t, \ldots C^T\}$ will be obtained.

E. PITY BEETLE ALGORITHM

The pseudo-code of the PBA is shown in Algorithm 1, where $RST(\bullet)$ represents a random sampling technique. Here $X^0 = RST\left(B, A, D, N_{pop}\right)$ is generated as follows. First, each dimension space in the D-dimensional search space is evenly divided into N_{pop} segments, and a value is randomly generated according to Eq. (7) in each segment. Then randomly pair the N_{pop} samples obtained in the k-th search space with the k-th dimension of the N_{pop} individuals in the population.

$$\Delta_{k,i} = a_k + \frac{b_k - a_k}{N_{pop}}(i - 1 + rand) \quad (7)$$

where $\Delta_{k,i}$ represents the sample value of the i-th segment in the k-th dimensional space, and rand represents a random number in the range of [0, 1].

III. PROPOSED DYNAMIC COMMUNITY DETECTION METHOD

A. PROCEDURE OF THE PROPOSED ALGORITHM

In this article, we contribute to the framework of evolutionary clustering for dynamic networks, and propose a dynamic community detection method based on the optional pathway guide pity beetle algorithm (DYN-OPGPBA). As shown in Fig. 3, the proposed DYN-OPGPBA contains two stages. The first stage aims to obtain the optimal snapshot quality at the first time step by using an improved single-objective PBA (described in Section III.B), while the second stage uncovers communities by optimizing the snapshot quality and the temporal cost at subsequent time steps simultaneously by a multi-objective PBA (described in Section III.C). Obviously, the objectives to be optimized and the adopted optimization algorithms in the two above stages are different. In the first stage, we optimize the modularity as Eq. (3) by using an improved PBA with a discrete search strategy based on adjacent nodes, a closeness-based community modification strategy and a crowded

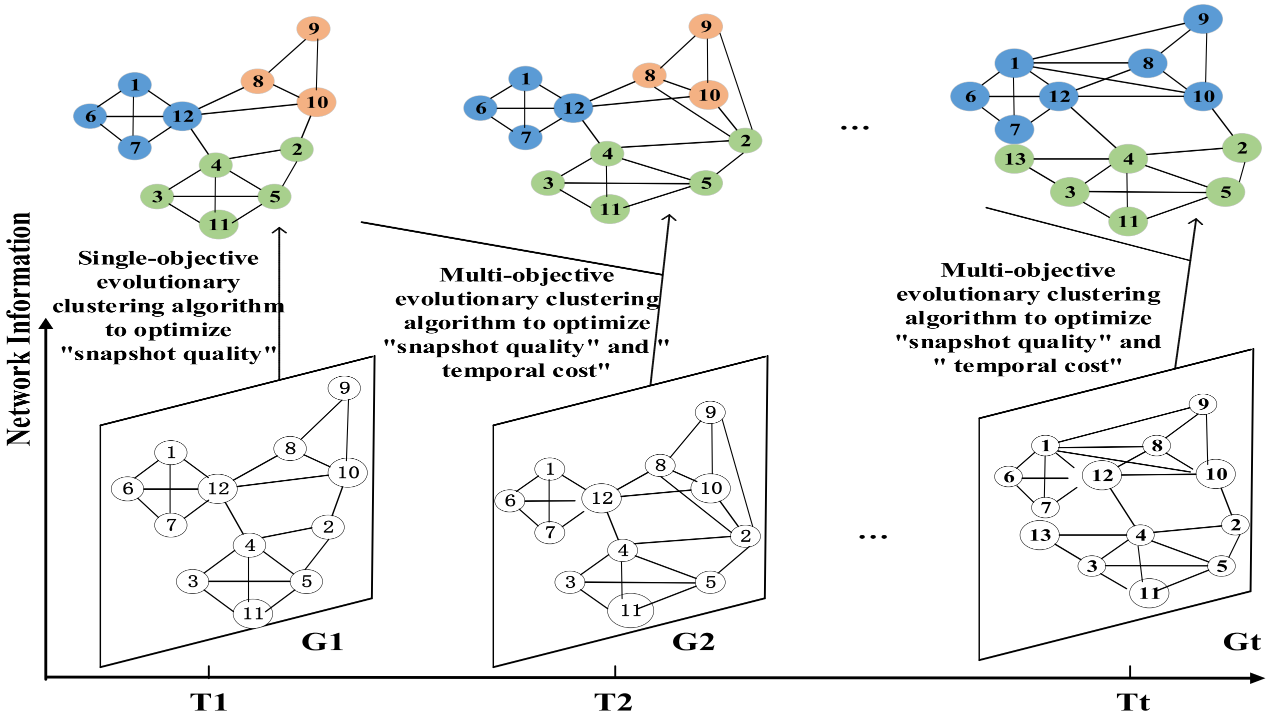

Network Information
Single-objective evolutionary clustering algorithm to optimize "snapshot quality"
Multi-objective evolutionary clustering algorithm to optimize "snapshot quality" and " temporal cost"
Multi-objective evolutionary clustering algorithm to optimize "snapshot quality" and " temporal cost"
G1
G2
Gt
T1
T2
Tt
...
...

FIGURE 2. The framework of evolutionary clustering-based community detection methods for the dynamic networks.

community split strategy. In the second stage, we optimize the two following objectives simultaneously: the modularity as Eq. (3) which is used to evaluate the snapshot quality and NMI as Eq. (2) which is used to evaluate the temporal cost, by using MOEAD/PBA (Algorithm 3), which employs the framework of decomposition-based approaches, a neighbour vector competition-based individual update strategy and an external population size restriction mechanism. To identify the optimal community division from a set of community divisions obtained by MOEAD/PBA at each subsequent time step, we proposed a novel identification method based on the module density, as mentioned in Section III.C.(2).

B. COMMUNITY DETECTION FOR THE FIRST TIME STEP

In this section, the proposed CD-OPGPBA for community detection at the first time step is described. As shown in Fig. 3, the CD-OPGPBA mainly contains four parts: population initialization, the optimization process, the closeness-based community modification strategy and the crowded community split strategy.

1) POPULATION INITIALIZATION

Our proposed method uses a popular and straightforward adjacent node-based encoding method [21] to generate a set of initial individuals, where the process of generating an individual according to the adjacent node-based encoding method is as follows: Assume that a network contains n nodes, the size of an individual X is equal to n, and each dimension X_j is a random integer between 1 and n, which represents the node the j-th node is connected to.

Fig. 4 illustrates of how an individual is encoded and decoded. The network G contains 7 nodes. Assume that $X = \{2, 3, 2, 5, 6, 5, 4\}$ is generated according to the adjacent node-based encoding method. It indicates that each node from nodes 1 to 7 is connected to nodes 2, 3, 2, 5, 6, 5 and 4, respectively. Generally, we call two connected nodes as a two-node set with adjacency relation. Therefore, two-node sets $\{\{1, 2\}, \{2, 3\}, \{3, 2\}, \{4, 5\}, \{5, 6\}, \{6, 5\}, \{7, 4\}\}$ are present in $X = \{2, 3, 2, 5, 6, 5, 4\}$. When we decode the individual X, according to the connection relation between nodes, we can find that $\{\{1, 2\}, \{2, 3\}, \{3, 2\}, \{4, 5\}, \{5, 6\}, \{6, 5\}, \{7, 4\}\}$ contain the connected vectors $\{1, 2, 3\}$ and $\{4, 5, 6, 7\}$, which indicates that nodes 1, 2 and 3 belong to the same community, and the remaining nodes are in the other community, as shown in network Gd. Therefore, $X = \{2, 3, 2, 5, 6, 5, 4\}$can be decoded as $\{1, 1, 1, 2, 2, 2, 2\}$ or $\{2, 2, 2, 1, 1, 1, 1\}$.

2) ADJACENT NODE-BASED DISCRETE SEARCH STRATEGY

In this article, a discrete search strategy for updating individuals is designed based on adjacent nodes, because original PBA is proposed to solve continuous optimization problems and cannot be applied in the discrete community detection problem directly. The details of our adjacent nodes-based discrete search strategy are shown in Algorithm 2, which contains five search modes. Note that all the offspring individuals in each brood population are generated only by the same search mode.

The details of the cutoff operation and search modes 4 and 5 are as follows.

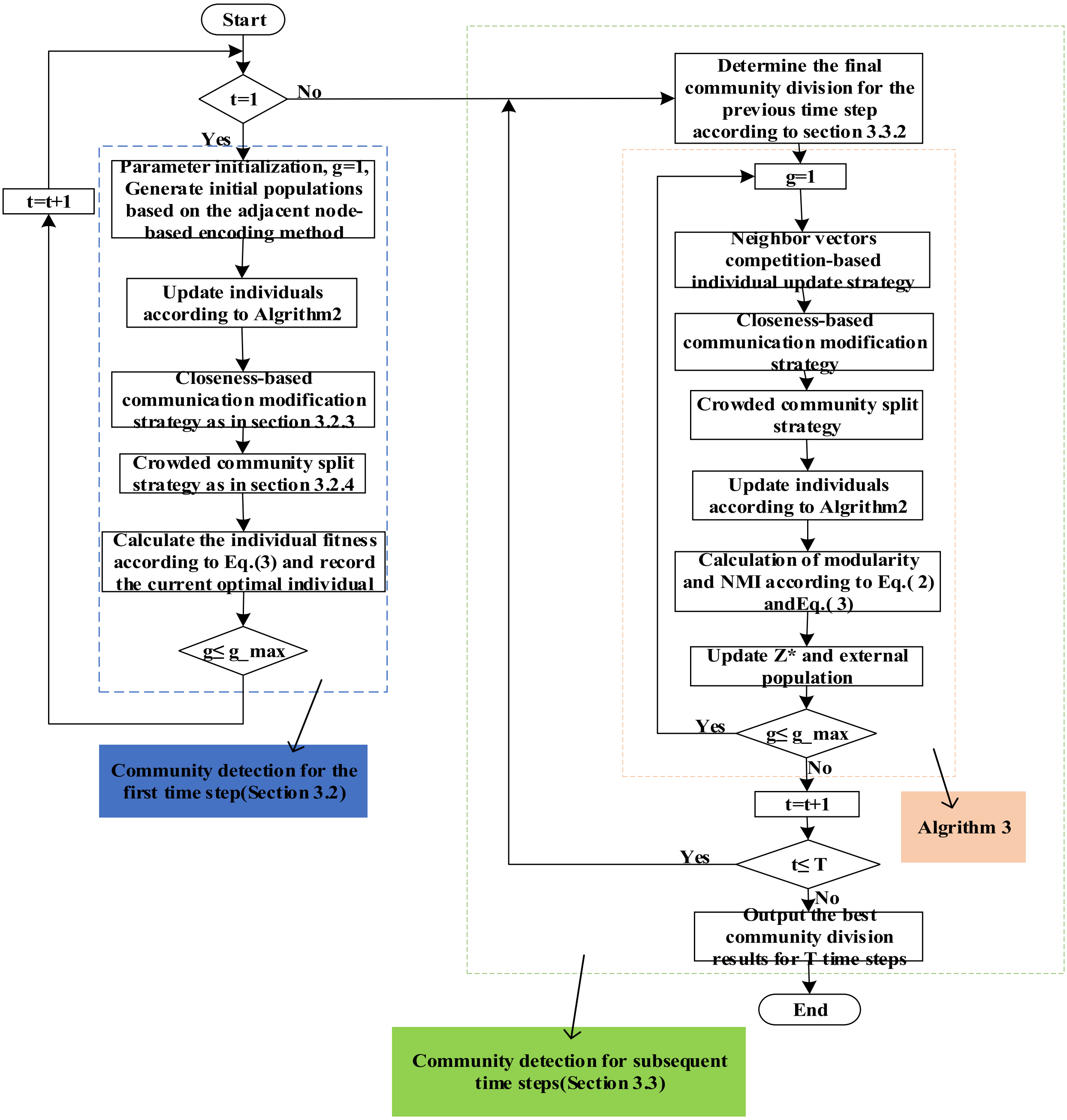

Start
t=1
No
Yes
t=t+1
Parameter initialization, g=1, Generate initial populations based on the adjacent node-based encoding method
Update individuals according to Algrithm2
Closeness-based communication modification strategy as in section 3.2.3
Crowded community split strategy as in section 3.2.4
Calculate the individual fitness according to Eq.(3) and record the current optimal individual
g≤ g_max
Community detection for the first time step(Section 3.2)
Determine the final community division for the previous time step according to section 3.3.2
g=1
Neighbor vectors competition-based individual update strategy
Closeness-based communication modification strategy
Crowded community split strategy
Update individuals according to Algrithm2
Calculation of modularity and NMI according to Eq.(2) andEq.(3)
Update Z* and external population
Yes
g≤ g_max
No
t=t+1
Algrithm 3
Yes
t≤ T
No
Output the best community division results for T time steps
End
Community detection for subsequent time steps(Section 3.3)

FIGURE 3. Framework of the proposed DYN-OPGPBA.

