Received June 17, 2020, accepted June 26, 2020, date of publication June 30, 2020, date of current version July 20, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3006108

A Memetic Algorithm for Community Detection in Signed Networks

SHIWEI CHE, WU YANG, AND WEI WANG
Information Security Research Center, Harbin Engineering University, Harbin 150001, China

Corresponding author: Wu Yang (yangwu@hrbeu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61572459 and Grant 61672180.

ABSTRACT Community discovery (i.e. community detection) in signed networks is a division of nodes, such that the edges in the communities are positive and the edges between the communities are negative. Davis and Harary have solved the problem of community detection when a signed graph is balanced or weakly balanced. When the signed network is unbalanced, community detection becomes very complex. In this paper, we propose a novel memetic algorithm (MA) called MACD-SN for community partition (i.e. community detection) in signed networks. Firstly, we present a novel initialization algorithm used in initialization of MACD-SN. This method can accelerate the convergence rate of MACD-SN algorithm. Next, in addition to using frequently-used variation operation (in this paper, variation and mutation are interchangeable), this paper presents a novel crossover operation and a novel variation operation, which contributes to increasing the correctness of the MACD-SN algorithm's operation result and reduces its running time. Lastly, this paper proposes a new local search algorithm, which may enable the algorithm's result to jump away the local best result with a certain probability and draw near the global best result quickly. For testing the performance of MACD-SN algorithm, we have done many experiments using five kinds of synthetic signed networks and five real-world signed networks. The test outcomes show that the proposed algorithm is valid and efficient for signed network cluster partition (i.e. community detection).

INDEX TERMS Genetic algorithm, social network, signed network, community detection.

I. INTRODUCTION

Modern network science is an active field in understanding complex systems. Actually, a lot of complicated systems in various fields can be expressed by means of networks, for example, complex collaborative relationships [1], social systems [2], information systems [3], etc. In these networks, nodes (or vertices) represent individual participants, and edges (or links) represent relationships between participants. A great deal of research efforts have been done on complex networks, such as correlation clustering, dynamic network evolution. Generally speaking, identifying community partition is an important task in complex network analysis. Community structures exist in a lot of network systems, such as politics, economics, engineering, computer science, biology and so on. A comment on network community discovery can be found in Ref. [4]. The purpose of community discovery is to identify clusters with dense links within clusters and only sparse links between clusters [5]. In theoretical research and practical activities, community discovery is of great significance. For instance, in the purchasing relations network between customers and online retailers' products (such as www.taobao.com), identifying clusters of customers with similar interests can establish an effective recommendation system [4], [6].

In human society, many relationships between people are signed, either positive or negative. Compared with traditional networks, the positive and negative edges of signed networks can more accurately describe cooperation (friendship/trust) relations and competition (hostility/distrust) relations. When two people have a relationship of trust, respect or love, the relationship can be regarded as a positive connection. But, the relationship with mistrust, disrespect or hatred can be considered as a negative connection. This network is called signed network [7], that is, the edge weight is greater than 0, indicating a positive relationship; the edge weight is less than 0, indicating a negative relationship; and the edge weight is equal to 0, indicating that there is no relationship between these two individuals. Figure 1 shows a simple signed network. In the figure, the solid line edge

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

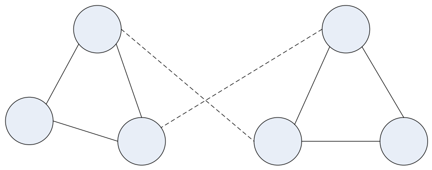


FIGURE 1. A schematic diagram of a simple signed graph.

represents a positive relationship, and the dotted line edge represents a negative relationship. Community discovery in signed networks is quite different from that in unsigned networks (that is, networks only contain positive connections). In unsigned networks, the community structure is defined as a group of nodes or vertices which have dense connections within groups and sparse connections between groups. Whereas for signed social networks, communities (i.e. clusters) are defined not only by the density of connections but also by the signs of connections. The connections should be densely positive and sparsely negative in a community while densely negative and sparsely positive between communities. Many negative connections exist in the communities and many positive connections exist between the communities make community detection more difficult. A strongly (or weakly) balanced signed network can be divided into two (or more) clusters, so that all connections within clusters are positive, and all connections between clusters are negative [8]–[10]. However, due to the existence of negative connections in clusters and positive connections between clusters, the real world signed networks are often unbalanced. Therefore, it is a great challenge to design an effective and efficient algorithm to discover the community structure in signed graphs.

Unlike previous work, this paper proposes a new memetic algorithm to detect community structure in signed networks. In order to accelerate the convergence of algorithm (decrease the numbers of loop), a novel population initialization method of memetic algorithm is presented. Besides employing the frequently-used variation operation, a novel crossover operation (called randomized two-way crossover operation in this article) and a novel variation operation (called community variation operation in this article) are also proposed, which are capable of enhancing the accuracy of the result of the algorithm and speeding up the convergence speed of population. Randomized two-way crossover operation can preferable retain the hereditary properties of previous generation individuals, and community variation operation is capable of enhancing greatly the chromosomes set (i.e. chromosomes population) multiformity of MACD-SN algorithm. Furthermore, this paper also presents a local solution space search subroutine to drive the optimal result of the offspring individuals of the MACD-SN method approach the global optimal result more quickly in the search region. This subroutine is capable of driving the MACD-SN jump away local best solution and attain global best solution with a specified odds. Many experimental results show that the proposed algorithm is effective and efficient for signed network community partition.

The remaining sections of this thesis are arranged as follows: The work related to this study is illuminated in Section II. A few key concepts and background knowledge connected with this study are introduced in Section III. Section IV describes presented MACD-SN algorithm for community identification (i.e. community detection) in the signed network in detail. This section introduces the proposed MACD-SN algorithm's chromosome coding method, initialization algorithm of chromosomes set, computational formula of fitness used, tournament selection operator for chromosomes selection, crossover operator and mutation operators of genetic operation, local search function, etc. Section V shows the test results on synthetic and real signed networks. Section VI summarizes the whole paper.

II. RELATED WORKS

In recent decades, due to the emergence of a large number of community partition problems, scholars have proposed many algorithms to settle the community partition problems. Girvan and Newman presented a dividing method, which is called GN algorithm. In addition, Newman also put forward a method called FN based on GN algorithm, which uses modularity function. It is a kind of agglomeration algorithm [11]. In the FN algorithm, each node in the graph is initially located in a community with only one node. Afterwards, at every stage, the method continuously consolidates cluster pair with the largest modularity function increment. According to majorization of modularity function, Moore *et al.* [12] proposed a method named CNM to detect community structures in complex networks. In comparison to the FN method, CNM method can save computing time and is appropriate for discovering cluster partition in large scale graphs. In [13], Newman put forward a spectral method as well. The algorithm used a modularity matrix.

In recent years, researchers have proposed many multi-objective majorization algorithms to solve community discovery problems. The multi-objective majorization algorithm finds the optimal solution of the task to be solved by majorizing multiple majorization functions at the same time. These majorization functions evaluate the discovered cluster structure from multiple viewpoints. In the literatures, several frequently cited multi-objective majorization algorithms for solving community discovery problems are listed as follows. Shi *et al.* [14] put forward a multi objective evolutionary algorithm called MOCD. In [15], the authors presented an algorithm named MOEA/D-Net, which is also a multi-objective evolutionary algorithm (MOEA). Liu *et al.* [65] proposed a COMpression based Multi-Objective Evolutionary Algorithm with Decomposition (Com-MOEA/D) for community detection. In the prevenient literatures, there are also two other multi-objective evolutionary algorithms, which are called MODTLBO/D [16] and MODBSA/D [17], respectively. The author of [18] developed a multi objective genetic algorithm called MOGA-Net.

Although there are many excellent algorithms that can be used to detect communities in the unsigned network, these algorithms can not be directly applied to the signed network because of the existence of edge signs in the signed network. Hence, a large number of scholars have proposed a lot of community detection algorithms for signed networks after considering the edge signs of signed networks.

Albayrak *et al.* [19] apply the spectral algorithm to the signed network, and present a spectral method on the basis of the signed Laplacian. They concluded that using signed Laplacian kernel to divide signed networks into two clusters is similar to the ratio cut in unsigned networks. Based on the spectral Laplacian, Tewari *et al.* [20] propose a community detection algorithm for signed networks, and defines the social imbalance (MOIs) on the basis of the l-cycles in signed networks. However, the spectral method is very time consuming.

Arenas *et al.* [21] modified the modularity definition of the unsigned network and extended it to the signed network. Bruggeman *et al.* [22] modified an existing Potts model to include negative links, resulting in a method similar to signed graphs clustering (i.e. community detection). The author of [23] proposed a statistical probability model to identify the community partition of signed networks. Moura *et al.* [24] put forward a mixed integer programming model for clustering problems related to structural balance. Ismail *et al.* [25] presented a high-efficiency two stage algorithm to identify the community structures in signed social networks. The objective functions they used are to minimize frustration and maximize modularity. Dhillon *et al.* [26] presented a scalable and efficient clustering algorithm using balance normalized cut and a multilevel clustering algorithm. Pizzuti *et al.* [27] obtained the community structure of the signed network by maximizing the cluster modularity and minimizing the number of negative edges within communities and the number of positive edges between communities. Liu *et al.* [28] presented two novel evolutionary algorithms and conducted a large number of experiments to compare them. The experimental results show the effectiveness and efficiency of the two algorithms.

Cheung *et al.* [29] presented a random walk algorithm for community detection of signed networks. Firstly, select a node (i.e. vertex) in the network that has not yet assigned a community label, and then use the node as the starting node to perform a specific number of random walks to determine the set of nodes it can reach along the edge path. Next, the authors propose a function to determine which vertices are in the same community as the starting vertex. At the same time, the function considers the distribution of positive connections and negative connections between the community and the rest of the network. From [29], it can see that the result matrix has block characteristics. By using the above function to segment the matrix, the matrix will be split into different block matrices representing different communities.

Du *et al.* [30] proposed a multi objective discrete particle swarm optimization algorithm for multi-resolution signed network clustering. Firstly, the algorithm generates initial population information including location, speed, individual optimal solution and neighborhood information. Next, the new velocity and position of each particle are calculated, and a small disturbance is added to the new position; then, the disturbed position is evaluated, and the neighborhood information and optimal solution of the particle are updated according to the evaluation result.

Jiang *et al.* [31] propose a multi objective evolutionary algorithm, which is based on similarity. In order to consider the sign characteristic of signed networks, the authors proposed two new indexes. One index is the signed similarity index according to the existing similarity index [32]. Another index is the signed tightness index based on the existing tightness index [32]. During the run of the algorithm, when the movement can increase the signed tightness of the cluster (i.e., community), the node will be moved to another cluster. However, if the movement can not increase the signed tightness of the cluster, the node will be independent as a new cluster. A special case is that multiple clusters have the same vertices during the run of the algorithm, so when two clusters have more than half of the same vertices, they will be merged into a new cluster. The author of [33] used the Signed Stochastic Block-Model in the process of community detection. Haseyama *et al.* [34] partition network video by constructing a weighted signed network and maximizing the local modularity.

Recently, some scholars have done some very good works for the community detection of signed networks. For example, Attea *et al.* [44] proposed a new multi-objective signed community detection model and a new anti-frustration heuristic operator for the community detection of signed networks. Ma *et al.* [64] used the relationship between balancedness and spectrum space, and proposed a spectrum algorithm based on leading eigenvectors of signed networks to partition clusters, so as to maximize the balancedness. Zhu *et al.* [45] proposed a new evolution algorithm for community detection in imbalanced signed networks which can be modeled as an optimal partition problem. And the evolving mechanism of nodes is updated by its neighbors' information which leads to form optimal community structure. Yan *et al.* [48] proposed a new modularized convex nonnegative matrix factorization (NMF) model, which combined signed modularized information with convex NMF model to improve the accuracy of community detection in signed and unsigned networks. As for model selection, Yan *et al.* extended the modularity density to signed networks and employed the signed modularity density to determine the number of communities automatically.