Search Mode 4: A node ***ir*** is selected randomly, and all the neighbour nodes that are not in the community to which the node ***ir*** belongs are found. Then, node ***jr*** is selected according to the roulette wheel selection method from those found neighbour nodes. Finally, the ***ir***-th dimension of the candidate individual ***X*** (***ir***) is replaced by ***jr***.

Search Mode 5: The search modes 5 and 4 are similar. The difference is that the offspring individual is generated by changing the ***jr***-th dimension of the candidate individual ***X*** (***jr***) to ***ir***.

Cutoff Operation: Given an individual ***X*** and its offspring individual ***XX***, the value of some dimension of the individual ***X***(r) is assumed to be not equal to ***XX***(r), and the connected vector to which the node ***r*** belongs is found from two-node set obtained by decoding individual X. Then, all the nodes from the connected vector to which node ***r*** belongs whose dimensions in individual XX are equal to r are selected, and their dimensions in individual X' are changed to the values of the labels of nodes connected to themselves randomly. Fig. 5 shows how our cutoff operator works and compares the results before and after the cutoff operation. Given an

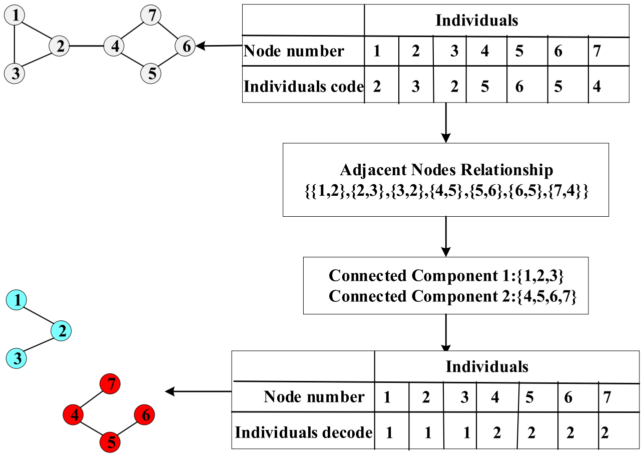
Algorithm 1 The Pseudo-Code of PBA

Input: N_{pop}: population size; D: Dimension of the single-objective optimization problems; $B = \{b_1, b_2, \dots b_k \dots b_D\}$ and $A = \{a_1, a_2, \dots a_k \dots a_D\}$: the bottom and top bounds of the variable X; max_Fes: the maximum number of function evaluations; $pr \in [0, 1]$: control parameter; $f_{nb} \in [0.01, 0.2]$: neighborhood factor; $f_{ms} \in [0.01, 1.00]$: mid-scale factor; $f_{ls} \in [1, 100]$: large-scale factor; $f_{tn} \in [0.005, 0.05]$: fine tuning factor; MEM: memory matrix; N_{broods}: the quantity of broods populations

Output: Minimum value in $f\ (X)$

```
Initialize parameters including $Npop, D, B, A, MEM, max\_FEs,$ $pr, f_{nb}, f_{ms}, f_{ls}, .f_{tn}, FEs = 0$, g = 1,
Initialize the population $X^0 = RST\left(B, A, D, N_{pop}\right)$
$MEM \leftarrow X^0$
Compute the objective vector $F(X^0)$ for each $X \in X0$
$X^0_{birth} \leftarrow arg\ \min(F(X^0))$
While $FEs <= max\_FEs$ do
  for k = 1: $N_{broods}$ do
    if k == 1 %% neighborhood search mode
      $X^g_k = RST\left(x^{g-1}_{birth}(1 - f_{nb}), x^{g-1}_{birth}(1 + f_{nb}), D, N_{pop}\right)$
    elseif $FE > FE_{un}$ %% global search mode
      $X^g_k = RST\left(B, A, D, N_{pop}\right)$
    elseif $\exists j,\ fitness(x^g_{j,k-1}) < fitness(x^{g-1}_{birth})$
        %% medium-scale search mode
      $X^g_k = RST\left(x^{g-1}_{birth}(1 - f_{ms}), x^{g-1}_{birth}(1 + f_{ms}), D, N_{pop}\right)$
    elseif rand < $Pr$ %% large-scale search mode
      $X^g_k = RST\left(x^{g-1}_{birth}(1 - f_{ls}), x^{g-1}_{birth}(1 + f_{ls}), D, N_{pop}\right)$
    else %% memory store search mode
      y ← MEM(randint([1,Npop],1,1),:)
      for $jj = 1{:}D$ do
        y1 ← y;
    $y1(jj) = x_{birth}(jj)^{g-1}(1 - f_{tn}) + rand \times (x_{birth}(jj)^{g-1}(1 + f_{tn}) - x_{birth}(jj)^{g-1}(1 - f_{tn}))$
        fit_xx ← max($F(MEM)$), xx ← $arg$ max($F(MEM)$)
        if fit_y1<fit_xx
          xx ← y1
        end if
      end for
      $X^g_k = MEM$
    end if
  $FEs = FEs + N_{pop}$
  end for
  $POP = X^g_1 \cup X^g_2 \cup \dots\dots \cup X^g_{Nbroads}$
$X^0_{birth} \leftarrow arg(\min(F(POP)))$
g = g + 1;
end while
```

individual $\boldsymbol{X}$ is $\{4, 4, 4, 1, 6, 7, 6, 9, 11, 8, 8, 11\}$, which represents that nodes $\{1, 2, 3, 4\}$ are in the same community with community identifier equal to 1 (the community 1), nodes $\{5, 6, 7\}$ are in the same community with commID equal to 2 (the community 2), and nodes $\{8, 9, 10, 11, 12\}$ are in the same community with commID equal to 3 (the community 3). Assume that when performing the search mode 4 is performed on individual X, X (9) is changed from 11 to 7. Thus, an offspring individual XX is generated, i.e. $XX = \{4, 4, 4, 1, 6, 7, 6, 9, \mathbf{7}, 8, 8, 11\}$, which represents that nodes $\{1, 2, 3, 4\}$ are all in community 1 and nodes $\{8, 9, 10, 11, 12\}$ are all in community 2. Notably although only X (9) is changed in individual $\boldsymbol{X}$, other nodes $\{8, 10, 11, 12\}$ that are in the same community as node 9 are arranged in the other community with commID equal to 2. Obviously, nodes $\{8, 10, 11, 12\}$ are wrongly partitioned with high probability. When cutoff operator is applied to XX, $XX(8)$ is changed from 9 to 11, i.e. $XX = \{4, 4, 4, 1, 6, 7, 6, 11, 7, 8, 8, 11\}$, which represents that nodes $\{1, 2, 3, 4\}$ are in community 1, and nodes $\{5, 6, 7, 8, 9, 10, 11, 12\}$ are in community 2. When the cutoff operator is applied, only the commID of some node is changed, and others still belong to the original communities.



	Individuals						
Node number	1	2	3	4	5	6	7
Individuals code	2	3	2	5	6	5	4

Adjacent Nodes Relationship
{{1,2},{2,3},{3,2},{4,5},{5,6},{6,5},{7,4}}

Connected Component 1:{1,2,3}
Connected Component 2:{4,5,6,7}

	Individuals						
Node number	1	2	3	4	5	6	7
Individuals decode	1	1	1	2	2	2	2

FIGURE 4. Simple illustration of adjacent node-based individual encoding and decoding.

3) CLOSENESS-BASED COMMUNITY MODIFICATION STRATEGY

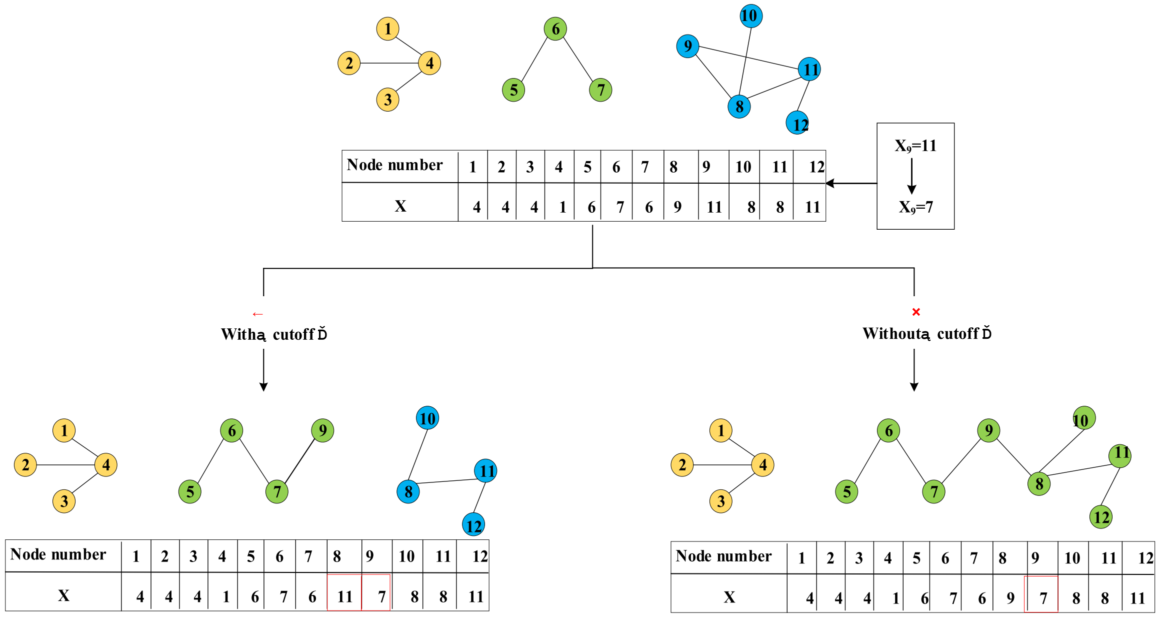
Though the randomness of the adjacent node-based encoding method can help the search modes of Algorithm 2 provide more alternative community divisions, the nodes that should belong to the same community are assigned to different communities with high probability. Therefore, the number of communities obtained by Algorithm 2 might be much larger than the number of real communities.

To address the above issues, we proposed a new closeness-based community modification strategy to correct the unreasonable divisions and enhance their quality. We firstly calculate the closeness between each node and each community according to Eq. (8) and then divide each node into the community with the highest closeness.

$$Cl_i^k = s_k \times \sum \{D\,(j_k)\,|J_i \in C_k\} \tag{8}$$

where Cl_i^k is the closeness of the node i to the k-th community C_k; J_i is the neighbor nodes of the node i; $D(j)$ represents the degree of the node j; j_k is the nodes of J_i belonging to C_k; s_k is the size of j_k.

Fig. 6 gives the community division obtained by performing Algorithm 2 on the Karate network. Compared with the real community division as shown in Fig. 7, node 3 is wrongly assigned to a unreasonable community.



Node number	1	2	3	4	5	6	7	8	9	10	11	12
X	4	4	4	1	6	7	6	9	11	8	8	11

$X_9=11$

$X_9=7$

Witha cutoff Ď

Withouta cutoff Ď

Node number	1	2	3	4	5	6	7	8	9	10	11	12
X	4	4	4	1	6	7	6	11	7	8	8	11

Node number	1	2	3	4	5	6	7	8	9	10	11	12
X	4	4	4	1	6	7	6	9	7	8	8	11

FIGURE 5. Illustration of the cutoff operation.

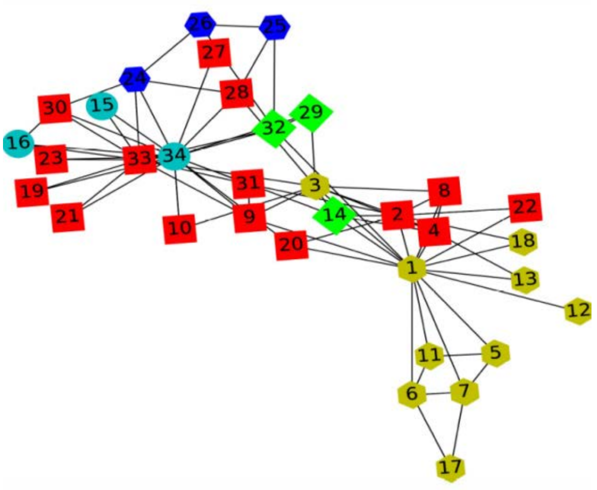


FIGURE 6. Communities identified by Algorithm 2 on Karate network.

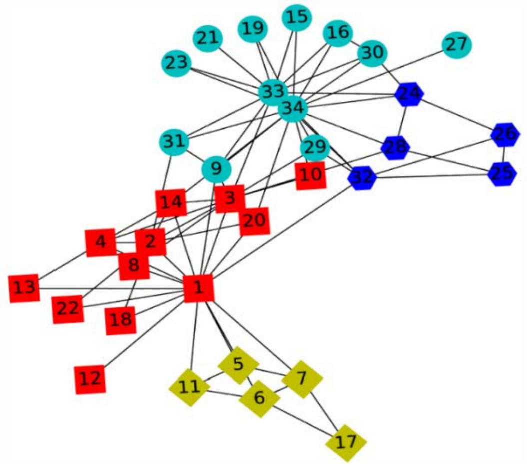


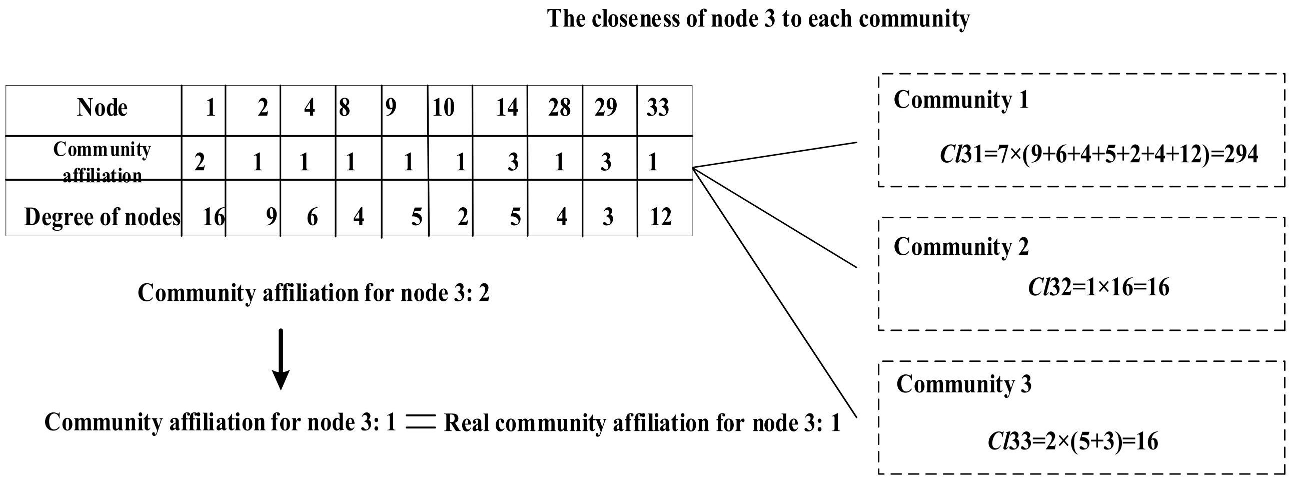
FIGURE 7. The real communities division on Karate network.

We perform our closeness-based communication modification strategy on node 3 as shown in Fig. 8. The details are as follows. First, we identify all the neighbour nodes of node 3, i.e. nodes {1, 2, 4, 8, 9, 10, 14, 28, 29, 33}, where nodes {2, 4, 8, 9, 10, 28, 33} are in community 1, node 1 is in community 2 and nodes {14, 29} are located in another community with commID equal to 3. Then, we calculate the closeness between node 3 and each community according to Eq. (8). We take an example of calculating the closeness between node 3 and community 1. Community 3 contains seven neighbour nodes of the node 3, i.e. nodes {2, 4, 8, 9, 10, 28, 33}, whose degrees are 9, 6, 4, 5, 2, 4 and 12, respectively. Thus, according to Eq. (8), we can obtain $Cl_3^1 = 7\times(9+6+4+5+2+4+12) = 294$. Similarly, we can obtain $Cl_3^2 = 1 \times 16 = 16$, $Cl_3^3 = 2 \times (5 + 3) = 16$, $Cl_3^4 = 0$ and $Cl_3^5 = 0$. Therefore, node 3 is modified into a new community 1 with the highest closeness.

Fig. 9 shows the community divisions after our closeness-based community modification strategy is performed. The network in Fig. 6 has 5 communities, while Fig. 9 contains 4 communities equal to the number of real communities shown in Fig. 7. Node 3 is assigned to the real community 1 from the original community 2, and we can obtain improved partition with a larger modularity value.

4) CROWDED COMMUNITY SPLIT STRATEGY

Two undesirable cases may occur in the community detection: insufficient community division and excessive community division. Insufficient community division means


The closeness of node 3 to each community

Node	1	2	4	8	9	10	14	28	29	33
Community affiliation	2	1	1	1	1	1	3	1	3	1
Degree of nodes	16	9	6	4	5	2	5	4	3	12

Community 1
$CI31=7\times(9+6+4+5+2+4+12)=294$
Community 2
$CI32=1\times16=16$
Community 3
$CI33=2\times(5+3)=16$
Community affiliation for node 3: 2
Community affiliation for node 3: 1 == Real community affiliation for node 3: 1

FIGURE 8. Illustration of closeness-based community modification strategy on node 3.

Algorithm 2 The Pseudo-Code of Adjacent Node-Based Discrete Search Strategy

Input: x_{birth}^{g-1}: the individual with the best modularity in the previous iteration; N_{pop}: the size of X_k^g

Output: $X^g = \left[X_1^g, \ldots, X_k^g\right], k = 1, 2, \ldots, N_{broods}$: offspring population

```
for $k = 1$ to $N_{broods}$ do
  Step 1: Select an search mode
  If $k == 1$
    Search mode = 1;
  elseif FE > FEun
    Search mode = 2;
  elseif $\exists x \in X_{k-1}^g$, $fit(x) \le fit(x_{birth}^{g-1})$
    Search mode = 3;
  elseif r < pr
    Search mode = 4;
  Else
    Search mode = 5;
  End if
  Step 2: generate brood population
  for $i = 1$ to $N_{pop}$ do
    if Search mode == 1 then
      $ir$←a random integer with [1,n]
      $Neighbor\_set$←all the neighbor nodes of node $ir$
      $X_i$←only re-encode nodes $ir$ and $Neighbor\_set$ using adjacent node encoding
    elseif Search mode == 2 then
      $X_i$←regenerate individual by adjacent node encoding
    elseif Search mode == 3 then
      $ir$←a random integer with [1,n]
      $X_i$←only re-encode node $ir$ using adjacent node encoding
    elseif Search mode == 4 then
      $X_i$←generate by search mode 4
      $X_i$←Perform cutoff operation on $X_i$
    else Search mode == 5 then
      $X_i$←generate by search mode 5
      $X_i$←Perform cutoff operation on $X_i$
    end if
  End for
  $X_k^g = X_1 \cup X_2 \cup \ldots \cup X_{Npop}$
End for
```

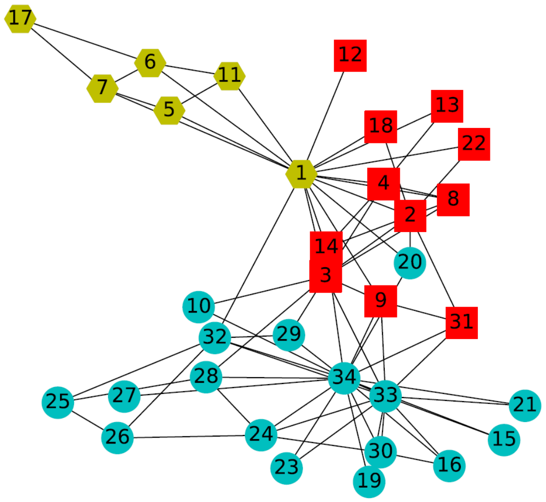


FIGURE 9. Communities identified after closeness-based community modification strategy on Karate network.

the number of communities obtained is greater than the real one, while excessive community division means the number of communities obtained is less than the real one. The insufficient community division can be addressed by Algorithm 2 and our closeness-based community modification strategy. To solve the excessive community division, we propose the following crowded community split strategy. The concepts defined in our crowded community split strategy are listed in Table 2, including the crowded community, reference node, comparison node, reference node, comparison node set and central node set.

Our crowded community split strategy could be performed only when the best individual with the optimal modularity in the population does not change in successive iterations. The concrete steps are as follows. First, we identify the best individual X_{birth} with the optimal modularity. Thus a community structure $\boldsymbol{C}$ can be identified from X_{birth} by performing decoding operation. Then we split the crowded community identified from $\boldsymbol{C}$ into two new split subcommunities as follows. In the first step, the reference node (i.e. node a) is identified and assigned to split

TABLE 2. **The concepts defined in the crowded community split strategy.**

Concept	Description
crowded community	the community with the most nodes
reference node	the first node in the crowded community
comparison node	All the other nodes in the crowded community except for the reference node
reference node set	The nodes are in the intersection of the neighbor nodes of the reference node and the crowded community
comparison node set	The nodes are in the intersection of the neighbor nodes of each comparison node and the crowded community
central node set	The nodes are in the intersection of the reference node set and the comparison node set

sub-community 1. In the second step, for each comparison node (i.e. node b), the number of nodes in its reference node set, comparison node set and central node set is calculated according to Eqs. (9)-(11), respectively (denotes as μ, η_b and θ_{ab}). In the third step, if $\theta_{ab}/\mu \geq \rho or\ \theta_{ab}/\eta_b \geq \rho (\rho$ is equal to 0.6) is satisfied, then most neighbour nodes of node a are the neighbors of node b, and then node b is assigned to split sub-community 1 to which node a belongs to; otherwise, node b is assigned to split sub-community 2.

$$\mu = card\left(\{j\,|(a,j) \in E\} \cap Cr\right) \tag{9}$$

$$\eta_b = card\left(\{j\,|(b,j) \in E\} \cap Cr\right) \tag{10}$$

$$\theta_{ab} = card\left(\{j\,|(a,j) \in E\} \cap \{j\,|(b,j) \in E\} \cap Cr\right) \tag{11}$$

where $card()$ represents the number of elements in the set; E is the set of edges; and Cr is the crowded community.

X_{birth} should be updated, because the connection relationships between the nodes of the original crowded community were changed after the crowded community split strategy. The details are as follows. First, we identify each node that belongs to the split sub-community 1 whose dimension in X_{birth} is equal to the identifier of the node belongs to the split sub-community 2 and change its dimension to the identifier of its neighbour node that belongs to split sub-community 1. Second, we identify each node that belongs to the split sub-community 2 whose dimension in X_{birth} is equal to the identifier of the node that belongs to the split sub-community 1, and change its dimension to the identifier of its neighbour node that belongs to the split sub-community 2.

C. COMMUNITY DETECTION FOR THE SUBSEQUENT TIME STEPS

As shown in Fig. 3, the community detection method for the subsequent time steps except the first time step designed in this paper, consists of two main aspects: a decomposition-based multi-objective pity beetle algorithm (MOEAD/PBA) and the identification method of the optimal community partition at the current time step.