Most of the algorithms described in this section are based on global information to identify communities. So far, some scholars have begun to use local information for community detection. The knowledge of using local information for community detection is beyond the scope of this paper, which is not discussed here.

III. FUNDAMENTAL NOTIONS AND BACKGROUND KNOWLEDGE

A. THE DEFINITION OF SIGNED NETWORK AND THE DEFINITION OF CLUSTER DETECTION OF SIGNED NETWORK

A signed social network can be modeled as a graph $G = (V,E)$, where $V = (v_1, v_2, \ldots, v_n)$ is the set of nodes (or vertices), $E = \{(v_i, v_j) \mid v_i, v_j \in \text{V} \cap i \neq j\}$ is the set of edges (or links). We can represent graph G by an adjacency matrix $A = (A_{ij})_{n\times n}$, where $A_{ij} = 1(\text{or } -1)$ if we observe it is the positive (negative) relationship between v_i and v_j, and $A_{ij} = 0$ means that there is no edge between v_i and v_j. If $A_{ij} = 1$, nodes v_i and v_j are positive neighbors of each other; if $A_{ij} = -1$, nodes v_i and v_j are negative neighbors of each other. Given a node $v_i \in V$, $a_i^+ = \sum_{(v_i,v_j)\in E \bigwedge A_{i,j}=1} A_{ij}$ and $a_i^- = \sum_{(v_i,v_j)\in E\wedge A_{i,j}=-1} |A_{ij}|$ are defined respectively as the positive degree and the negative degree of v_i. $a^+ = \sum_{v_i\in V} a_i^+$ and $a^- = \sum_{v_i\in V} a_i^-$ are the total positive degree and the total negative degree of the signed network, respectively. If for any i and j, $A_{ij} >= 0$, then G is an unsigned network. Here, we do not consider the direction of the edge between any two nodes, that is, G is an undirected graph in this paper.

Let $C = \{c_1, c_2, \ldots, c_k\}$ be a set of communities in G, that is, $c_i \subset V$ for $i = 1, 2, \ldots, k$. The problem of community detection in signed networks is accurately expressed in (1).

$$\begin{cases} A_{ij} > 0, & (v_i, v_j) \in E \wedge v_i \in c_p \wedge v_j \in c_p \\ A_{ij} < 0, & (v_i, v_j) \in E \wedge v_i \in c_p \wedge v_j \in c_q \end{cases} \tag{1}$$

where $p \neq q, p, q = 1, 2, \ldots, k$. The problem can be described as identifying the community partition that maximizes the sum of positive edges within communities and negative edges between communities.

If all the positive edges in the signed network are in the communities, and all the negative edges are between the communities, the signed network is balanced; otherwise, the signed network is unbalanced.

B. A SIGNED MODULARITY FUNCTION OF SIGNED GRAPHS

In this section, a signed modularity function Q_s [21] of the signed networks will be described. We will use it later in this study. Its expression is as follows:

$$Q_s = \frac{1}{2a^+ + 2a^-} \sum_i \sum_j \left[A_{ij} - \left(\frac{a_i^+ a_j^+}{2a^+} - \frac{a_i^- a_j^-}{2a^-}\right)\right] \delta\left(c_i, c_j\right) \tag{2}$$

where A_{ij} is the element of adjacency matrix A, and c_i, c_j denote the communities to which nodes v_i and v_j belong, respectively. If $c_i = c_j$, $\delta\left(c_i, c_j\right) = 1$, otherwise, $\delta\left(c_i, c_j\right) = 0$. To understand the meaning of $a_i^+(a_i^-)$, $a_j^+(a_j^-)$ and $a^+\left(a^-\right)$, please refer to section III-A.

IV. PROPOSED ALGORITHM

Here, we put forward a memetic algorithm use in community discovery in signed networks, called MACD-SN. At first, we describe the representation of chromosomes in a population. Then, a novel population initialization algorithm is presented, which can significantly speed up the convergence rate of the MACD-SN algorithm. Next, a computational formula for assessing chromosomes within a population is introduced. At the same time, a selection operator for selecting parent chromosomes for subsequent genetic operations is described. Afterwards, we elaborate on a crossover operation (called randomized two-way crossover operation) and two mutation operations (called traditional mutation operation and community mutation operation respectively), which are used in MACD-SN algorithm. The novel crossover operation and novel community mutation operation proposed by us can significantly improve the accuracy of MACD-SN algorithm's result and reduce the running time of the algorithm. In the end, we present a local solution space search subroutine. This subroutine can make the result of MACD-SN algorithm jump out of the local optimal result with a certain probability and approach the global optimal result quickly. The rest of Section IV will elaborate on each theme. Figure 2 shows the flow diagram of MACD-SN algorithm.

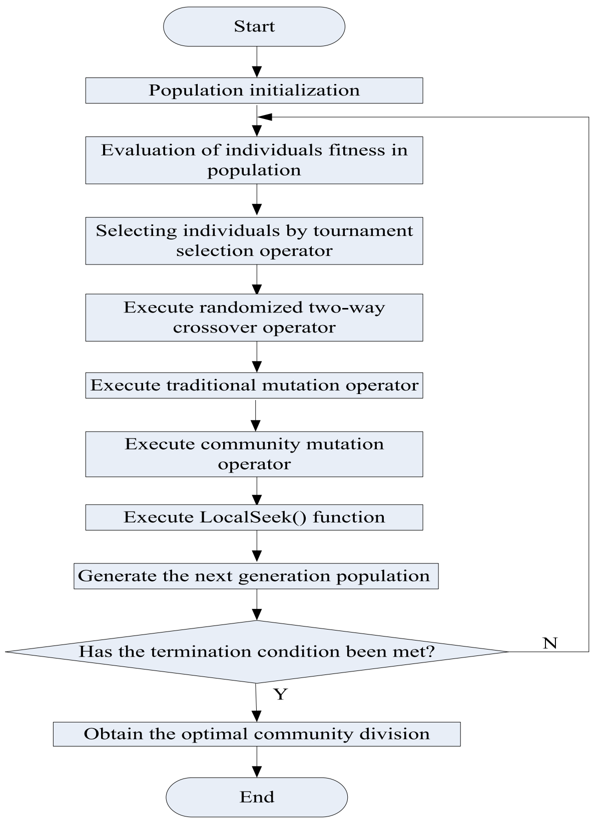

Start
Population initialization
Evaluation of individuals fitness in population
Selecting individuals by tournament selection operator
Execute randomized two-way crossover operator
Execute traditional mutation operator
Execute community mutation operator
Execute LocalSeek() function
Generate the next generation population
Has the termination condition been met?
N
Y
Obtain the optimal community division
End

FIGURE 2. The flow diagram of MACD-SN Algorithm.

A. CHROMOSOME REPRESENTATION

In the classical memetic algorithm, each candidate community division of network is denoted by a chromosome, also

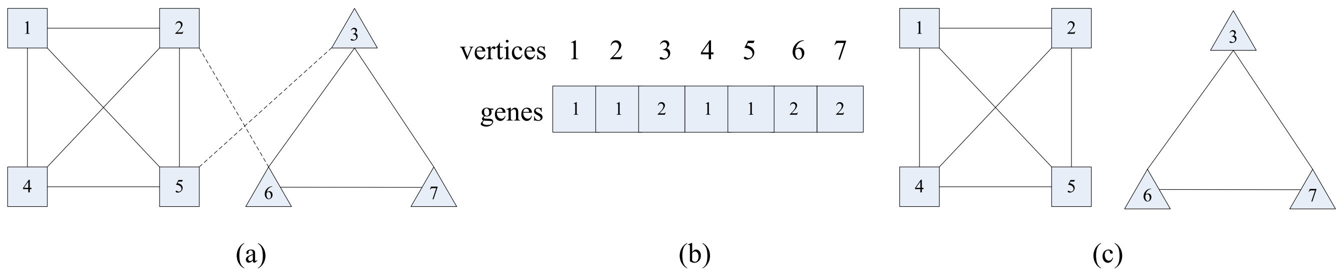


FIGURE 3. (a)A signed network contains 7 vertices, the solid line edge represents a positive relationship, and the dotted line edge represents a negative relationship; (b)string-based coding method of a individual; (c)schematic diagram of the community structure of the network in (a) decoded by the individual in (b).

called a solution or an individual. A set of a certain number of chromosomes is known as a population, i.e., population $popu = \{ch_1, ch_2, \ldots, ch_q\}$, where ch_j is the jth chromosome within the chromosomes set and q is the number of chromosomes in the set. Chromosome coding methods frequently used in the literatures consist of locus-based coding method and string-based coding method. In order to obtain the corresponding community partition by decoding chromosome conveniently, we make use of string-based coding method. The jth chromosome in the population of memetic algorithm can be represented as: $ch_j = [ge_1, ge_2, \ldots, ge_n]$. Among them, ge_k is the kth gene (or component) of individual ch_j, and the amount of vertices in the signed network is n. The value range of each gene is $\{1, 2, \ldots, n\}$. Genes denote the vertices in the signed network, and the value of the kth gene denote the community to which vertex v_k belongs. In this coding method, if vertices v_i and v_j are in the same community, $ge_i = ge_j$.

The chromosome shown in Figure 3(b) is a string-based coding method of the signed network illustrated in Figure 3(a). This signed graph consists of seven vertices. The number of vertices in the network is 1, 2, 3, 4, 5, 6 and 7 respectively. In Figure 3(b), we can see that the gene values assigned to vertices 1, 2, 4 and 5 are all 1, and the gene values assigned to vertices 3, 6 and 7 are all 2. This shows that the network contains two clusters, in which vertices 1, 2, 4 and 5 are in the same cluster, while vertices 3, 6 and 7 are in the other cluster. Figure 3(c) illustrates the cluster partition decoded by the chromosome in Figure 3(b).

B. CREATION OF INITIAL CHROMOSOMES POPULATION

In order to get an excellent memetic algorithm, it is very important to generate a good initial population. The reason is that the properties of the initial chromosomes will have an effect on the convergence rate and the quality of the final result of the method. Therefore, we propose an efficient population initialization algorithm to generate a good initial population and to reduce the convergence time of the whole algorithm. Before introducing the initialization algorithm, we first introduced a definition to be used in the algorithm.

Definition 1 [Node Imbalance Degree (NID)]: Let v_i be a node in the signed network G, and c be a community in G, then the node imbalance degree $NID(v_i, c)$ of node v_i relative to community c is calculated as follows:

$$NID(v_i, c) = \sum\nolimits_{v_j \in c \wedge (v_i, v_j) \in E \wedge A_{ij} = -1} |A_{ij}| + \sum\nolimits_{v_j \notin c \wedge (v_i, v_j) \in E \wedge A_{ij} = 1} A_{ij} \quad (3)$$

Let c_1 and c_2 be two communities in G. Obviously, according to the definition of community structure of signed network, when $NID(v_i, c_1) < NID(v_i, c_2)$, the priority should be given to assigning node v_i to community c_1. Therefore, in the initialization process of the MACD-SN algorithm proposed by us, node k (equivalent to v_i above) is assigned to the neighbor community which reduces its node imbalance degree (NID) to the minimum. Algorithm 1 describes the pseudocode of the initialization process of the presented MACD-SN algorithm.

In the above algorithm, the for loop at lines 1 to 24 is used in producing the initial chromosomes set. The codes in lines 2 to 4 generate an initial chromosome, each gene value of which is a random integer randomly generated in the range of 1 to n. The while loop of lines 6 to 23 continuously optimizes the generated chromosomes, until in a while loop, each node cannot optimize its community label according to its neighbor's community label. The for loop of lines 8 to 22 optimizes chromosome by a single pass loop from 1 to n. In line 9, the community labels of the neighbor nodes of the current node k are stored in the set *comms*. The codes in lines 10 to 17 is responsible for finding the community in *comms* that minimizes the node imbalance degree (NID) of node k, and storing it in m. The codes in lines 18 to 21 check whether the current community label of vertex k is m, if not, use the m value as the community label of vertex k, and set the value of the variable *flag* to 1 to continue the while loop. In the end, the codes in line 25 return the generated population of chromosomes.