Algorithm 3 MOEAD/PBA for Community Detection at a Single Time Step

Input: $N_{\mathbf{pop}}$: the number of the brood populations; $\lambda_1, \ldots, \lambda^N$: N uniformly distributed weight vectors; $\boldsymbol{wn}$: the number of neighborhood weight vectors for each weight vector

Output: $\boldsymbol{EP}$: external population

```
Initialize parameters ($N_{pop}$, N, wn)
$P = \{X^1, X^2 \ldots X^N\}$←Population_initialization according to the adjacent node-based encoding method
$FV = \{FV(X^1), \ldots, FV(X^N)\}$←Calculate Q and NMI for each X ∈ P
$z*$←Initialize the ideal point
$\lambda = \{\lambda_1, \ldots, \lambda^N\}$←Generate $N$ uniformly distributed weight vectors
$EP = \varphi$, and t = 1;
REPEAT
  $P1$←Perform neighbor vectors competition-based individual update strategy on $P$
  $EP = EP \cup y_{best}$ // $y_{best}$ is the individual with the optimal module density in population $P1$
  $y'_{best}$←Perform closeness-based communication modification strategy on ybest
if $FE > FE_{un}$
  $y'_{best}$←Perform crowded community split strategy on $y'_{best}$
end if
  $P = \{P_1, P_2 \ldots P_{Npop}\}$←Perform adjacent nodes-based discrete search strategy on $y'_{best}$
// update EP
for $i = 1$ to $N_{\mathrm{pop}}$ do
  $X_{best}$←the individual with the optimal module density in the $i$-th brood population y
  update $z*$ individual by comparing the original $z*$ and the $FV(X_{best})$
  for j = 1 to $|B(x_{best})|$ do //$B(x_{best})$ represents the set consist of individuals in the $x'_{best}s$ neighborhood
  if $g^{te}\left(x_{best}\,|\lambda^{B(j)}, z\right) \leq g^{te}\left(B(j)\,|\lambda^{B(j)}, z\right)$
    $x^{B(j)} = x_{best},\quad FV^{B(j)} = FV(xbest)$
  end if
  $EP \leftarrow EP \backslash xx,\ if\ \ xx \in EP,\ and\ \forall xx\ \prec xbest$
  $EP \leftarrow EP \cup xbest$ if there is no $xx\ \in EP$ making $xx\ \prec xbest$
  end for
end for
Perform external population size restriction operation on EP
Until termination criterion met
```

1) MOEAD/PBA FOR COMMUNITY DETECTION AT THE CURRENT TIME STEP

The detailed procedure of our decomposition-based multi-objective pity beetle algorithm (MOEAD/PBA) for community detection at the subsequent time step except for the first time step is shown in Algorithm 3, which contains two main operations: neighbour vector competition-based individual update strategy and external population size restriction mechanism.

a: NEIGHBOUR VECTOR COMPETITION-BASED INDIVIDUAL UPDATE STRATEGY

For the multi-objective evolutionary algorithms based on decomposition, the neighborhood of an individual can make contributions to generate its candidate individual in general. On this basis, we propose the following neighbour vector competition-based individual update strategy to generate the candidate individuals. First, the weight vector λ_i associated with the individual x_i and the individuals associated with the weight vectors with w_n-nearest neighbour distances around λ_i are identified. Then these individuals and the individual i are combined to constitute the competition population of the individual x_i, i.e., $X^i = \left[x^i, x^{i_1}, \ldots, x^{i_{w_n}}\right]^{\mathrm{T}}$. Thus, the j-dimension of a new individual x_i' can be generated by selecting one of the elements of $X^i = \left[x^i, x^{i_1}, \ldots, x^{i_{w_n}}\right]^{\mathrm{T}}$ in column j according to the roulette selection method, where the probability of element k of $X^i = \left[x^i, x^{i_1}, \ldots, x^{i_{w_n}}\right]^{\mathrm{T}}$ in column j is as follows:

$$p_{k,j}^{i} = \frac{\dfrac{D_{in}\left(X_{k,j}^{i}\right)}{D\left(X_{k,j}^{i}\right)}}{\sum\limits_{k=1}^{w_n+1} \dfrac{D_{in}\left(X_{k,j}^{i}\right)}{D\left(X_{k,j}^{i}\right)}} \tag{12}$$

where $P_{k,j}^{i}$ represents the probability of element k of $X^i = \left[x^i, x^{i_1}, \ldots, x^{i_{w_n}}\right]^{\mathrm{T}}$ in column j, $D(X_{k,j}^{i})$ represents the degree of node $X_{k,j}^{i}$ and $D(X_{k,j}^{i})$ represents the number of nodes connected to node $X_{k,j}^{i}$ in the community to which node $X_{k,j}^{i}$ belongs.

b: EXTERNAL POPULATION SIZE RESTRICTION MECHANISM

In Algorithm 3, external population is used to store all the nondominated optimal solutions until the current iteration. At the end of the iterations, we can obtain a large number of local optimal community partitions, especially for the large networks, which are nondominated between each other. Some solutions are relatively dense, whereas others are relatively dispersed. Those relatively dense individuals have searched in their neighborhoods many times, which is why they have difficultly generating candidate individuals with higher values of modularity. The relatively dispersed individuals can generate excellent candidate individuals because they have not been explored deeply. On the basis of the above considerations, we propose the following external population size restriction mechanism. When the size of EP exceeds the threshold EP_{um}, the individuals with EP_{um}-farthest crowding distances are preserved in EP.

2) FINAL OPTIMAL COMMUNITY DIVISION IDENTIFICATION SCHEME

In general, after Algorithm 3 is applied to community detection at each time step, a set of nondominated solutions (PS) can be obtained. The community partitions of the networks at the current time step and the previous time step are required to change slowly, which is why the optimal solution of PS obtained in the previous time step needs to be identified to start with the community detection at the current time step.

The modularity is served to evaluate the snapshot quality in Algorithm 3, which reveals the community structure through comparing the actual density of links in a subgraph with the density one would expect to have in the subgraph if edges fall at random without regard to the community structure. The modularity is the most competitive and widely used criterion function. Unfortunately, it can only detect communities larger than a certain size, i.e. resolution limitation problem. The modularity density described as (4) reveals the community structure by comparing interlink density in all communities and intralink density between different communities, which can solve the so-called resolution limitation problem caused by the modularity. We select the solution with the optimal modularity density from PS as the optimal solution at the previous time step.

The optimal solution at the previous time step is not directly served to detect the community structure of the new network at the current time step, because the network at the current time step is different from the one at the previous time step, such as the number of nodes and the membership. We need to modify the optimal solution at the previous time step to adapt the detection of the network at the current time step. The details are as follows. Firstly, the deleted nodes in the new network at the current time step are removed from the optimal solution. Then, the nodes added and the nodes that are not connected to the according dimensions in the optimal solution in the new network at the current time step are identified and subjected to adjacent node-based encoding.

D. TIME COMPLEXITY

In this work, n represents the number of nodes, T indicates the size of EP, k shows the average number of each node's neighbour communities and NP indicates the population size. The proposed DYN-OPGPBA contains two separate community detection stages for the first time step and the subsequent time steps, and the time cost of the latter is greater than that of the former. Therefore, the time complexity of the community detection algorithm for the subsequent time steps will be analyzed in this section. The community detection algorithm for the subsequent time steps contains two separate stages. The first stage is Algorithm 3, and the second stage is shown in the final optimal community division identification scheme.

(1) In Algorithm 3, the time complexity cost mainly relies on five strategies in the main loop, namely, neighbour vector competition-based individual update strategy (shown in line 7), closeness-based community modification strategy (shown in line 9), crowded community split strategy (shown in line 11), adjacent node-based discrete search strategy (shown in line 13) and updating strategy on EP (shown in lines 14 to 24). Their time complexity cost are $O(T{*}NP{*}n)$, $O(c{*}NP{*}n)$, $O(NP{*}n)$, $O(NP{*}n)$ and $O(NP{*}n)$, respectively. Therefore, the total time complexity cost of Algorithm 3 can be $O(T{*}NP{*}n)$.

(2) The final optimal community division identification scheme needs $O(NP^2)$ computation time.

Therefore, the worst overall time complexity cost of this algorithm in each iteration is simplified as $O(NP^2) + O(T{*}NP{*}n)$.

IV. EXPERIMENTAL STUDY

In this section, two experiments are conducted to fully evaluate the performance of our proposed DYN-OPGPBA. We verify the effectiveness of the community detection of CD-OPGPBA, because the detection quality of the network at the first time step has a greater effect on the community detection of the dynamic networks, compared with that of the network at subsequent time steps. The effectiveness of the proposed DYN-OPGPBA is then verified. The proposed algorithm is implemented in Matlab 2016. All the tests are conducted on a personal computer equipped with a core i5 CPU (1.6GHz) and a 8.0-GB memory.

A. VERIFICATION OF THE DETECTION EFFECT OF CD-OPGPBA

In this section, the performance of the proposed CD-OPGPBA is verified on artificial synthetic networks, small-scale real-world networks and real-world networks.

In our experiments, our proposed CD-OPGPBA uses the same parameter setting on all test networks as follows. The maximum number of iterations Tmax is set to 800, the population size NP is set to 20, the number of the nodes selected randomly is set to10 (nod = 10), pr = 0.6, $\rho = 0.6$ in the crowded community split strategy.

1) RESULTS ON ARTIFICIAL SYNTHETIC NETWORKS

In this section, a number of synthetic networks produced by the Lancichinetti-Fortunato-Radicchi (LFR) [25] mode are generated to further evaluate the performance of CD-OPGPBA and the compared approaches, where the distribution of node degrees and size of communities are both power laws with tuneable exponents, thus making it much closer to real-world networks. The mixing parameter *mu* determines the fraction of edges between each node and its neighbours in other communities, and a bigger value of *mu* indicates an unclear corresponding community structure of the network and more difficult community detection. In our experiments, six different LFR networks are generated with the following settings: Each network contains 128 nodes, the average degree is fixed to 16, the community size is set to 32, and the value of *mu* varies from 0.1 to 0.6 with an interval of 0.1.

We choose six typical high-performance modularity optimization algorithms for comparison, including ECSD [26], FN [27], GN [28], Meme-net [29], MAGA-net [30], and walktrap [31]. The parameters of the other compared algorithms are set the same as in the original literature. For each network, the performance of CD-OPGPBA is evaluated and compared with that of the six competitors. The experimental results are shown in Fig. 10, where each data point represents the best value of Q achieved by each algorithm on each LFR network in a random run.

Fig. 10 shows that, compared with the six other community detection algorithms, CD-OPGPBA can always obtain the best value of Q on all the six different LFR networks. We can also find that, for the networks with small values of *mu* ($mu \leq 0.3$), the performance of CD-OPGPBA is the same or slightly better than that of the competitors. However, with the growth of *mu*, the superiority of CD-OPGPBA over other competitors becomes much more significant. Especially when *mu* is equal to 0.5, the performance of each algorithms have the greatest differences. When $mu = 0.6$, the performance of CD-OPGPBA is the same or slightly better than that of FN and walktrap, because the community structure of the network is too vague to detect communities.

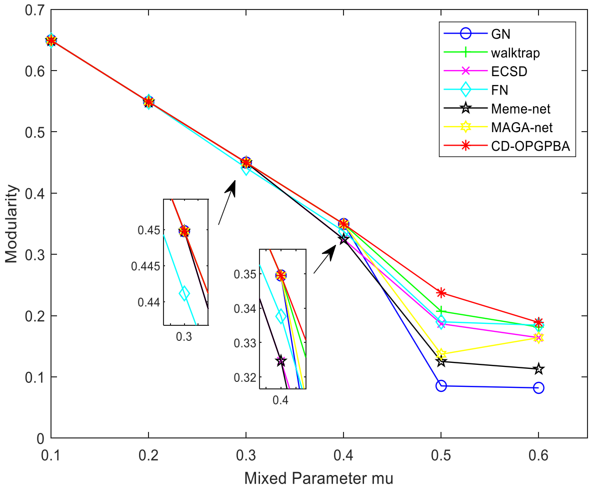


FIGURE 10. Best modularity of CD-OPGPBA and six competitors on a set of LFR networks with mu increasing from 0 to 0.6 with an interval of 0.1.

TABLE 3. Characteristics of 4 small-scale real-world networks.

Networks	Number of nodes	Number of edges	Number of communities	Clustering coefficient	Average degree
Karate	34	78	[2,10]	0.25	4.69
Dolphin	63	318	[2,10]	0.26	5.13
polBooks	105	882	[3,10]	0.49	8.40
Football	115	616	[5,15]	0.40	10.67

2) RESULTS ON SMALL-SCALE REAL-WORLD NETWORKS

To further investigate the performance of our CD-OPGPBA, we try to verify its effectiveness on four widely used small-scale real-world networks: the Zachary's karate club network (Karate), the Dolphin network (Dolphins), the American political books network (polBooks) and the American college football network (Football). Their unique topological information [7] is shown in Table 3.

The performance of CD-OPGPBA is compared with many state-of-the-art static community detection methods, including ECSD [26], FN [27], GN [28], Meme-net [29], walktrap [31], CNM [32], BGLL [33], MSFCM [34], FMM/H1 [35], Informap [36], LAP_CL [37], TNS-LPA [38], ECES [7], MAGA-net [30], GATHB [39], MOGA [40], ECGA [41], CC-GA [42], MOEA/D [43], DBA [44], ACODCS [45], MPSOA [46], MODPSO [47], DECD [48], CCDECD [49], IDDE [50], CDMFOA [51], Com-MOEA/D [57], CDEMO [52], MDSTA [53], and SOSCD [54]. Here, the first 13 community detection methods adopt the traditional methods to optimize the modularity function, and the other 17 community detection methods adopt the meta-heuristics evolutionary optimization approaches to optimize the modularity function, where GA is employed as an optimization strategy by MAGA-net, GATHB, MOGA, ECGA, CC-GA, and MOEA/D, DE is employed as an optimization strategy by DECD, CCDECD, IDDE, and CDEMO, and BA, ACO, PSO, FOA, STA, and SOS is employed as optimization strategies by the rest of the algorithms, respectively.