Each time the for loop of line 1 is executed, it needs to iterate the *popu_size* times. Each time the for loop of line 2 is executed, it needs to iterate n (n is the number of nodes in the signed network) times. A large number of experiments show that every time the while loop of line 6 is executed, it needs to iterate 8 times on average, no more than 13 times at most. Each time the for loop of line 8 is executed, it needs

Algorithm 1 Pseudocode of Population Creation Function Initialize() of MACD-SN Method

```
Algorithm Parameters: chromosomes set size $popu\_size$;
Algorithm Input: A matrix $G$ representing a signed graph;
Algorithm Output: The initial set of chromosomes
generated;
1: for $j = 1$ to $popu\_size$ do
2:    for $k = 1$ to $n$ do
3:        $popu[j][k]$ = A random integer in the range of 1
          to $n$ generated randomly; //$popu[]$ is an array
          //of chromosomes population. $popu[j][k]$
          //represents the $k$th gene of the $j$th
          //chromosome.
4:    end for
5:    $flag = 1$;
6:    while $flag$ do
7:        $flag = 0$;
8:        for $k = 1$ to $n$ do
9:            $comms$ = The set of communities to which
                 the neighbor nodes of node $k$ belong;
10:           $t = \infty$;
11:           $m = popu[j][k]$;
12:           for each community $comm \in comms$ do
13:               if $NID(k, comm) < t$ then
14:                   $t = NID(k, comm)$;
15:                   $m = comm$;
16:               end if
17:           end for
18:           if $m \neq popu[j][k]$ then
19:               $popu[j][k] = m$;
20:               $flag = 1$;
21:           end if
22:       end for
23:   end while
24: end for
25: return $popu[]$;
```

to iterate n times. It is assumed that the average degree of nodes in the network is d ($d \ll n$). The time complexity of the statement in line 9 is $O(d)$. In the worst case, the for loop of line 12 needs to iterate n-1 times for each execution. Therefore, the time complexity of the initialization process of MACD-SN algorithm is $O(popu\_size \times n^2)$.

C. FITNESS FUNCTION

In the MACD-SN algorithm proposed in this paper, we use the signed modularity formula Q_s (Eq. 2) introduced in section III-B as the fitness function. According to [21], the larger the value of Q_s, the better the community partition obtained.

D. SELECTION OPERATOR

For selecting parent individuals for subsequent genetic operations, we should propose a good selection operator. In recent years, many methods have been developed as the selection algorithm of memetic algorithm (MA). Tournament selection algorithm is one of them. In order to make the low fitness individuals appear in the offspring chromosomes set, we need a method to control elitism. The tournament selection algorithm meets this requirement, so we chose it. One of the most attractive features of this algorithm is that chromosomes in the current population have the same probability of becoming the parent chromosomes of subsequent genetic operations. The flow chart of the algorithm is shown in Figure 4.

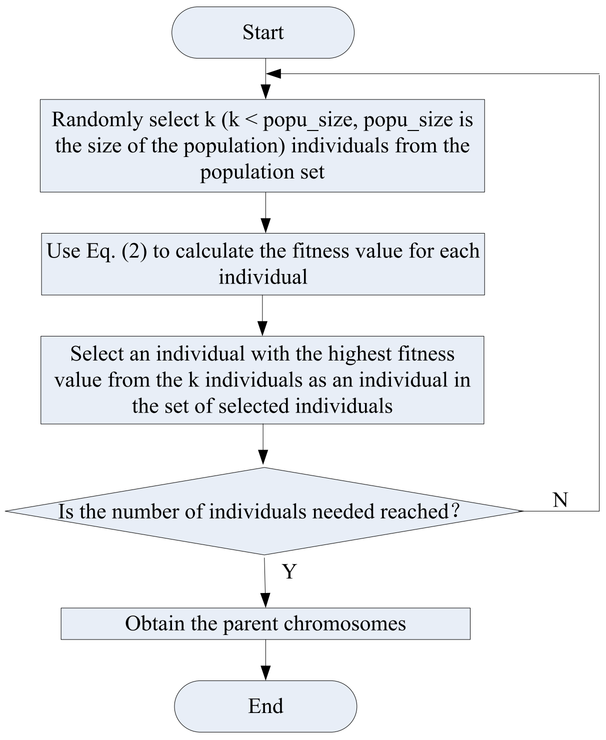

Start
Randomly select k (k < popu_size, popu_size is the size of the population) individuals from the population set
Use Eq. (2) to calculate the fitness value for each individual
Select an individual with the highest fitness value from the k individuals as an individual in the set of selected individuals
Is the number of individuals needed reached?
N
Y
Obtain the parent chromosomes
End

FIGURE 4. The flow chart of tournament selection algorithm.

The time complexity of tournament selection algorithm is $O(k^2)$ (k is the number of parent chromosomes to be selected by the tournament selection algorithm.).

E. CROSSOVER OPERATOR

Crossover operator is also one of the genetic operators of MA algorithm. The crossover operator applies simultaneously to two parent chromosomes chosen by the selection operator, and reproduces new chromosomes through interchanging contents between the selected chromosomes. Hence, the chromosomes produced through crossover operator possess simultaneously the genetic properties of two chromosomes of the previous generation [35]. The execution process of crossover operator is: (1) choosing two chromosomes from the parent chromosomes set; (2) exchanging contents between them in the light of crossover operation rules to generate novel offspring chromosomes; (3) giving two novel chromosomes. First, randomly select a crossing location. Afterwards, according to the specified crossover probability, the corresponding contents of the two parent chromosomes around the location are interchanged. Traditional crossover operations are listed below: the uniform crossover, the one-point crossover, the one-way crossover, the two-point crossover, the two-way crossover, and so on.

For taking full advantage of the community partition information of two individuals of the previous generation, enhance the accuracy of the offsprings produced through crossover operation and reduce the running time of algorithm, this paper presents a novel crossover operation, which is named randomized two-way crossover operation. The pseudo code of randomized two-way crossover operation is shown in algorithm 2.

Figure 5 shows an executive process of the crossover operator of the MACD-SN method. Figure 6 shows an example of community partition encoded by chromosomes ch_1, ch_2, d_1 and d_2 in Figure 5, respectively.

In Figure 5, ch_1 and ch_2 are two parent individuals chosen by the tournament selection algorithm from the previous generation population. The steps of obtaining a descendant individual d_1 by the individuals ch_1 and ch_2 of the previous generation are listed below. Step one: Label all components of ch_1 and ch_2 as untouched, assign 0 to all components of the descendant individual d_1. Step two: Generate a random number r. Suppose $r = 0.7$. Because $0.5 < r \leq 1$, in individual ch_2, an untouched component 4 is randomly chosen, and all components with the identical component value as component 4 in individual ch_2, namely components 2, 4, 6 and 9, are located. Afterwards, 1 is assigned to the counter ct, and the present ct value is written in the components 2, 4, 6 and 9 of the descendant individual d_1. Next, components 2, 4, 6 and 9 within individual ch_2 are labeled as touched. Step three: Generate a random number r. Suppose $r = 0.4$. Because $0 \leq r \leq 0.5$, in individual ch_1, an untouched component 8 is randomly chosen, and all components with the identical component value as component 8 in individual ch_1, namely components 8 and 9, are located. Afterwards, $ct + 1 = 2$ is assigned to ct, and write the current ct value into the component 8 of the descendant individual d_1. Due to the prevenient operation of the algorithm has written component 9 of d_1, this algorithm step doesn't change it. Component 9 of the individual d_1 remains unchanged. Afterwards, components 8 and 9 in individual ch_1 are labeled as touched. Step four: Generate a random number r. Suppose $r = 0.2$. Because $0 \leq r \leq 0.5$, in individual ch_1, an untouched component 2 is randomly chosen, and all components with the identical component value as component 2 in individual ch_1, namely components 2 and 6, are located. Because the components 2 and 6 of d_1 have been written in, the algorithm first labels components 2 and 6 of ch_1 as touched, and then randomly chooses another untouched component 5 in ch_1. Next, the algorithm locates all components with the identical component value as component 5 in individual ch_1, namely components 4 and 5. Afterwards, let $ct = ct + 1 = 3$, and write the current ct value

Algorithm 2 Pseudocode for Function Crossover() That Performs Randomized Two-Way Crossover Operation

Algorithm Parameters: Two parent individuals (i.e. chromosomes) ch_1 and ch_2 selected by tournament selection algorithm;
Algorithm Input: A matrix G representing a signed graph;
Algorithm Output: The two generated individuals;

```
1: for $i = 1$ to 2 do
2:    for $j = 1$ to $n$ do
3:       $ch_1\_s[j]$ = 'U'; //$ch_1\_s[j]$ = 'U' means $ch_1[j]$ has
            //not been touched yet, $ch_1\_s[j]$ = 'V' means
            //$ch_1[j]$ has been touched.
4:       $ch_2\_s[j]$ = 'U'; //$ch_2\_s[j]$ = 'U' means $ch_2[j]$ has
            //not been touched yet, $ch_2\_s[j]$ = 'V' means
            //$ch_2[j]$ has been touched.
5:       $d_i[j] = 0$;
6:    end for
7:    $ct = 1$;
8:    repeat
9:       Generate a random number $r$ between 0 and 1;
10:       if $0 \leq r \leq 0.5$ then
11:          Randomly choose an untouched component in
               $ch_1$, assuming that the chosen component is
               $ch_1[k]$, that is, $ch_1\_ s[k]$ = 'U';
12:          Locate all the component locations in $ch_1$ that
                have the same component value as the
                $ch_1[k]$, assuming that these components are
                $ch_1[k_1], ch_1[k_2], \ldots, ch_1[k_s]$;
13:         if $d_i[k_1] \neq 0 \wedge d_i[k_2] \neq 0 \wedge \cdots \wedge d_i[k_s] \neq 0$ then
14:            for each $u \in \{k_1, k_2, \ldots, k_s\}$ do
15:               $ch_1\_s[u]$ = 'V';
16:            end for
17:            goto 11;
18:         end if
19:         for each $u \in \{k_1, k_2, \ldots, k_s\}$ do
20:            if $d_i[u] == 0$ then
21:               $d_i[u] = ct$;
22:            end if
23:            $ch_1\_s[u]$ = 'V';
24:         end for
25:       else
26:         Randomly choose an untouched component in
               $ch_2$, assuming that the chosen component is
               $ch_2[k]$, that is, $ch_2\_ s[k]$ = 'U';
27:         Locate all the component locations in $ch_2$
                that have the same component value as the
                $ch_2[k]$, assuming that these components are
                $ch_2[k_1], ch_2[k_2], \ldots, ch_2[k_s]$;
28:         if $d_i[k_1] \neq 0 \wedge d_i[k_2] \neq 0 \wedge \cdots \wedge d_i[k_s] \neq 0$
                 then
29:            for each $u \in \{k_1, k_2, \ldots, k_s\}$ do
30:               $ch_2\_s[u]$ = 'V';
31:            end for
```

Algorithm 2 *(Continued.)* Pseudocode for Function Crossover() That Performs Randomized Two-Way Crossover Operation

```
32:             goto 26;
33:           end if
34:           for each $u \in \{k_1, k_2, \ldots, k_s\}$ do
35:             if $d_i[u] == 0$ then
36:               $d_i[u] = ct$;
37:             end if
38:             $ch_2\_s[u] = $ 'V';
39:           end for
40:         end if
41:         $ct = ct + 1$;
42:     until all component locations of the descendant
              individual $d_i$ are written in;
43: end for
44: return the generated descendant individuals $d_1$ and $d_2$;
```