Table 4 records the highest value of Q obtained by each algorithm. Where the results of the following competitors are collected from literature [54], including BGLL, MSFCM, FMM/H1, Informap, ECES, GATHB, MOGA, ECGA, CC-GA, MOEA/D, DBA, ACODCS, MPSOA, MODPSO, DECD, CCDECD, IDDE, CDMFOA, CDEMO, MDSTA and SOSCD, the results of LAP_CL, TNS-LPA and ECES are taken from literature [37], [38], [7], respectively, and the other results that could not be found from literature are obtained by running the codes of each algorithm, including GN, FN, Walktrap, Meme-net, ESCD and MAGA-net.

Table 4 shows that those traditional and classical community detection algorithms can identify certain community structures of small-scale real-world networks, and their detection accuracy is not too high. Meme-net obtains the best value of Q equal to 0.4198 on the Karate network only; this value is the highest Q that we obtained. On the other networks (including Dolphins, polBooks, and Football), none of the traditional and classical community detection algorithms can obtain the best value of Q. Compared with the traditional and classical community detection algorithms, the detection algorithms based on meta-heuristics evolutionary optimization approaches obtain better results. For instant, on the Karate network, 12 methods obtain the best value of Q (Q = 0.4198) that is the highest Q we have known, including MAGA-net, ECGA, CC_GA, MPSOA, MODPSO, DECD, CCDECD, IDDE, CDMFOA, CDEMO, MDSTA, SOSCD, and CD-OPGPBA. On the Dolphins network, 7 methods obtain the best value of Q (Q = 0.5285) that is the highest Q we have known, including MAGA-net, CC_GA, CDMFOA, CDEMO, MDSTA, SOSCD, and CD-OPGPBA. On the polBooks network, 4 methods obtain the best value of Q (Q = 0.5272) that is the highest Q we have known, including CDMFOA, MDSTA, SOSCD, and CD-OPGPBA. On the Football network, 8 methods obtain the best value of Q (Q = 06046) that is the highest Q we have known, including MAGA-net, DECD, CCDECD, IDDE, CDEMO, MDSTA, SOSCD, and CD-OPGPBA. Moreover, the detection algorithms based on FOA, STA, SOS and PBA perform better than those with GA. Only SOSCD and our proposed CD-OPGPBA obtained the best results on all the tested small-scale real-world networks.

TABLE 4. Highest values of Q obtained BY Classical Algorithms and meta-heuristics on small-scale real-world networks.

Algorithms	Karate	Dolphins	Polbooks	Football
GN[28] (2002)	0.4013	0.5194	0.5168	0.5996
FN[27] (2004)	0.3807	0.4820	0.5020	0.5497
Walktrap[31] (1997)	0.3532	0.4872	0.5070	0.6029
Meme-net[29] (2011)	**0.4198**	0.5025	0.5173	0.5492
ECSD[26] (2006)	0.3934	0.4912	0.4672	0.4926
CNM[32] (2004)	0.3807	0.4950	0.5019	0.5770
BGLL[33] (2008)	0.4150	0.4950	0.5150	0.6010
MSFCM[34] (2013)	0.4231	0.3991	0.4601	0.5268
FMM/H1[35] (2015)	0.3941	0.4882	0.5175	0.5960
Infomap[36] (2008)	0.401	0.527	0.522	0.600
LPA_CL[37] (2022)	0.372	0.483	0.451	0.546
TNS-LPA[38] (2020)	0.3715	0.5187	0.4256	0.6010
ECES[7] (2018)	0.371	0.501	0.530	0.601
MAGA-net[30] (2005)	**0.4198**	**0.5285**	0.5272	**0.6046**
GATHB[39] (2011)	0.4024	0.5219	0.5176	0.5508
MOGA[40] (2012)	0.4160	0.5050	0.5180	0.5220
ECGA[41] (2013)	**0.4198**	0.5242	0.5269	0.6010
CC-GA[42] (2018)	**0.4198**	**0.5285**	0.5270	0.5940
MOEA/D[43] (2012)	0.3715	0.3735	0.5180	0.6005
Com-MOEA/D[57] (2020)	0.38	0.52	0.53	0.60
DBA[44] (2016)	0.4028	0.5277	0.5265	0.5990
ACODCS[45] (2018)	0.4165	0.5268	0.5262	0.6031
MPSOA[46] (2016)	**0.4198**	0.5191	0.5255	0.6030
MODPSO[47] (2014)	**0.4198**	0.5268	0.5260	0.6032
DECD[48] (2012)	**0.4198**	0.5249	0.5262	**0.6046**
CCDECD[49] (2012)	**0.4198**	0.5216	0.5268	**0.6046**
IDDE[50] (2015)	**0.4198**	0.5282	0.5271	**0.6046**
CDMFOA[51] (2016)	**0.4198**	**0.5285**	**0.5272**	0.6033
CDEMO[52] (2018)	**0.4198**	**0.5285**	0.5271	**0.6046**
MDSTA[53] (2019)	**0.4198**	**0.5285**	**05272**	**0.6046**
SOSCD[54] (2019)	**0.4198**	**0.5285**	**05272**	**0.6046**
CD-OPGPBA	**0.4198**	**0.5285**	**05272**	**0.6046**

TABLE 5. Characteristics of 3 large-scale real-world networks.

Networks	Number of nodes	Number of edges	Clustering coefficient	Average degree	Maximum degree
Euroroad	1174	2834		2.41	10
NetScience	1589	5484	0.88	3.45	34
PGP	10680	24316	0.44	4.60	205

3) RESULTS ON LARGE-SCALE REAL-WORLD NETWORKS

To further investigate the detection ability of CD-OPGPBA on the large-scale real-world networks, three well-known large-scale real-world networks are tested: the NetScience network, the Euroroad network and the PGP network. Their unique topological information [55] is shown in Table 5.

Experiments proved that most community detection methods can obtain the excellent performance on some networks and may lose their effectiveness on other networks. Therefore, different methods will be chosen as the competitors of CD-OPGPBA on different networks.

For NetScience network, 14 well-known community detection methods are chosen as the competitors, including

TABLE 6. Maximum and average values of Q obtained by CD-OPGPBA and the competitors on the NetScience network.

algorithms	Q_{best}	Q_{mean}
Meme-net[29] (2011)	0.9580	0.9553
MAGA-net[30] (2005)	0.9560	0.9357
GN[28] (2002)	0.9257	0.9257
Walktrap[31] (1977)	0.9559	0.9559
ECSD[26] (2008)	0.9515	0.9515
FN[27] (2004)	0.9551	0.9551
LPA[59] (2007)	0.9255	0.9197
BGLL[33] (2008)	0.9597	0.9597
CC-GA[42] (2018)	0.9580	0.9550
FSA[60] (2019)	0.9585	0.9585
MDP[56] (2015)	0.8506	NA
Com-MOEA/D[57] (2020)	0.93	NA
LPA_CL[37] (2022)	0.9480	0.9480
SOSCD[54] (2019)	**0.9599**	0.9597
CD-PBA	0.9592	0.9567

ECSD [26], FN [27], GN [28], Meme-net [29], MAGA-net [30], walktrap [31], MDP [56], LPA [59], BGLL [49], CC-GA [42], FSA [60], Com-MOEA/D [57], LPA_CL [37] and SOSCD [54]. Table 7 records the maximum and average values of Q (i.e. Q_{best} and Q_{mean}) obtained by each algorithm. Where the results of CC-GA, FSA, LPA, BGLL and SOSCD are collected from the literature [54], the results of MDP, Com-MOEA/D and LPA-CL are respectively collected from the literature [56], [57], [37], and other results were obtained by running the codes of each algorithm for 15 times. Note that the symbol "NA" indicates that the results of the corresponding algorithms didn't given in the corresponding literature.

Table 6 shows the proposed CD-OPGPBA obtains the slightly less Q ($Q = 0.9592$) than that of SOSCD ($Q = 0.9599$) and BGLL ($Q = 0.9597$). However, SOSCD and CD-OPGPBA cost 20000 and 16000 function evaluation times, respectively. It indicates that the performance of CD-OPGPBA is similar to SOSCD on the NetScience network. Moreover, the proposed CD-OPGPBA outperforms other competitors, including ECSD, FN, GN, Meme-net, MAGA-net, walktrap, MDP, LPA, CC-GA, FSA, MDP, Com-MOEA/D, and LPA_CL.

For the Euroroad network, seven well-known community detection methods are chosen as the competitors, including ECSD [26], FN [27], GN [28], Meme-net [29], MAGA-net [30], walktrap [31], and MDP [56]. Table 7 records Q_{best} and Q_{mean} obtained by each algorithm. The results of MDP were collected from the literature [56], and other results were obtained by running the codes of each algorithm for 15 times.

Table 7 shows that CD-PBA obtains a partition with higher Q_{best}. It is only less than that of FN and MDP which obtain the highest Q_{best} value of 0.8722, which indicates CD-PBA outperforms Meme-net, ECSD, GN, MAGA-net, and walktrap on the NetScience network.

TABLE 7. Maximum and average values of Q obtained by CD-OPGPBA and the competitors on the Euroroad network.

algorithms	Q_{best}	Q_{mean}
Meme-net[29] (2011)	0.8452	0.8409
MAGA-net[30] (2005)	0.8316	0.7954
GN[28] (2002)	0.7995	0.7995
Walktrap[31](1997)	0.8374	0.8374
ECSD[26] (2008)	0.8546	0.8546
FN[27] (2004)	**0.8722**	0.8722
MDP[56] (2015)	**0.8722**	NA
CD-PBA	0.8591	0.8505

TABLE 8. Maximum and average values of Q obtained by CD-OPGPBA and the competitors on the PGP networks.

algorithms	Q_{best}	Q_{mean}
LPA[55] (2008)	0.8192	0.8055
CC-GA[42] (2018)	0.8520	0.8430
FSA[60] (2019)	0.8717	0.8717
CDEP[58] (2020)	0.7865	NA
FUC[59] (2007)	0.8474	NA
SOSCD[54] (2019)	0.8796	0.8794
MDP[56] (2015)	0.8832	NA
CDOPG-PBA (Tmax=800)	0.8197	0.8191
CD-OPGPBA (Tmax=900)	0.8231	0.8240
CD-OPGPBA (Tmax=1000)	0.8271	0.8279

To further investigate the detection ability of CD-OPGPBA on the large-scale real-world networks with more than 10000 nodes, the PGP network is conducted. 7 well-known community identify methods are chosen as the competitors, including LPA [55], CC_GA [42], FSA [60], CDEP [58], FUC [59], SOSCD [54] and MDP [56]. Table 8 records Q_{best} and Q_{mean} obtained by each algorithm. Where the results of LPA, CC_GA, FSA and SOSCD are collected from the literature [54], the results of CDEP, MDP and FUC are respectively taken from the literature [58], [55], [56], and the results of CD-OPGPBA are obtained by run its codes until the maximum iterations reach to 800, 900 and 1000, respectively.

As we can see from Table 8, some conclusions can be obtained as follows. First, Q_{best} of the proposed CD-OPGPBA only higher than that of LPA and CDEP. It indicated that the performance of CD-OPGPBA only outperforms LPA and CDEP, and is worse than other competitors; Second, the best value of Q obtained by CD-OPGPBA has been still increasing, as the number of iterations increases. Moreover, the best values of Q obtained in each iteration change rapidly at the beginning of evolution, and increase slowly at the end of evolution. In particular, the best value of Q obtained by CD-OPGPBA will be still increased when the iteration number is set to a value which is higher than 1000.