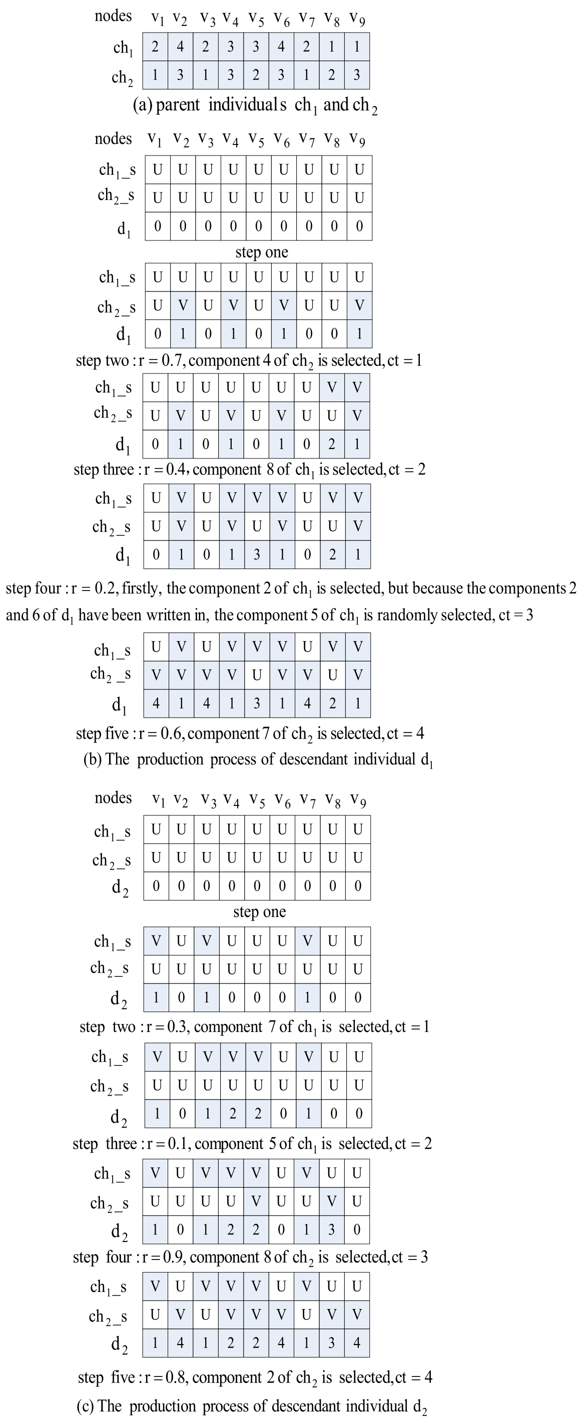
into the component 5 of the descendant individual d_1. Due to the prevenient operation of the algorithm has written component 4 of d_1, this operation doesn't change it. Then, components 4 and 5 in individual ch_1 are labeled as touched. Step five: Generate a random number r. Suppose $r = 0.6$. Because $0.5 < r \leq 1$, in individual ch_2, an untouched component 7 is randomly chosen, and all components with the identical component value as component 7 in individual ch_2, namely components 1, 3 and 7, are located. Afterwards, let $ct = ct + 1 = 4$, and write the current ct value into the components 1, 3 and 7 of the descendant individual d_1. Then, components 1, 3 and 7 in individual ch_2 are labeled as touched. At this time, the individual d_1 has been filled, so the work of producing descendant individual d_1 has been completed. We obtain descendant individual d_1. The generative steps of the other descendant individual d_2 are analogous to that of d_1. Figure 5 (c) shows the generation process of descendant individual d_2.

If the chromosome contains m communities on average, the time complexity of crossover operation is $O(m \times n)$. In the worst case, there are n communities in the parent chromosomes of crossover operation, then the time complexity of crossover operation is $O(n^2)$.

F. MUTATION OPERATOR

In the MACD-SN method, in addition to the frequently-used variation operation (this paper calls it the traditional variation operation), we also put forward the other variation operation (this paper calls it the community variation operation). In this paper, mutation and variation are interchangeable.

As indicated in Figure 3, in classical string-based coding method, every component denotes a node, and the value of component stands for the cluster containing the node. In the cluster identification methods, the frequently-used mutation operation first stochastically chooses a vertex v_s, after that stochastically chooses a neighbour vertex v_t of v_s (v_s and v_t are not included in the identical cluster), and at last assigns the cluster label (component value) of v_t to the component corresponding to vertex v_s [18]. The time complexity of frequently-used mutation operation is $O(1)$.



nodes	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
ch_1	2	4	2	3	3	4	2	1	1
ch_2	1	3	1	3	2	3	1	2	3

(a) parent individuals ch_1 and ch_2

nodes	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
ch_1_s	U	U	U	U	U	U	U	U	U
ch_2_s	U	U	U	U	U	U	U	U	U
d_1	0	0	0	0	0	0	0	0	0

step one

ch_1_s	U	U	U	U	U	U	U	U	U
ch_2_s	U	V	U	V	U	V	U	U	V
d_1	0	1	0	1	0	1	0	0	1

step two : $r = 0.7$, component 4 of ch_2 is selected, $ct = 1$

ch_1_s	U	U	U	U	U	U	U	V	V
ch_2_s	U	V	U	V	U	V	U	U	V
d_1	0	1	0	1	0	1	0	2	1

step three : $r = 0.4$, component 8 of ch_1 is selected, $ct = 2$

ch_1_s	U	V	U	V	V	V	U	V	V
ch_2_s	U	V	U	V	U	V	U	U	V
d_1	0	1	0	1	3	1	0	2	1

step four : $r = 0.2$, firstly, the component 2 of ch_1 is selected, but because the components 2 and 6 of d_1 have been written in, the component 5 of ch_1 is randomly selected, $ct = 3$

ch_1_s	U	V	U	V	V	V	U	V	V
ch_2_s	V	V	V	V	U	V	V	U	V
d_1	4	1	4	1	3	1	4	2	1

step five : $r = 0.6$, component 7 of ch_2 is selected, $ct = 4$

(b) The production process of descendant individual d_1

nodes	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9
ch_1_s	U	U	U	U	U	U	U	U	U
ch_2_s	U	U	U	U	U	U	U	U	U
d_2	0	0	0	0	0	0	0	0	0

step one

ch_1_s	V	U	V	U	U	U	V	U	U
ch_2_s	U	U	U	U	U	U	U	U	U
d_2	1	0	1	0	0	0	1	0	0

step two : $r = 0.3$, component 7 of ch_1 is selected, $ct = 1$

ch_1_s	V	U	V	V	V	U	V	U	U
ch_2_s	U	U	U	U	U	U	U	U	U
d_2	1	0	1	2	2	0	1	0	0

step three : $r = 0.1$, component 5 of ch_1 is selected, $ct = 2$

ch_1_s	V	U	V	V	V	U	V	U	U
ch_2_s	U	U	U	U	V	U	U	V	U
d_2	1	0	1	2	2	0	1	3	0

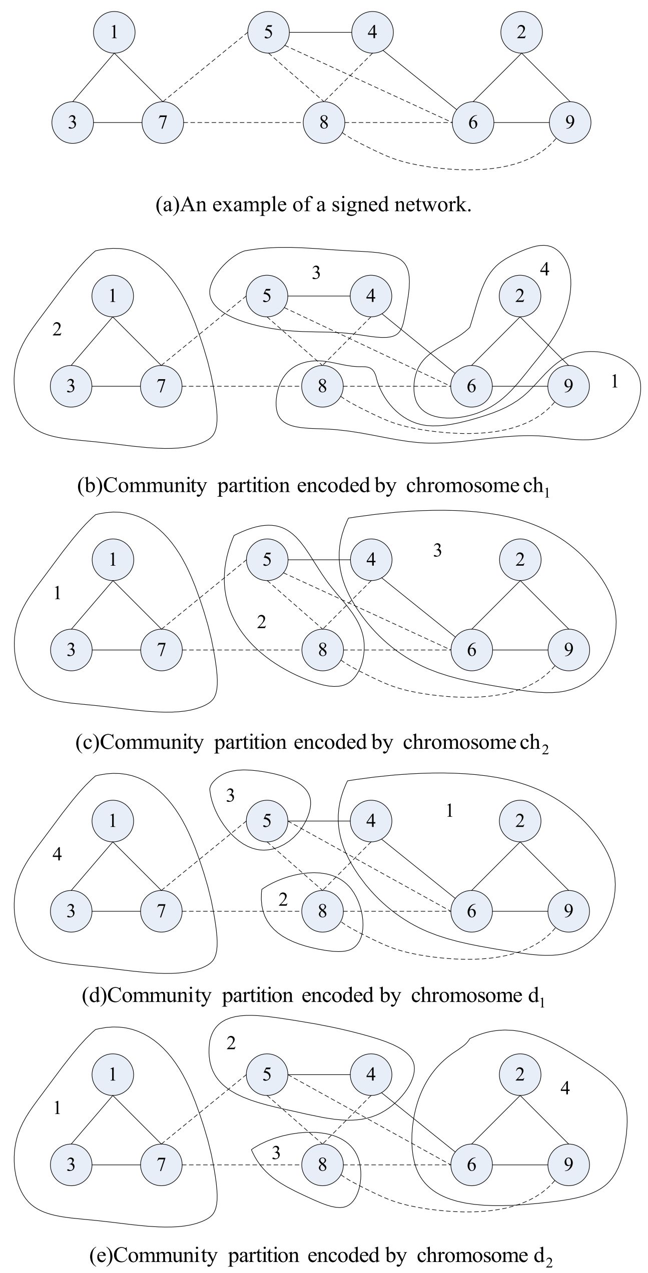
step four : $r = 0.9$, component 8 of ch_2 is selected, $ct = 3$

ch_1_s	V	U	V	V	V	U	V	U	U
ch_2_s	U	V	U	V	V	V	U	V	V
d_2	1	4	1	2	2	4	1	3	4

step five : $r = 0.8$, component 2 of ch_2 is selected, $ct = 4$

(c) The production process of descendant individual d_2

FIGURE 5. **An executive process of the crossover operator of the MACD-SN method. In the figure, U represents that the relevant component has been untouched, and V represents that the relevant component has been touched. The figure in the rectangle represents the label of the cluster that contains the relevant node. The value 0 represents that the relevant component has not been written. (a)Previous generation individuals ch_1 and ch_2. (b)The producing process of individual d_1. (c)The producing process of individual d_2.**



(a)An example of a signed network.

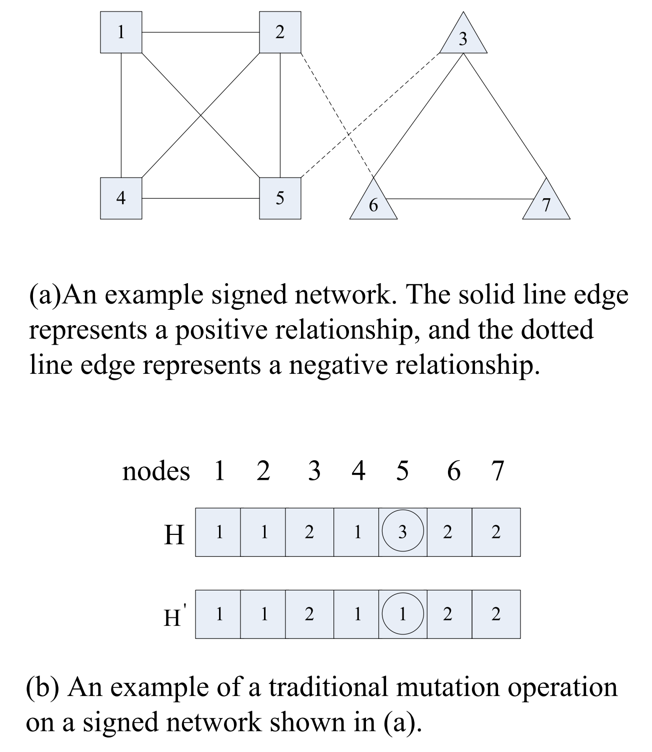
(b)Community partition encoded by chromosome ch_1

(c)Community partition encoded by chromosome ch_2

(d)Community partition encoded by chromosome d_1

(e)Community partition encoded by chromosome d_2

FIGURE 6. An example of community partition encoded by chromosomes ch_1, ch_2, d_1 and d_2 in Figure 5, respectively. A solid line indicates a positive edge and a dotted line indicates a negative edge.