In sum, the proposed CD-OPGPBA is an effective community detection method for small-scale networks, and

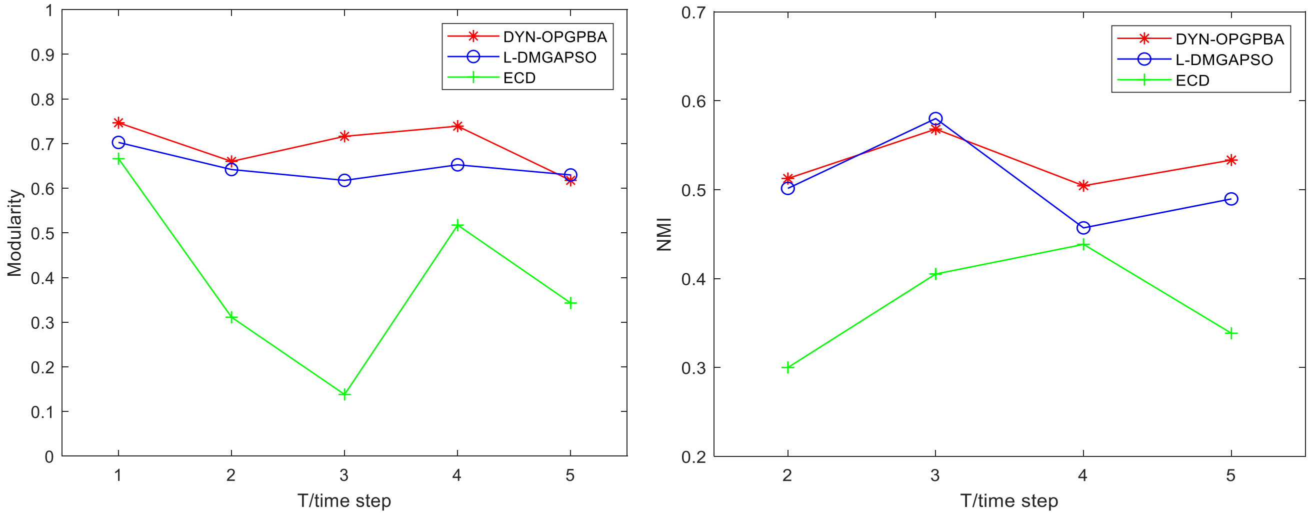


FIGURE 11. Comparison of DYN-OPGPBA, L-DMGAPSO, and ECD in terms of modularity and NMI on the infectious social patterns network.

outperforms many state-of-the-art community detection methods. Whereas the performance of CD-OPGPBA decreases with the increase of network scale. However, CD-OPGPBA only is the first stage of the proposed dynamic community detection method which adopts the frame of evolutionary clustering, and the evolutionary clustering-based methods for the dynamic community detection usually are used to sever for the small-scale dynamic networks. Therefore the performance of the proposed CD-OPGPBA on the small-scale networks is more important.

B. VERIFICATION OF THE DETECTION EFFECT OF THE DYNAMIC NETWORK

In this section, the performance of the proposed DYN-OPGPBA is validated by comparing it with that of two state-of-the-art dynamic community detection approaches: L-DMGAPSO (Label-based Dynamic Multi-objective Genetic Algorithm Particle Swarm Optimization) [8] and ECD (Evolutionary Community Detection) [9]. The comparisons are conducted on artificial synthetic dynamic networks and well-studied real-life dynamic networks.

To ensure a fair comparison, the populations utilized by different community detection approaches are initialized with the same size $NP = 20$ and the same termination criterion (i.e. the maximum number of iterations) $T\max = 800$. The other parameters of each algorithm are set as follows: In DYN-OPGPBA, the number of the randomly selected nodes is set to 10 (nod = 10), pr = 0.6, $\rho = 0.6$ in the crowded community split strategy, and the number of neighbourhood weight vectors in Algorithm 3 $w_n = 4$. In L-DMGAPSO, according to the author's suggestion, the crossover probability and the mutation probability are set to 0.9 and 0.1, respectively. The inertia weight $\omega = 0.7298$ and learning factor $s1 = s2 = 1.4961$. In ECD, according to the author's suggestion, community connection threshold R is set to 0.2, the selection probability, the mutation probability and the immigration probability are set to 0.5, 0.2 and 0.5, respectively.

TABLE 9. Characteristics of infectious social patterns network at five time steps.

Time Steps	Time	nodes	edges
1	2009-04-28	206	1428
2	2009-04-29	222	1628
3	2009-04-30	201	1318
4	2009-05-01	220	1160
5	2009-05-02	217	1934

1) RESULTS ON REAL-LIFE DYNAMIC NETWORKS

In this section, a real-life Infectious Social Patterns dynamic network is adopted to test the performance of our algorithm. This network was built by Isella *et al.* by tracking science museum visitors [30], where the vertices represent the visitors and the link represents the close encounter relationships of visitors. Our experiment focuses on visitors over five days from 28 April to 2 May 2009. To perform daily analysis, the data are divided into five subsets, one for each day. Some statistical information regarding the network used in our experiments are given in Table 9.

Fig. 11 depicts the performance of our algorithm and the compared algorithms on the above dynamic Infectious Social Patterns network, respectively, where each data point represents the value obtained by each algorithm at each time step in a random run. Fig. 11(a) plots the modularity values of the competing algorithms. Compared with the L-DMGAPSO, the modularity obtained by the DYN-OPGPBA is slightly smaller at the fifth time step, slightly larger at the second time step and significantly larger at the other time steps. In addition, the values of modularity obtained by DYN-OPGPBA are higher than those of ECD. DYN-OPGPBA is slightly better than L-DMGAPSO and

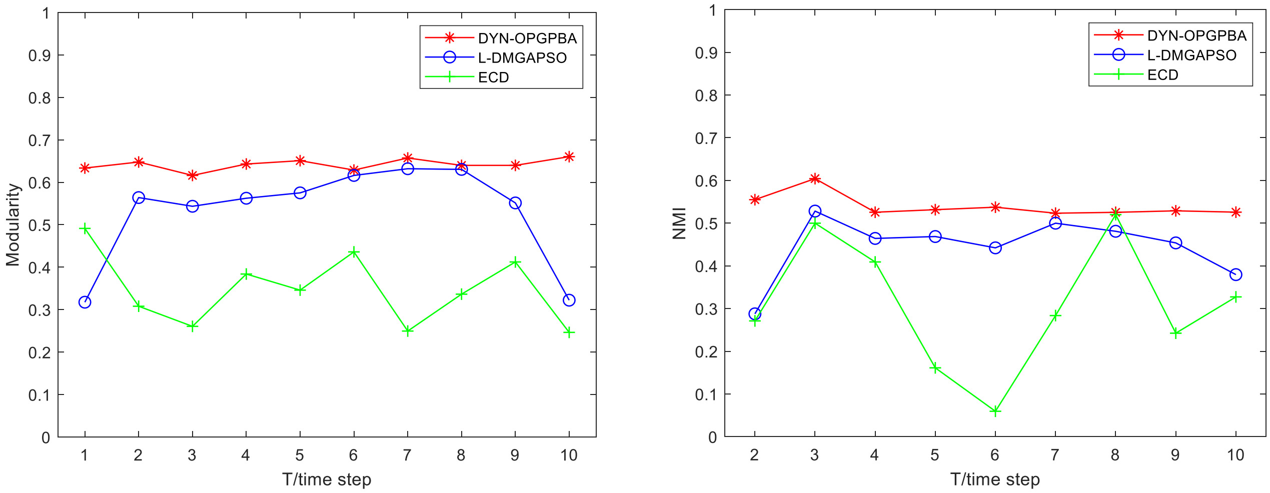


FIGURE 12. Comparison of DYN-OPGPBA, L-DMGAPSO, and ECD in terms of modularity and NMI on the dynamic synthetic networks with mu = 0.1.

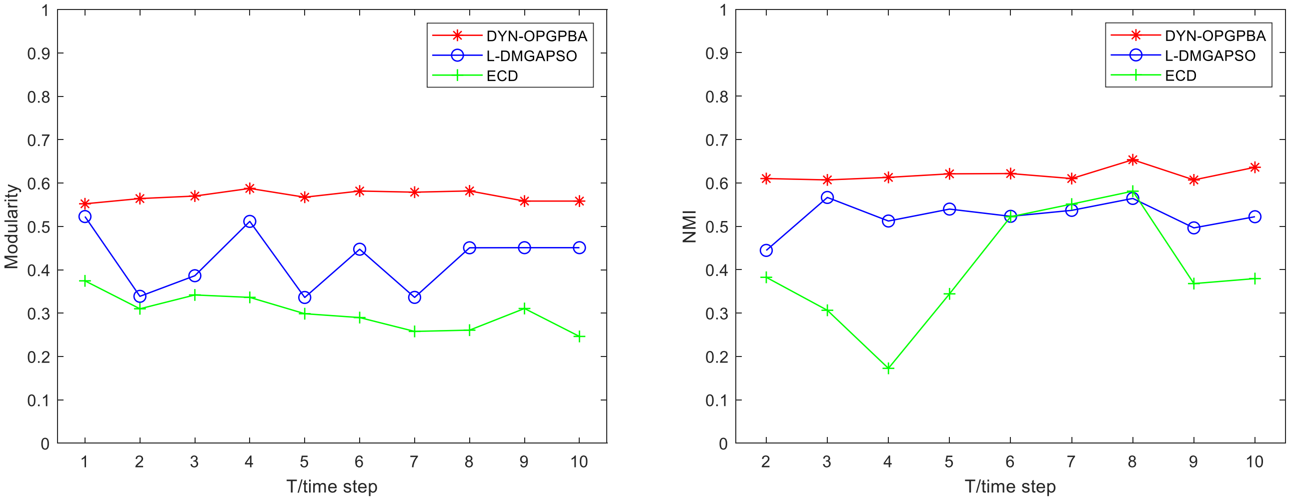


FIGURE 13. Comparison of DYN-OPGPBA, L-DMGAPSO, and ECD in terms of modularity and NMI on the dynamic synthetic networks with mu == 0.2.

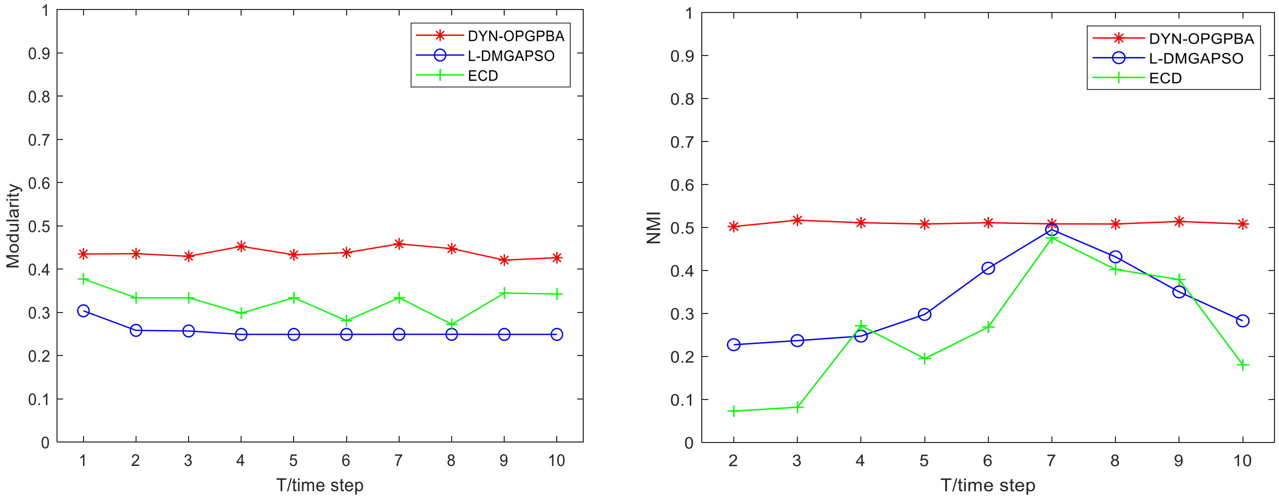


FIGURE 14. Comparison of DYN-OPGPBA, L-DMGAPSO, and ECD in terms of modularity and NMI on the dynamic synthetic networks with mu = 0.3.