(a)An example signed network. The solid line edge represents a positive relationship, and the dotted line edge represents a negative relationship.

(b) An example of a traditional mutation operation on a signed network shown in (a).

FIGURE 7. A schematic diagram of a traditional mutation operator.

Figure 7 shows an instance of using a traditional variation operation. As indicated in Figure 7, a chromosome H is selected first, and afterwards a vertex 5 on chromosome H is randomly chosen. The adjacent vertices of vertex 5 include vertices 1, 2, 3 and 4. Suppose that vertex 2 is randomly chosen. Thus, the component value of vertex 2 is assigned to the component corresponding to vertex 5.

Previous to describe community mutation operator in detail, the definition of community imbalance degree is first described.

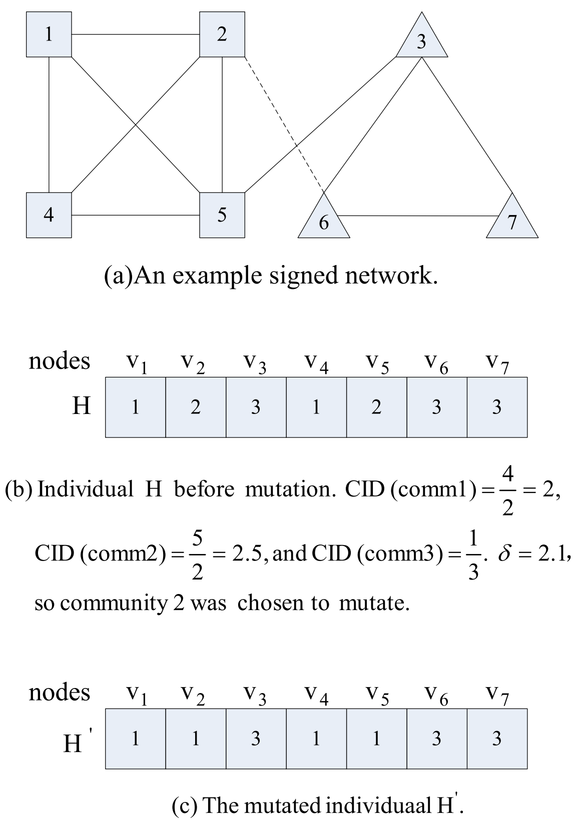
Definition 2 [Community Imbalance Degree (CID)]: Let *comm* be a community in the signed network G, then the community imbalance degree *CID* (*comm*) of community *comm* is calculated as follows:

$$CID(comm) = \frac{1}{n_C} \sum\nolimits_{v_i \in comm} NID(v_i, comm) \qquad (4)$$

where $\boldsymbol{n_C}$ is the number of nodes in community *comm*. The smaller the value of number of CID (*comm*), the more evident the cluster structure of cluster *comm*.

The community variation operation will be described in detail below. If the CID (*comm*) is higher than a parameter $\boldsymbol{\delta}$, we think that the vertices in *comm* can not constitute a signed cluster. At this point, we need to use community mutation operator to reallocate the vertices in *comm* to more suitable clusters. In algorithm 3, the operation steps of community mutation operator of the MACD-SN method are presented. Figure 8 is an explanatory drawing of the execution steps of community mutation operator of the MACD-SN method.

Figure 8(a) shows an example of a simple signed graph. Figure 8(b) illustrates an individual produced through the graph in Figure 8(a). This individual is composed of three clusters, i.e. cluster 1, 2 and 3. By equation (4), we may get $CID\ (comm1) = \frac{4}{2} = 2$, $CID\ (comm2) = \frac{5}{2} = 2.5$, and $CID\ (comm3) = 1/3$. Let $\boldsymbol{\delta} = 2.1$. Thus, cluster 2 is chosen for mutation. Among the neighbors of vertex $\boldsymbol{v_2}$, vertices $\boldsymbol{v_1}$ and $\boldsymbol{v_4}$ are its positive neighbors, and they are not in the same cluster as vertex $\boldsymbol{v_2}$. Since the cluster labels of vertices $\boldsymbol{v_1}$ and $\boldsymbol{v_4}$ are both 1, the cluster label of vertex $\boldsymbol{v_2}$ are set as 1. Among the positive neighbors of vertex $\boldsymbol{v_5}$, vertices $\boldsymbol{v_1}$, $\boldsymbol{v_3}$ and $\boldsymbol{v_4}$ are not in the same community as vertex $\boldsymbol{v_5}$. Suppose vertex



(a)An example signed network.

nodes	v_1	v_2	v_3	v_4	v_5	v_6	v_7
H	1	2	3	1	2	3	3

(b) Individual H before mutation. CID (comm1) $= \frac{4}{2} = 2$, CID (comm2) $= \frac{5}{2} = 2.5$, and CID (comm3) $= \frac{1}{3}$. $\delta = 2.1$, so community 2 was chosen to mutate.

nodes	v_1	v_2	v_3	v_4	v_5	v_6	v_7
H'	1	1	3	1	1	3	3

(c) The mutated individuaal H'.

FIGURE 8. An instance of the execution process of community variation operation of the MACD-SN method.

v_4 is selected randomly. Therefore, the community label 1 of vertex v_4 is assigned to vertex v_5.

The time complexity of community mutation operation is $O(n)$.

G. LOCAL SEARCH FUNCTION

For memetic algorithm, the local search process is a crucial part. A good local search subroutine may significantly increase the accuracy of the ultimate result of the algorithm and reduce the time to find the optimal solution. Therefore, this paper presents a subroutine called LocalSeek(), which achieves the above goals well. Algorithm 4 describes the details of the subroutine LocalSeek().

In algorithm 4, the codes in line 1 generate a random permutation of the natural numbers from 1 through n and put it in a set called *seq*. The iteration from 1 to n of lines 2 to 36 attempt to assign each node in the network to another more appropriate cluster (i.e. community) in the order of the nodes specified in the set *seq*. Obviously, according to the definition of the signed community, we should first try to put the current node i into the community to which most of its positive neighbors belong. When node i has no positive neighbors, we should try to put node i into the community that one of its negative neighbors belongs to. Algorithm 4 embodies this idea. The codes in line 5 put the clusters containing the most positive neighbours of the current vertex i into the set *comms*. (there may be multiple clusters that meet the condition).

Algorithm 3 Pseudocode for Function CommunityMutation() That Performs Community Mutation Operator

Algorithm Parameters: parent individual H, a threshold parameter δ;
Algorithm Input: A matrix G representing a signed graph;
Algorithm Output: improved individual H';

```
1: $comms$ = The set of clusters in $H$;
2: $k = -\infty$;
3: $m = 0$;
4: for each cluster $comm \in comms$ do
5:     if $CID(comm) > k$ then //$CID$ ($comm$) is the
               //community imbalance degree of community
               //$comm$.
6:         $k = CID(comm)$;
7:         $m = comm$;
8:     end if
9: end for
10: if $m \neq 0 \bigwedge CID(m) > \delta$ then
11:     $ns$ = All vertices in $m$;
12:     $H' = H$;
13:     while every vertex $v_s \in ns$ do
14:         Stochastically choose a positive neighbour $v_t$ of
              vertex $v_s$ ($v_s$ and $v_t$ are not included in the
              identical cluster);
15:         if there is $v_t$ satisfying the condition then
16:             $H'[v_s] = H'[v_t]$;
17:         end if
18:     end while
19: $qs_H = Q_s$ value of $H$ computed using formula (2);
20: $qs_{H'} = Q_s$ value of $H'$ computed using formula (2);
21: if $qs_{H'} > qs_H$ then
22:         return $H'$;
23:     end if
24: end if
25: return $H$;
```

The codes in lines 9 to 15 deal with the case where node i has positive neighbors. Suppose there are p clusters in *comms*, which are $comm_1, comm_2, \ldots, comm_p$. After node i is assigned to clusters $comm_1, comm_2, \ldots, comm_p$, the new clusters formed are $newComm_1, newComm_2, \ldots, newComm_p$, respectively. The codes in lines 9 to 15 are responsible for finding the community $comm_j$ so that the CID ($newComm_j$) is the minimum value in CID ($newComm_1$), CID ($newComm_2$), ..., $CID(newComm_p)$, that is, $comm_j = \arg\min_{comm_r \in comms} CID(comm_r \cup i)$. Obviously, at this point, $comm_j$ is the most suitable community in the set *comms* to merge with node i. The codes in lines 17 to 24 handle the case where node i has only negative neighbors. The codes in line 17 put the clusters containing the negative neighbours of vertex i into the set *comms*. Suppose there are q clusters in *comms*, which are $comm_1, comm_2, \ldots, comm_q$. After node i is assigned to clusters $comm_1, comm_2, \ldots, comm_q$, the new clusters formed are $newComm_1$,

Algorithm 4 LocalSeek

```
Algorithm Parameters: best individual bestOffspring in
the offspring individuals set, a threshold parameter $\rho$;
Algorithm Input: A matrix $G$ representing a signed graph;
Algorithm Output: revised individual bestOffspring;
1: A stochastic permutation of the natural numbers in
      the range of 1 to $n$ is generated and saved in a set
      called seq;
2: for $count = 1$ to $n$ do
3:    $i = seq[count]$;
4:    $comms = \emptyset$;
5:    Put the clusters (i.e. communities) containing most
         positive neighbors of vertex $i$ into the set
         comms(there may be multiple clusters that meet
         the condition);
6:    $k = \infty$;
7:    $m = 0$;
8:    if $comms \neq \emptyset$ then
9:       for each cluster $comm \in comms$ do
10:         $newComm$ = Put node $i$ into cluster $comm$ to
                  form a new cluster;
11:         if $CID\ (newComm) < k$ then //$CID\ (newComm)$
                  //is the community imbalance
                  //degree of community $newComm$.
12:            $k = CID\ (newComm)$;
13:            $m = comm$;
14:         end if
15:      end for
16:   else
17:      Put the clusters containing negative
         neighbors of vertex $i$ into the set $comms$;
18:      for each cluster $comm \in comms$ do
19:         $newComm$ = Put node $i$ into cluster $comm$ to
                  form a new cluster;
20:         if $CID(newComm) < k$ then //$CID\ (newComm)$
                  //is the community imbalance
                  //degree of community $newComm$.
21:            $k = CID(newComm)$;
22:            $m = comm$;
23:         end if
24:      end for
25:   end if
26:   if $m \neq 0$ then
27:      $newDivision$ = Put the vertex $i$ of bestOffspring into
                  cluster $m$ to produce a novel
                  individual;
28:      $qs_o = Q_s$ value of individual bestOffspring was
            computed through formula (2);
29:      $qs_n = Q_s$ value of individual newDivision
            was computed through formula (2);
30:      if $qs_n > qs_o$ then
31:         $bestOffspring = newDivision$;
32:      else if random number $r$ between 0-1 generated
                  randomly $< \rho$
33:         $bestOffspring = newDivision$;
34:      end if
35:   end if
36: end for
37: return bestOffspring;
```

$newComm_2, \ldots, newComm_q$, respectively. The codes in lines 18 to 24 are responsible for finding the cluster $comm_j$ so that the $CID\ (newComm_j)$ is the minimum value in $CID\ (newComm_1)$, $CID\ (newComm_2), \ldots, CID(newComm_q)$, that is, $comm_j = \arg\min_{comm_r \in comms} CID(comm_r \cup i)$. Obviously, at this point, $comm_j$ is the most suitable cluster in the set $comms$ to merge with node i. The codes in lines 26 to 35 are responsible for deciding whether node i can be placed in the cluster $comm_j$ (i.e. cluster m) found. If possible, place node i in the cluster m. The codes in line 27 put the node i of the individual *bestOffspring* into the cluster m to form a new individual *newDivision*. The codes in line 28 use equation (2) to calculate the Q_s value for individual *bestOffspring* and assign the calculated Q_s value to variable qs_o. The codes in line 29 use equation (2) to calculate the Q_s value for individual *newDivision* and assign the calculated Q_s value to variable qs_n. The codes in line 30 determines if qs_n is greater than qs_o. If qs_n is greater than qs_o, the codes in line 31 take the individual *newDivision* as the best individual *bestOffspring* in the offspring population. If qs_n is not greater than qs_o, the codes in line 32 randomly generate a real number r between 0 and 1, and determines whether it is less than the input parameter ρ of the algorithm. If less than, the codes in line 33 take the individual *newDivision* as the best individual *bestOffspring* in the offspring population. The codes in line 37 return the improved individual *bestOffspring*.

Our subroutine LocalSeek() has one outstanding merit. This merit is that it may accept a worse result within a certain odds. In the process of solving, this merit enables LocalSeek() to increase the accuracy of the result of the algorithm, and also helps the algorithm jump away from the local best solution. In algorithm 4, the conditional statement in line 32 implements this function. When the algorithms find the local optimal solution, some other algorithms will terminate the iteration. On account of some other algorithms can not find the global best solution by movement in a small area near the local best solution. Nevertheless, the LocalSeek() presented in this paper can do this with a specified odds, namely, it may adopt a result with a certain odds that is worse than the present result. This is helpful for the subroutine to jump out of the local best result and attain the global best result after a few moving operations.

In algorithm 4, the time complexity of the statement in line 1 is $O(n \times \log n)$. Each time the for loop of line 2 is executed, it needs to iterate n times. It is assumed that the average degree of nodes in the network is d ($d \ll n$). The time complexity of the statement in line 5 is $O(d)$. In the worst case, the for loop of line 9 needs to iterate $n - 1$ times for each execution. Similarly, in the worst case, the for loop of line 18 needs to iterate $n - 1$ times for each execution. The time complexity of the statement in line 17 is $O(d)$. Because $d \ll n$, the time complexity of LocalSeek() is O (n^2).

Algorithm 5 gives the pseudo code of the main function of MACD-SN algorithm.

Algorithm 5 MACD-SN Method

```
Algorithm Parameters: population size $popu\_size$, number of parent individuals selected by tournament selection
  algorithm $k$, crossover probability $p_1$, mutation probability $p_2$ of traditional mutation operator, mutation
  probability $p_3$ of community mutation operator, a threshold parameter $\delta$ used in community mutation
  operator, a threshold parameter $\rho$ used in the function LocalSeek(), amount of iterations without revision $gt$;
Algorithm Input: A matrix $G$ representing a signed graph;
Algorithm Output: a cluster partition $comms$ of the signed graph;
1: $popu$ = initialize($popu\_size$); //produce initial individual
   //population;
2: repeat
3:   Use formula (2) to compute the fitness function
       value of every individual in the population $popu$;
4:   for $i = 1$ to $k$ do //Using tournament selection
       //algorithm to select parent individuals for
       //subsequent genetic operations.
5:     Randomly select $k$ individuals from the
       population set $popu$;
6:     Using the selected $k$ individuals to build a
       maximum heap;
7:     $offs[i]$ = The top element of the maximum heap;
         //$offs[]$ is a individuals set selected for
         //subsequent genetic operations.
8:   end for
9:   $i = 1$;
10:  while $i \leq k$-1 do
11:    if rand(0,1) $\leq p_1$ then
12:      ($offs[i]$, $offs[i+1]$)=Crossover($offs[i]$, $offs[i+1]$);
           //Perform a randomized two-way crossover
           //operation. The Crossover function returns
           //two individuals, $offs[i]$ and $offs[i+1]$.
13:    end if
14:    $i = i + 2$;
15:  end while
16:  for $i = 1$ to $k$ do
17:    if rand(0,1) $\leq p_2$ then
18:      $offs[i]$ = The traditional mutation operator is
           executed on the $offs[i]$;
19:    end if
20:  end for
21:  for $i = 1$ to $k$ do
22:    if rand(0,1) $\leq p_3$ then
23:      $offs[i]$ = CommunityMutation($offs[i]$, $\delta$); //The
           //community mutation operator is executed
           //on the $offs[i]$;
24:    end if
25:  end for
26:  $bestOffspring$ = The individual with the maximum
       fitness function value in $offs$;
```

Algorithm 5 *(Continued.)* MACD-SN Method

```
27:  $bestOffspring$ = LocalSeek($bestOffspring$, $\rho$);
28:  $popu = popu + offs + bestOffspring$;
29:  $popu$=The set of the first $popu\_size$ individuals
       with the largest fitness in the population $popu$;
30: until no improved amount of iterations for the
       optimal individual in population $popu >= gt$;
31: $comms$ = Cluster partition of individual with the
       biggest fitness function value in the population
       $popu$;
32: return $comms$;
```

According to the analysis of the prevenient sections, in algorithm 5, the time complexity of the statement in line 1 is $O(popu\_size \times n^2)$. The time complexity of the statement in line 3 is $O(popu\_size)$. The time complexity of the codes in lines 4 to 8 is $O(k^2)$(k is the number of parent chromosomes to be selected by the tournament selection algorithm.). On average, the time complexity of the codes in lines 10 to 15 is $O(k \times m \times n)$(m is the average number of communities contained in a single chromosome.). The time complexity of the codes in lines 16 to 20 is $O(k)$. The time complexity of the codes in lines 21 to 25 is $O(k \times n)$. The time complexity of the statement in line 26 is $O(k)$. The time complexity of the statement in line 27 is $O(n^2)$. The time complexity of the statement in line 29 is $O(popu\_size \times \log(popu\_size))$. The time complexity of the statement in line 31 is $O(popu\_size+n)$. Through a large number of experiments, we can see that the average execution times of repeat statement in line 2 is about $popu\_size/3$. Therefore, the time complexity of MACD-SN algorithm is $O(popu\_size \times k \times m \times n + popu\_size \times n^2)$.

V. EXPERIMENTAL RESULTS

In our experiment, the MACD-SN algorithm proposed in this paper is verified in the synthetic and real-world signed networks. We also make comparisons with other four famous algorithms, namely, FEC [29], SSL [36], DM [7] and SISN [37]. We use normalized mutual information (NMI) [62] to assess the capability of the methods. Given that A and B are two partitions of a network, then $NMI\ (A, B)$ is defined as follows.

$$NMI(A, \boldsymbol{B}) = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} C_{ij} \log\left(\frac{C_{ij}N}{C_{i.}C_{.j}}\right)}{\sum_{i=1}^{C_A} C_{i.} \log\left(\frac{C_{i.}}{N}\right) + \sum_{j=1}^{C_B} C_{.j} \log\left(\frac{C_{.j}}{N}\right)} \quad (5)$$

where N is the number of nodes of the network, C is a confusion matrix. C_{ij} equals to the number of nodes shared in common by community i in partition A and by community j in partition B. C_A (or C_B) is the number of clusters in partition A (or B), $C_{i.}$ (or $C_{.j}$) is the sum of elements of C in row i (or column j). NMI takes value between 0 and 1. The larger the value of number of NMI, the more evident the cluster structure obtained.

In this paper, the source codes of FEC, SSL, DM and SISN are obtained from the original author. MACD-SN is implemented by C# 4.0 using Microsoft Visual Studio 2010. In the experiment, we set the parameters $popu\_size = 1000$, $k = 500, p_1 = p_2 = p_3 = 0.3, \delta = 3.9, gt = 3$, and $\rho = 0.15$.

A. SYNTHETIC SIGNED NETWORKS

By using the generation model in reference [29], the synthetic signed networks used in our experiment are produced. This model can be described as

$$SG(c, n, k, pm_{in}, pm-, pm+)$$

where c is the number of communities, n is the number of vertices in each community, k is the average degree of the vertices, pm_{in} is the probability that a link falls within a community, *pm-* is the probability that a link within communities is negative, and $pm+$ is the probability that a link between communities is positive. *pm-* and $pm+$ are also known as the yawp parameters.

In order to verify our method effectively, in our experiment, we produce five kinds of synthetic signed networks, which have different characteristics. These kinds of synthetic signed networks include two different types of networks: balanced networks and unbalanced networks. Among them, only networks 1 is balanced, and other kinds of networks are imbalanced.

Networks 1: The generation model of this kind of networks is SG (6, 42, 42, pm_{in}, 0, 0). Among them, in the interval [0, 1], pm_{in} increases gradually, and the increment of each step is 0.1. Because the positive edges are within the clusters and the negative edges lie between the clusters, this kind of signed networks are balanced. We carry out five methods on 11 networks of this kind of networks. In Figure 9, we describe the results of the five algorithms.

As shown in Figure 9, the MACD-SN and SISN can accurately discover all communities in the networks. This means that the two algorithms are not sensitive to the variation of the parameter pm_{in} and they have good detection ability within the balanced signed graphs. The SSL algorithm also has excellent detection ability, when $pm_{in} \leq 0.8$, it can correctly identify the communities.

Networks 2: The generation model of this kind of networks is SG(6, 42, 42, 0.5, *pm-*, 0). Among them, in the interval [0, 1], *pm-* increases gradually, and the increment of each step is 0.1. This kind of signed networks is imbalanced, but the yawp only consists in the clusters. In these signed networks, the positive edges do not exist between clusters. The greater the numerical value of *pm-*, the more minus edges the clusters contain.

The outputs of the five methods running on Networks 2 are shown in Figure 10. As we can see, when *pm-* varies from 0 to 1, the numerical values of *NMI* of the MACD-SN algorithm and SSL algorithm are both 1. This result shows that MACD-SN and SSL can perfectly identify the community structures in this kind of signed networks. The performance of the other three algorithms is relatively poor.

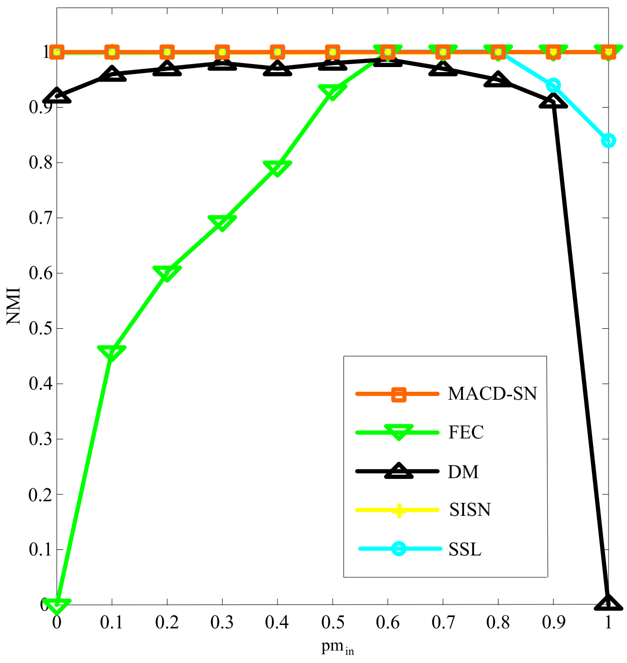


FIGURE 9. The output of MACD-SN algorithm and four comparison algorithms on Networks 1.