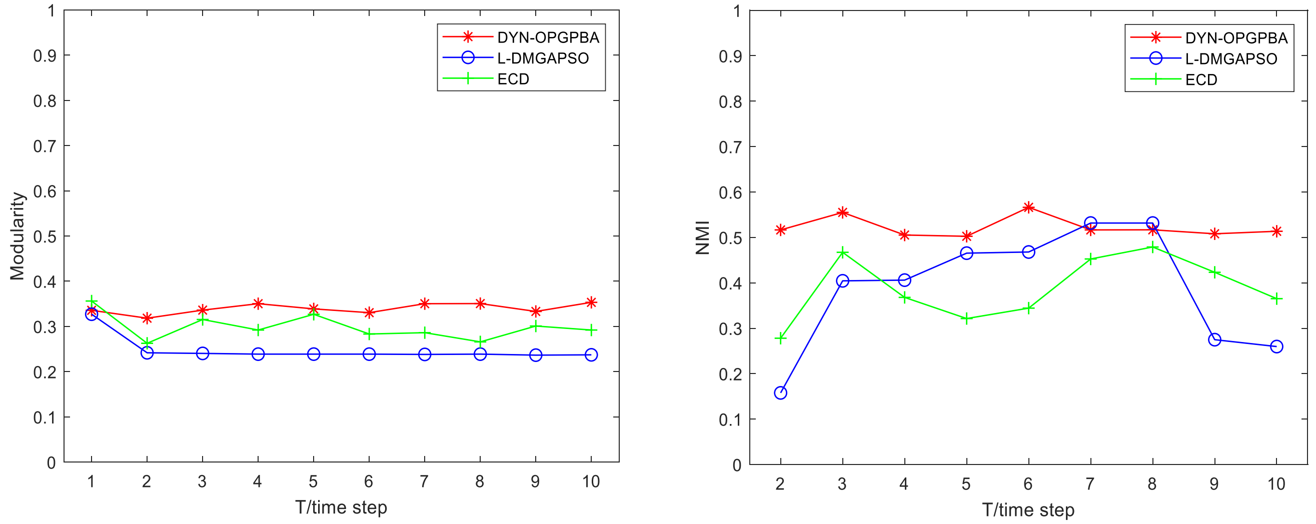


FIGURE 15. Comparison of DYN-OPGPBA, L-DMGAPSO, and ECD in terms of modularity and NMI on the dynamic synthetic networks with mu = 0.4.

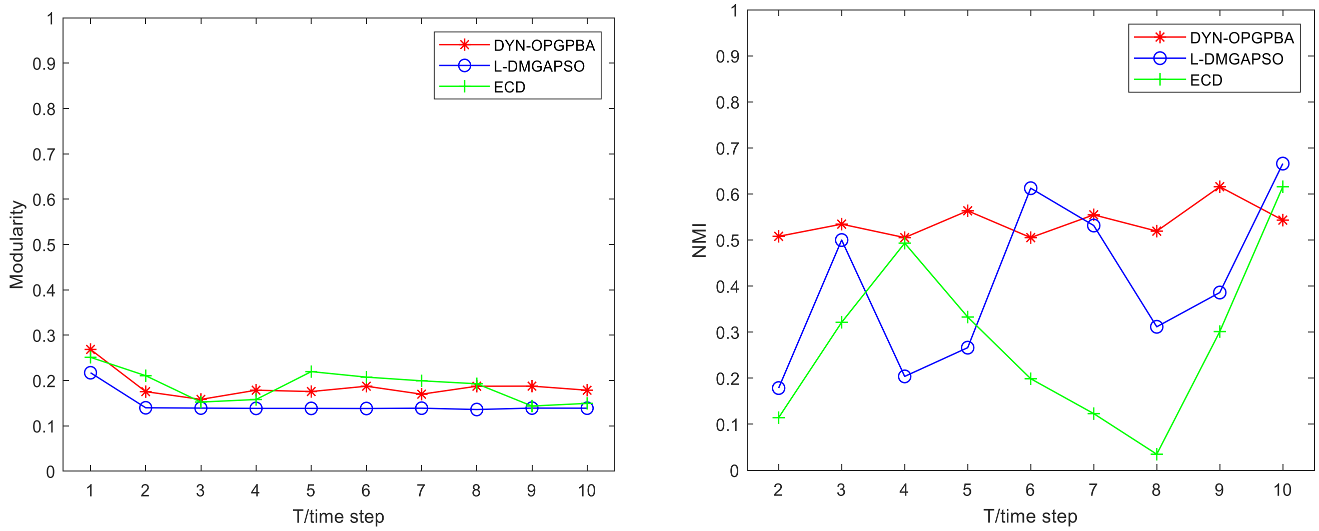


FIGURE 16. Comparison of DYN-OPGPBA, L-DMGAPSO, and ECD in terms of modularity and NMI on the dynamic synthetic networks with mu = 0.5.

significantly better than ECD in terms of snapshot quality according to the statistical values of modularity. Fig. 11(b) plots the NMIs of the competing algorithms. The community structures discovered by DYN-OPGPBA have higher NMIs at each time step than those of ECD, which indicates that DYN-OPGPBA can discover smoother community structures than ECD. Compared with L-DMGAPSO, NMIs obtained by the DYN-OPGPBA are higher at three out of four time steps except for the second time step. This condition is due to the difference between the optimal community division at the second time step and the original community structure at the third time step is obvious, because compared with the network at the second time step, many nodes and edges are deleted at the third time step. In general, DYN-OPGPBA can contribute more to smoothing the community structure between two consecutive time steps than ECD and L-DMGAPSO.

In sum, DYN-OPGPBA is slightly better than L-DMGAPSO and significantly better than ECD for dynamic community detection on infectious social patterns networks.

2) RESULTS ON ARTIFICIAL SYNTHETIC DYNAMIC NETWORKS

In this section, we build five dynamic synthetic networks to further evaluate the performance of our proposed DYN-OPGPBA and the compared algorithms. Each dynamic synthetic dataset contains 10 time steps, where each original synthetic network at the first time step is respectively generated by LFR with $mu = 0.1, 0.2, 0.3, 0.4$ and 0.5, while the networks at other time steps are obtained by evolution events that may characterize the evolution of dynamic networks. The evolution events can be described as follows: 20% of the nodes in each community are selected randomly to be assigned to an another community randomly, and the edges between these nodes and other nodes in the new community are regenerated according to the mixing parameter *mu*.

Figs. 12-16 depict the performance of our algorithm and the compared algorithms on the five dynamic synthetic networks, where each data point represents the value obtained by each algorithm at each time step in a random run.

As mentioned above, the community structures of the networks with $mu = 0.1$, 0.2 and 0.3 are obvious, whereas those of the networks with $mu = 0.4$ and $mu = 0.5$ are vague. As shown in Figs. 12(a), 13(a) and 14(a), the values of modularity obtained by DYN-OPGPBA on the networks with $mu = 0.1$ at the sixth and eighth time steps are slightly larger than those obtained by the other two algorithms. Moreover, DYN-OPGPBA outperforms the compared algorithms in terms of snapshot quality according to the statistical values of modularity. As shown in Figs. 12(b), 13(b) and 14(b), DYN-OPGPBA can always obtain the best NMIs on all 10 time steps. As shown in Fig. 15(a), with the growth of *mu*, the statistical values of modularity obtained by all algorithms decrease. However, compared with its competitors, DYN-OPGPBA performs the best performance at 9 out of 10 time steps except for the first time step. Fig. 15(b) demonstrates that except for its slightly poorer performance than that of L-DMGAPSO at the seventh and eighth time steps, DYN-OPGPBA can always obtain the best NMIs. As shown in Fig. 16, the performance of DYN-OPGPBA is statistically similar to that of ECD in terms of snapshot quality according to the statistical values of modularity, while the performance of DYN-OPGPBA is statistically significantly better than that of ECD in terms of temporal cost according to NMIs. Moreover, the temporal cost and snapshot quality of DYN-OPGPBA are statistically significantly better than those of L-DMGAPSO.

V. CONCLUSION

This paper proposes a novel dynamic community detection method that based on the framework of evolutionary clustering and is called DYN-OPGPBA. It consists of two steps. In the first step, the PBA is improved to maximize the modularity for the community detection of the first time step. Then, with the influence of the community affiliation of nodes and their neighbouring nodes' community affiliation on the division quality considered fully, the evolutionary strategy of PBA is discretized, and a discrete search strategy based on neighbouring nodes is proposed. A closeness correction strategy and a crowded community splitting strategy are proposed to standardize the community division of nodes further. Tested on 12 general networks and compared with many representative static community detection methods, the proposed algorithm obtains greater modularity, indicating its superior detection accuracy and reducing the adverse effects caused by error accumulation. In the second step, for community detection at subsequent time steps, modularity and NMI are chosen as optimization objectives. Modularity ensures the community quality of the current time step, and NMI ensures the temporal smoothness of the network. Then, for the decomposition-based multi-objective framework, a neighbourhood vector competition strategy and an external population size restriction mechanism are proposed, along with solving the modularity and NMI entropy. A method is also proposed to determine the final community division results for each time step according to the modularity density, which is used as the starting point for community division in the next time step. Tested on one real dynamic network and five artificial synthetic dynamic networks, DYN-OPGPBA provides a better balance between snapshot quality and temporal cost than two other more representative dynamic community detection methods.

DATA AVAILABILITY

The codes and the data used to support the findings of this study are available from the corresponding author upon request.

REFERENCES

[1] P. Wagenseller, F. Wang, and W. Wu, "Size matters: A comparative analysis of community detection algorithms," *IEEE Trans. Computat. Social Syst.*, vol. 5, no. 4, pp. 951–960, Dec. 2018, doi: 10.1109/TCSS.2018.2875626.

[2] S. Bilal and M. Abdelouahab, "Evolutionary algorithm and modularity for detecting communities in networks," *Phys. A, Stat. Mech. Appl.*, vol. 473, pp. 89–96, May 2017, doi: 10.1016/j.physa.2017.01.018.

[3] H. Papadakis, C. Panagiotakis, and P. Fragopoulou, "Distributed detection of communities in complex networks using synthetic coordinates," *J. Stat. Mechanics, Theory Exp.*, vol. 2014, no. 3, Mar. 2014, Art. no. P03013, doi: 10.1088/1742-5468/2014/03/P03013.

[4] Y. Zhang, Y. Xiong, Y. Ye, T. Liu, W. Wang, Y. Zhu, and P. S. Yu, "SEAL: Learning heuristics for community detection with generative adversarial networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 1103–1113.

[5] M. Bazzi, M. A. Porter, S. Williams, M. McDonald, D. J. Fenn, and S. D. Howison, "Community detection in temporal multilayer networks, with an application to correlation networks," *Multiscale Model. Simul.*, vol. 14, no. 1, pp. 1–41, Aug. 2016, doi: 10.1137/15M1009615.

[6] M. Seifikar, S. Farzi, and M. Barati, "C-blondel: An efficient louvain-based dynamic community detection algorithm," *IEEE Trans. Computat. Social Syst.*, vol. 7, no. 2, pp. 308–318, Apr. 2020, doi: 10.1109/TCSS.2020.2964197.

[7] K. Berahmand, A. Bouyer, and M. Vasighi, "Community detection in complex networks by detecting and expanding core nodes through extended local similarity of nodes," *IEEE Trans. Computat. Social Syst.*, vol. 5, no. 4, pp. 1021–1033, Dec. 2018, doi: 10.1109/TCSS.2018.2879494.

[8] K. Berahmand, E. Nasiri, R. Pir Mohammadiani, and Y. Li, "Spectral clustering on protein-protein interaction networks via constructing affinity matrix using attributed graph embedding," *Comput. Biol. Med.*, vol. 138, Nov. 2021, Art. no. 104933, doi: 10.1016/j.compbiomed.2021.104933.

[9] K. Berahmand, M. Mohammadi, and A. Faroughi, and R. P. Mohammadiani, "A novel method of spectral clustering in attributed networks by constructing parameter-free affinity matrix," *Cluster Comput.*, vol. 5, pp. 1021–1033. Nov. 2021, doi: 10.1007/s10586-021-03430-0.

[10] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proc. 12th ACM SIGKDD*, Philadelphia, PA, USA, 2006, pp. 554–560.

[11] F. Folino and C. Pizzuti, "An evolutionary multiobjective approach for community discovery in dynamic networks," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1838–1852, Aug. 2014, doi: 10.1109/TKDE.2013.131.

[12] X. Zeng, W. Wang, C. Chen, and G. G. Yen, "A consensus community-based particle swarm optimization for dynamic community detection," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2502–2513, Jun. 2020, doi: 10.1109/TCYB.2019.2938895.

[13] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng, ''Analyzing communities and their evolutions in dynamic social networks,'' *ACM Trans. Knowl. Discovery Data*, vol. 3, no. 2, pp. 1–31, Apr. 2009, doi: 10.1145/1514888.1514891.

[14] F. Folino and C. Pizzuti, ''A multiobjective and evolutionary clustering method for dynamic networks,'' in *Proc. Int. Conf. Adv. Social Netw. Anal. Mining*, Odense, Denmark, Aug. 2010, pp. 256–263.