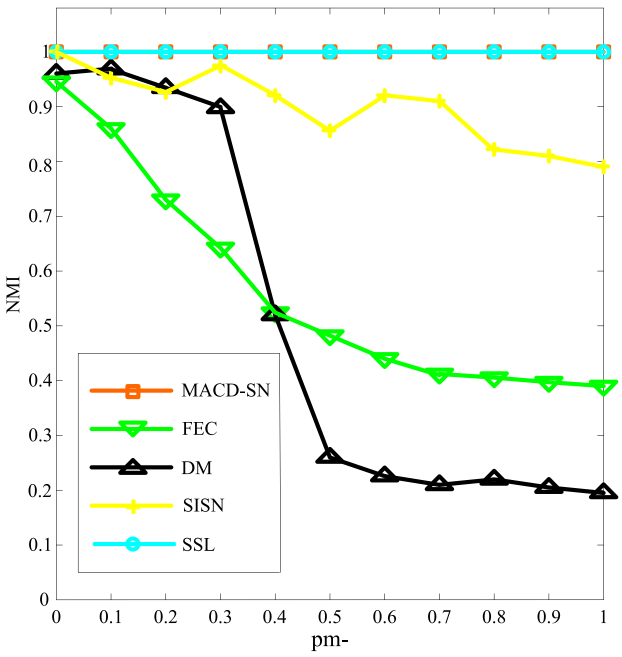


FIGURE 10. The outputs of MACD-SN algorithm and four comparison algorithms on Networks 2.

Networks 3: The generation model of this kind of networks is SG(6, 42, 42, 0.5, 0, $pm+$). Among them, in the interval [0, 1], $pm+$ increases gradually, and the increment of each step is 0.1. In this kind of networks, noises only exists between clusters, and there are no negative edges in the clusters, so this kind of signed networks is imbalanced. The greater the numerical value of $pm+$, the more positive edges there are between clusters in the network.

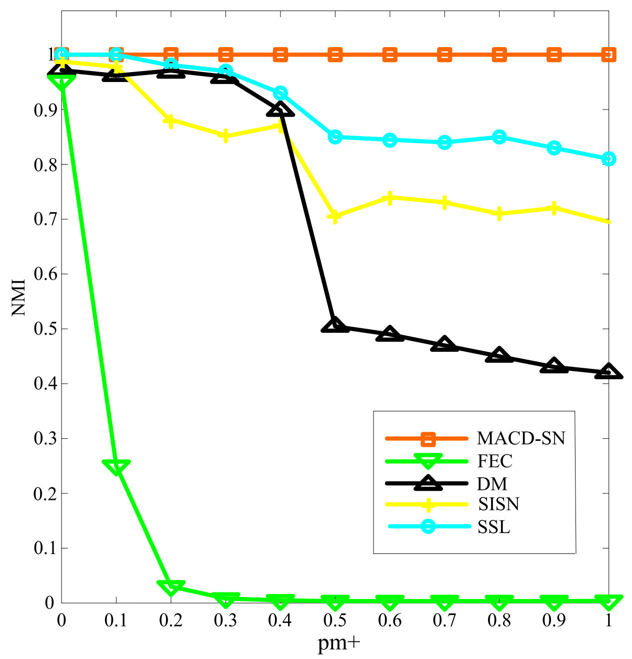


FIGURE 11. The outputs of MACD-SN algorithm and four comparison algorithms on Networks 3.

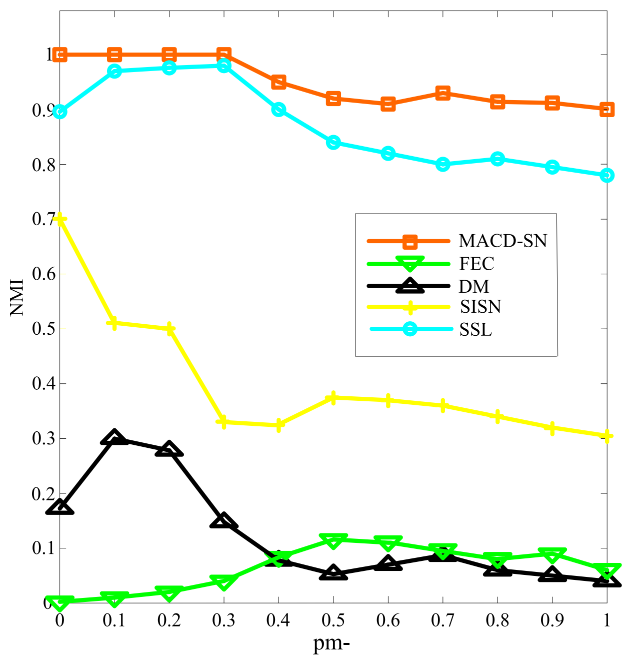


FIGURE 12. The outputs of MACD-SN algorithm and four comparison algorithms on Networks 4.

Figure 11 shows the outputs of the five methods on Networks 3. In Figure 11, we may observe that MACD-SN is capable of identifying the clusters in the networks well, but when $pm+ > 0.1$, the detection ability of SSL decreases. On the whole, the sorting results of the five algorithms by performance are MACD-SN, SSL, SISN, DM and FEC.

Networks 4: The generation model of this kind of networks is SG(6, 42, 42, 0.5, *pm-*, 0.5). Among them, in the interval [0, 1], *pm-* increases gradually, and the increment of each step is 0.1. In this kind of networks, not only the negative edges exist within the clusters, but also the positive edges exist between the clusters, so this kind of signed networks is imbalanced. The greater the numerical value of *pm-*, the more minus edges the clusters contain.

The outputs of the five methods running on Networks 4 are shown in Figure 12. We can see that in Figure 12, when $pm- \leq 0.3$, MACD-SN can perfectly identify communities in the networks; when $pm- > 0.3$, the performance of MACD-SN decreases. The SSL also has good accuracy, but the DM and FEC have very bad accuracy. Among the five algorithms, the performance of SISN is in the third place.

Networks 5: The generation model of this kind of networks is SG(6, 42, 42, 0.5, 0.5, *pm+*). Among them, in the interval [0, 1], *pm+* increases gradually, and the increment of each step is 0.1. Due to some negative edges consist in the clusters and a number of positive edges consist between the clusters, this kind of signed networks is imbalanced. The greater the numerical value of *pm+*, the more positive edges there are between clusters in the network.

Figure 13 shows the results of the five methods running on networks 5. We can see that in Figure 13, MACD-SN has the

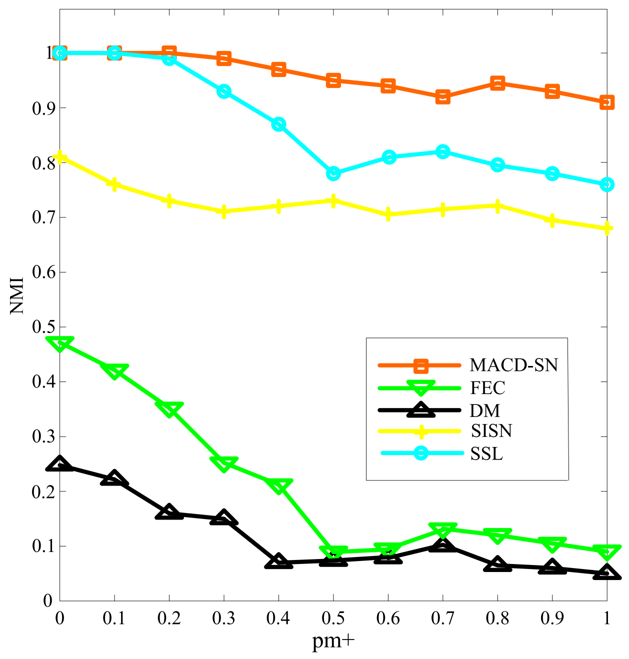


FIGURE 13. The outputs of MACD-SN algorithm and four comparison algorithms on Networks 5.

best accuracy among the five algorithms. SSL is in second place. The rest are in order: SISN, FEC, and DM. In order to further prove the high performance of our presented algorithm, in Chart 1, we show the average values of *NMI*. These average values are derived from the results of every method executing on the above five kinds of signed networks. In Chart 1, we may observe that our algorithm shows the best accuracy among the five comparison algorithms.

Chart 1. **The average NMI values of the five algorithms.**

	MACD-SN	SSL	SISN	DM	FEC
Networks I	**1**	0.9805	**1**	0.8746	0.7646
Networks II	**1**	**1**	0.8985	0.5091	0.5323
Networks III	**1**	0.9014	0.806	0.6846	0.114
Networks IV	**0.9494**	0.8699	0.3991	0.1225	0.0637
Networks V	**0.9597**	0.8659	0.7251	0.1185	0.2116

B. REAL SIGNED NETWORKS

About the real signed networks, in the experiment, we choose two real-world networks with true cluster partition and three real-world networks without true cluster partition to test our method. Two real signed networks that possess true cluster partition are Gahuku-Gama subtribes network (GGSN) [39] and Slovene parliamentary party network (SPPN) [38], separately. Three real signed networks that lack true cluster partition are Slashdot network [40], Country network [41], and Epinions network [40].

The SPPN shows the relationships among the Slovenian parliament's 10 political parties in 1994. The plus and minus edges separately represent the alike and unalike relationships between the political parties. The true cluster partition of the SPPN consists of two clusters. The GGSN was created based on Reads study on the cultures of the Eastern Central Highlands of New Guinea. This network describes the political alliance and enmities among the 16 Gahuku-Gama subtribes, which were distributed in a particular area and were engaged in warfare with one another in 1954. The positive and negative links of the network correspond to political arrangements with positive and negative ties, respectively.

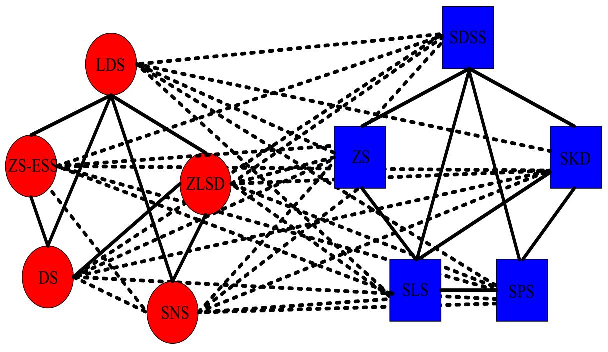


FIGURE 14. **Results of the MACD-SN running on SPPN.**

The results of our algorithm running on SPPN are shown in Figure 14. In Figure 14, we may observe that MACD-SN detects two clusters on SPPN. The red circle vertices are in one cluster, and the blue square vertices are in the other cluster. Since all the plus edges are in the clusters and all the minus edges are between clusters, SPPN is balanceable. The final results of our algorithm is in line with the true cluster partition of SPPN.

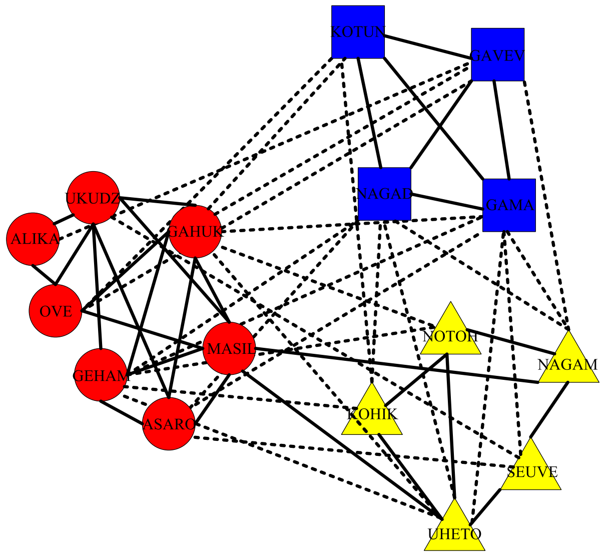


FIGURE 15. **Results of the MACD-SN running on GGSN.**

The results of MACD-SN algorithm running on GGSN are shown in Figure 15. In Figure 15, the vertices with the identical color and shape are in the same cluster. There are three clusters in GGSN. In Figure 15, we can see that most positive edges are in the clusters, and all negative edges are between the clusters. This shows that GGSN is an imbalanced signed network. The outputs of MACD-SN algorithm is in line with the true cluster partition of the GGSN.