[15] J. Ma, J. Liu, W. Ma, M. Gong, and L. Jiao, ''Decomposition-based multiobjective evolutionary algorithm for community detection in dynamic social networks,'' *Sci. World J.*, vol. 2014, pp. 1–22, Mar. 2014, doi: 10.1155/2014/402345.

[16] Z. Li, J. Liu, and K. Wu, ''A multiobjective evolutionary algorithm based on structural and attribute similarities for community detection in attributed networks,'' *IEEE Trans. Cybern.*, vol. 48, no. 7, pp. 1963–1976, Jul. 2018, doi: 10.1109/TCYB.2017.2720180.

[17] C. Gao, Z. P. Chen, X. H. Li, Z. H. Tian, and S. D. Li, ''Multiobjective discrete particle swarm optimization for community detection in dynamic networks,'' *Europhys. Lett.*, vol. 122, no. 2, pp. 1–6, Jun. 2018, doi: 10.1209/0295-5075/122/28001.

[18] C. Wang, Y. Deng, X. Li, J. Chen, and C. Gao, ''Dynamic community detection based on a label-based swarm intelligence,'' *IEEE Access*, vol. 7, pp. 161641–161653, 2019, doi: 10.1109/ACCESS.2019.2951527.

[19] F. Liu, J. Wu, C. Zhou, and J. Yang, ''Evolutionary community detection in dynamic social networks,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Budapest, Hungary, Jul. 2019, pp. 1–7.

[20] N. A. Kallioras, N. D. Lagaros, and D. N. Avtzis, ''Pity beetle algorithm—A new metaheuristic inspired by the behavior of bark beetles,'' *Adv. Eng. Softw.*, vol. 121, pp. 147–166, Jul. 2018, doi: 10.1016/j.advengsoft.2018.04.007.

[21] L. Danon, J. Duch, A. D. Guilera, and A. Arenas, ''Comparing community stuctrue identification,'' *J. Stat. Mech., Theory Exp.*, vol. 2005, pp. 1–10, Sep. 2005, doi: 10.1088/1742-5468/2005/09/P09008.

[22] C. Pizzuti, ''GA-Net: A genetic algorithm for community detection in social networks,'' in *Proc. PPSN*, Berlin, Germany, 2008, pp: 1081-1090.

[23] Z. Li, S. Zhang, R.-S. Wang, X.-S. Zhang, and L. Chen, ''Quantitative function for community detection,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 77, no. 3, Mar. 2008, Art. no. 036109, doi: 10.1103/PhysRevE.77.036109.

[24] M. E. J. Newman and M. Girvan, ''Finding and evaluating community structure in networks,'' *Phys. Rev. E*, vol. 69, Feb. 2004, Art. no. 026113, doi: 10.1103/PhysRevE.69.026113.

[25] Z. Li and J. Liu, ''A multi-agent genetic algorithm for community detection in complex networks,'' *Phys. A, Stat. Mech. Appl.*, vol. 449, pp. 336–347, May 2016, doi: 10.1016/j.physa.2015.12.126.

[26] M. E. J. Newman, ''Finding community structure in networks using the eigenvectors of matrices,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 74, no. 3, Sep. 2006, Art. no. 036104, doi: 10.1103/PhysRevE.74.036104.

[27] M. E. J. Newman, ''Fast algorithm for detecting community structure in networks,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 69, no. 6, Jun. 2004, Art. no. 066133, doi: 10.1103/PhysRevE.69.066133.

[28] M. Girvan and M. E. J. Newman, ''Community structure in social and biological networks,'' *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002, doi: 10.1073/pnas.122653799.

[29] M. G. Gong, B. Fu, L. C. Jiao, and H. F. Du, ''Memetic algorithm for community detection in networks,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 84, Nov. 2011, Art. no. 056101, doi: 10.1103/PhysRevE.84.056101.

[30] P. Pons and M. Latapy, ''Computing communities in large networks using random walks,'' in *Proc. ISCIS*, Berlin, Germany, 2005, pp: 284-293.

[31] W. W. Zachary, ''An information flow model for conflict and fission in small groups,'' *J. Anthropol. Res.*, vol. 33, no. 4, pp. 452–473, Dec. 1977, doi: 10.1086/jar.33.4.3629752.

[32] A. Clauset, M. E. J. Newman, and C. Moore, ''Finding community structure in very large networks,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 70, no. 6, Dec. 2004, Art. no. 066111, doi: 10.1103/physreve.70.066111.

[33] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, ''Fast unfolding of communities in large networks,'' *J. Stat. Mech., Theory Exp.*, vol. 2008, no. 10, pp. 155–168, 2008, doi: 10.1088/1742-5468/2008/10/p10008.

[34] T. C. Havens, J. C. Bezdek, C. Leckie, K. Ramamohanarao, and M. Palaniswami, ''A soft modularity function for detecting fuzzy communities in social networks,'' *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 6, pp. 1170–1175, Dec. 2013, doi: 10.1109/tfuzz.2013.2245135.

[35] J. Su and T. C. Havens, ''Quadratic program-based modularity maximization for fuzzy community detection in social networks,'' *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 5, pp. 1356–1371, Oct. 2015, doi: 10.1109/tfuzz.2014.2360723.

[36] M. Rosvall and C. T. Bergstrom, ''Maps of random walks on complex networks reveal community structure,'' *Proc. Nat. Acad. Sci. USA*, vol. 105, no. 4, pp. 1118–1123, Oct. 2008, doi: 10.1073/pnas.0706851105.

[37] B. Laassem, A. Idarrou, and L. Boujlaleb, ''Label propagetion algorithm for community detection based on Coulomb's law,'' *Physica A, Stat. Mech. Appl.*, vol. 593, May 2022, Art. no. 126881, doi: 10.1016/j.physa.2022.126881.

[38] X. G. Q., J. W. Guo, and P. L. Yang, ''TNS-LPA: An improved labelpropagation algorithm for community detection based on two-level neighbourhood similarity,'' *IEEE Access*, vol. 9, pp. 161641–161653, 2020, doi: 10.1109/ACCESS.2020.3045085.

[39] M. Gong, B. Fu, L. Jiao, and H. Du, ''Memetic algorithm for community detection in networks,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 84, no. 5, Nov. 2011, Art. no. 056101, doi: 10.1103/physreve.84.056101.

[40] C. Pizzuti, ''A multiobjective genetic algorithm to find communities in complex networks,'' *IEEE Trans. Evol. Comput.*, vol. 16, no. 3, pp. 418–430, Jun. 2012, doi: 10.1109/TEVC.2011.2161090.

[41] J. Li and Y. Song, ''Community detection in complex networks using extended compact genetic algorithm,'' *Soft Comput.*, vol. 17, no. 6, pp. 925–937, 2013, doi: 10.1007/s00500-012-0942-1.

[42] A. Said, R. A. Abbasi, O. Maqbool, A. Daud, and N. R. Aljohani, ''CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks,'' *Appl. Soft Comput.*, vol. 63, pp. 59–70, Feb. 2018, doi: 10.1016/j.asoc.2017.11.014.

[43] M. Gong, L. Ma, Q. Zhang, and L. Jiao, ''Community detection in networks by using multiobjective evolutionary algorithm with decomposition,'' *Phys. A, Stat. Mech. Appl.*, vol. 391, no. 15, pp. 4050–4060, 2012, doi: 10.1016/j.physa.2012.03.021.

[44] A. Song, M. Li, X. Ding, W. Cao, and K. Pu, ''Community detection using discrete bat algorithm,'' *IAENG Int. J. Comput. Sci.*, vol. 43, no. 1, pp. 37–43, 2016.

[45] C. Pizzuti, ''Evolutionary computation for community detection in networks: A review,'' *IEEE Trans. Evol. Comput.*, vol. 22, no. 3, pp. 464–483, Jun. 2018, doi: 10.1109/tevc.2017.2737600.

[46] C. Zhang, X. Hei, D. Yang, and L. Wang, ''A memetic particle swarm optimization algorithm for community detection in complex networks,'' *Int. J. Pattern Recognit. Artif. Intell.*, vol. 30, no. 2, Feb. 2016, Art. no. 1659003, doi: 10.1142/S0218001416590035.

[47] M. Gong, Q. Cai, X. Chen, and L. Ma, ''Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition,'' *IEEE Trans. Evol. Comput.*, vol. 18, no. 1, pp. 82–97, Feb. 2014, doi: 10.1109/TEVC.2013.2260862.

[48] G. Jia, Z. Cai, M. Musolesi, Y. Wang, D. A. Tennant, R. J. M. Weber, J. K. Heath, and S. He, ''Community detection in social and biological networks using differential evolution,'' in *Learning and Intelligent Optimization*. Berlin, Germany: Springer, 2012.

[49] Q. Huang, T. White, G. Jia, M. Musolesi, N. Turan, K. Tang, S. He, J. K. Heath, and X. Yao, ''Community detection using cooperative co-evolutionary differential evolution,'' in *Parallel Problem Solving from Nature*. Berlin, Germany: Springer, 2012.

[50] Y.-J. Zhang, Z.-H. Gong, and Q.-K. Chen, ''Community detection in complex networks using immune discrete differential evolution algorithm,'' *Acta Automatica Sinica*, vol. 41, no. 4, pp. 749–757, 2015, doi: 10.16383/j.aas.2015.c140018.

[51] Q. Liu, B. Zhou, S. Li, A.-P. Li, P. Zou, and Y. Jia, ''Community detection utilizing a novel multi-swarm fruit fly optimization algorithm with hill-climbing strategy,'' *Arabian J. Sci. Eng.*, vol. 41, no. 3, pp. 807–828, 2016, doi: 10.1007/s13369-015-1905-5.

[52] J. Xiao, Y.-J. Zhang, and X.-K. Xu, ''Convergence improvement of differential evolution for community detection in complex networks,'' *Phys. A, Stat. Mech. Appl.*, vol. 503, pp. 762–779, Aug. 2018, doi: 10.1016/j.physa.2018.02.072.

[53] X. Zhou, K. Yang, Y. Xie, C. Yang, and T. Huang, "A novel modularity-based discrete state transition algorithm for community detection in networks," *Neurocomputing*, vol. 334, pp. 89–99, Mar. 2019, doi: 10.1016/j.neucom.2019.01.009.

[54] J. Xiao, C. Wang, and X. K. Xu, "Community detection based on symbiotic organisms search and neighborhood information," *IEEE Trans. Comput. Soc. Syst.*, vol. 6, no. 6, pp. 1257–1272, Oct. 2019, doi: 10.1109/TCSS.2019.2941988.

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

[56] H. Y. Zhang, X. W. Chen, J. Li, and B. Zhou, "Fuzzy community detection via modularity guided membership-degree propagation," *Pattern Recognit. Letters.*, vol. 70, pp. 66–72, Dec. 2015, doi: 10.1016/j.patrec.2015.11.008.

[57] Z. Y. Liu, Y. H. Ma, and X. J. Wang, "A compression-based multi-objective evolutionary algorithm for community detection in social networks," *IEEE Access*, vol. 8, pp. 62137–62150, Mar. 2020, doi: 10.1109/ACCESS.2020.2984638.

[58] X. Zhao, J. Liang, and J. Wang, "A community detection algorithm based on graph compression for large-scale social networks," *Inf. Sci.*, vol. 551, pp. 358–372, Apr. 2021, doi: 10.1016/j.ins.2020.10.057.

[59] 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, no. 3, Sep. 2007, doi: 10.1103/PhysRevE.76.036106.

[60] H. S. Pattanayak, A. L. Sangal, and H. K. Verma, "Community detection in social networks based on fire propagation," *Swarm Evol. Comput.*, vol. 44, pp. 31–48, Feb. 2019, doi: 10.1016/j.swevo.2018.11.006.

YAN-JIAO WANG was born in Heilongjiang, China, in 1985. She received the Ph.D. degree in signal and information process from Harbin Engineering University, in 2013. She is currently an Associate Professor of information and communication engineering with the School of Electrical Engineering, Northeast Electric Power University. Her current research interests include evolutionary computing and intelligent information processing.

JIA-XU SONG was born in Shandong, China, in 1996. She received the B.S. degree in communication engineering from Jiangxi Normal University, in 2019. She is currently pursuing the master's degree in information and communication engineering with Northeast Electric Power University. Her research interests include evolutionary computing and intelligent information processing.

PENG SUN was born in Jilin, China, in 1997. He received the master's degree in communication engineering from Northeast Electric Power University, in 2020. He is currently working at China Unicom. His research interests include evolutionary computing and network analysis.