The Country network was generated using the Correlates of War data set from 1996 to 1999 [41]. The vertices represent the nations, the plus edges denote the military leagues, and the minus edges represent the military antagonisms. In the tests, we remove insular vertices in the graph and all connected components except the largest connected components in the graph, and only retain the maximal connected component. This maximal component consists of 144 vertices and 1243 edges.

MACD-SN algorithm puts all the vertices into eight clusters. The outcomes of the algorithm are presented in Figures 16 and 17. Figure 16 visually presents the cluster

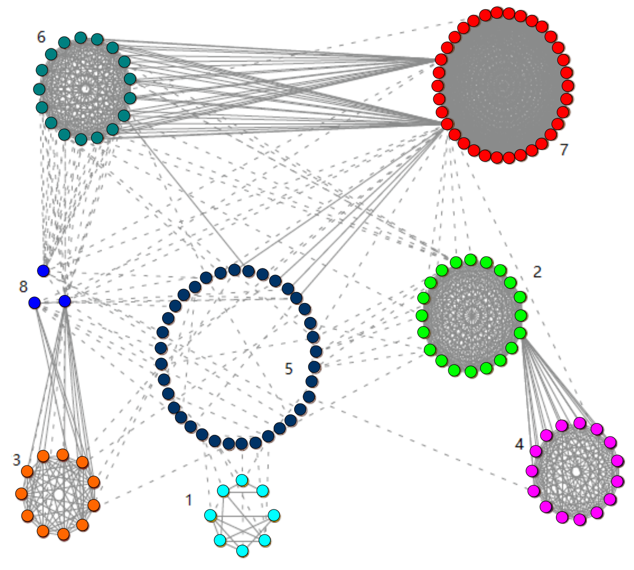


FIGURE 16. Result of the MACD-SN running on Country network.

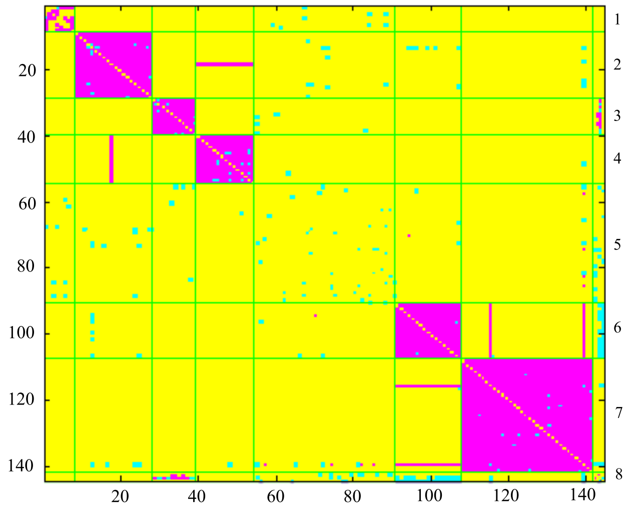


FIGURE 17. Result of the MACD-SN running on Country network.

partition of the Country network. In Figure 16, the minor circles are used to represent the vertices, solid lines are used to represent the positive edges, and dashed lines are used to represent the negative edges. The vertices with the identical color are in the identical cluster and constitute the larger circle. Figure 17 shows an adjacency matrix rearranged based on cluster partition. In Figure 17, pink and blue dots represent positive and negative edges, severally. Eight clusters of vertices are divided by solid lines with a green color. The number on the right in Figure 17 is the number of the clusters, which have a one-to-one correspondence with the number near the big circles in Figure 16.

In Figures 16 and 17, we can see that, (1) Among the eight clusters, clusters 2, 3, 4, 6 and 7 have dense positive edges. (2) Among the eight clusters, there are two clusters with relatively sparse positive edges, which are clusters 1 and 8 respectively. (3) The cluster 5 is composed of many peripheral vertices, in which there are few edges.

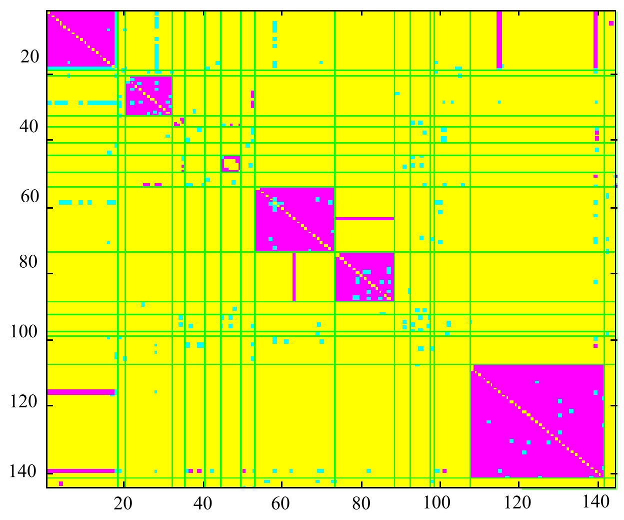


FIGURE 18. Output of SSL running on the Country network.

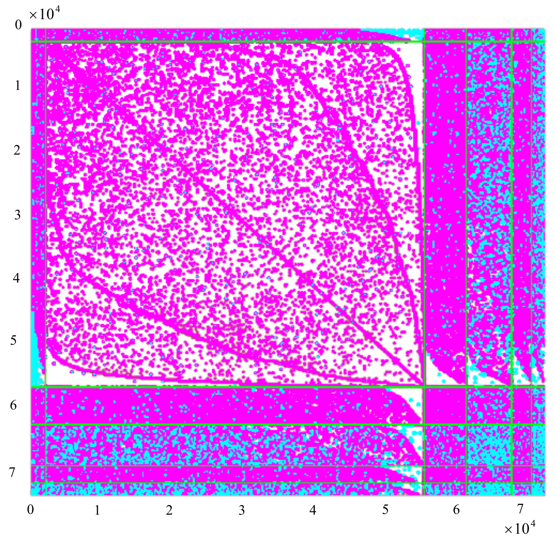


FIGURE 19. Result of the MACD-SN running on Slashdot network.

(4) The negative edges mainly exist between clusters. (5) There are no negative edges between the clusters 2 and 4, but there are a large amount of positive edges between one vertex of the cluster 2 and the vertices in the cluster 4. (6) There are few edges between clusters 6 and 7, but there are a large amount of positive edges between two vertices of the cluster 7 and the vertices in the cluster 6.

We also compare the result of our algorithm with that of SSL. Figure 18 presents the output of the SSL on the Country signed graph. In Figure 18, we may observe that the running result of MACD-SN is significantly superior to that of SSL.

We also run our algorithm MACD-SN on the Slashdot network and Epinions network [40]. Epicions is a famous network of consumers reviews. Users may believe or not believe other users reviews. Slashdot is a debate website where users can see other users as pals or foes. After all the insular vertices within the network are removed, 126828 vertices are left in the Epinions network and 73099 in the Slashdot network. In our experiment, the two remaining networks were used.

In the Epinions network, our algorithm identifies sixteen clusters. In Slashdot network, our algorithm identifies six clusters. Figure 19 shows an adjacency matrix of the Slashdot network rearranged based on cluster partition. In Figure 19, pink and blue dots represent positive and negative edges, respectively. Six clusters of vertices are divided by solid lines with a green color. According to the order from top left to bottom right on the main diagonal, the number of vertices in each cluster is 1741, 54398, 5922, 6413, 2746 and 1879, severally. In Figure 19, we may observe that the largest cluster with 54398 vertices is loose, and the rest of clusters are dense. Among the six clusters, the clusters with 6413 vertices and 1879 vertices include a large number of negative edges, while the clusters with 1741 vertices, 5922 vertices and 2746 vertices include a large number of positive edges.

VI. CONCLUSION

In the research field of signed graphs, cluster structure is an important network feature. For the sake of better study and take advantage of the signed networks, it is crucial to discover their cluster partition. In this paper, we propose a memetic method named MACD-SN for cluster partitions in signed networks. The individual coding method of MACD-SN algorithm adopts the well-known string-based coding method. In order to speed up the convergence, we proposed a new initialization algorithm for the cluster partitions of signed networks. The fitness function of MACD-SN algorithm uses the Q_s function presented in [21]. In order to select parent individuals for succedent genetic operations, we adopt a well-known operator (i.e. tournament selection operator), which provides chromosomes in the parent population identical probabilities to be chosen for subsequent genetic operators. In addition to the frequently-used mutation operation, this paper also presents a novel crossover operation and a novel mutation operation. The novel randomized two-way crossover operation can preferable retain the hereditary properties of the previous generation individuals, and the novel community mutation operator may greatly enhance the population diversity. Moreover, this paper presents a novel local search subroutine, which may enhance the accuracy of the ultimate output of the MACD-SN and reduce its running time, and enable the algorithm to jump out of the local best solution with a specified odds and attain the global best solution. For verifying the detection ability of the proposed algorithm, a large number of tests have been executed on five kinds of synthetic signed graphs and five real signed graphs. Next, we compare the test outcomes with four well-known signed network cluster partition methods. The comparison outcomes show that the performance of MACD-SN method is better than the other four methods, which indicates that the method proposed in this paper is an excellent method to identify cluster partitions in signed networks. The disadvantage of MACD-SN algorithm is that it can't detect overlapping communities in signed networks. We will solve this problem in our future work.

REFERENCES

[1] Z. Xia and Z. Bu, "Community detection based on a semantic network," *Knowl.-Based Syst.*, vol. 26, pp. 30–39, Feb. 2012.

[2] P. Doreian and A. Mrvar, "Partitioning signed social networks," *Soc. Netw.*, vol. 31, pp. 1–11, Dec. 2009.

[3] M. E. J. Newman, "The structure of scientific collaboration networks," *Proc. Natl. Acad. Sci. USA*, vol. 9781400841356, pp. 221–226, Oct. 2011.

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

[5] 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.

[6] K. P. Reddy, M. Kitsuregawa, P. Sreekanth, and S. S. Rao, "A graph based approach to extract a neighborhood customer community for collaborative filtering," in *Databases in Networked Information Systems*. Berlin, Germany: Springer, 2002, pp. 188–200.

[7] P. Doreian and A. Mrvar, "A partitioning approach to structural balance," *Social Netw.*, vol. 18, no. 2, pp. 149–168, Apr. 1996.

[8] B. Yang, X. Zhao, and X. Liu, "Bayesian approach to modeling and detecting communities in signed network," in *Proc. AAAI*, Austin, TX, USA, 2015, pp. 1952–1958.

[9] D. Cartwright and F. Harary, "Structural balance: A generalization of Heider's theory," *Psychol. Rev.*, vol. 63, no. 5, pp. 277–293, 1956.

[10] J. A. Davis, "Clustering and structural balance in graphs," *Hum. Relations*, vol. 20, no. 2, pp. 181–187, May 1967.

[11] 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.

[12] 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.

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

[14] C. Shi, Z. Yan, Y. Cai, and B. Wu, "Multi-objective community detection in complex networks," *Appl. Soft Comput.*, vol. 12, pp. 850–859, Oct. 2012.

[15] 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, Aug. 2012.

[16] D. Chen, F. Zou, R. Lu, L. Yu, Z. Li, and J. Wang, "Multi-objective optimization of community detection using discrete teaching–learning-based optimization with decomposition," *Inf. Sci.*, vol. 369, pp. 402–418, Nov. 2016.

[17] F. Zou, D. Chen, S. Li, R. Lu, and M. Lin, "Community detection in complex networks: Multi-objective discrete backtracking search optimization algorithm with decomposition," *Appl. Soft Comput.*, vol. 53, pp. 285–295, Apr. 2017.

[18] 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.

[19] J. Kunegis, S. Schmidt, A. Lommatzsch, J. Lerner, E. W. De Luca, and S. Albayrak, "Spectral analysis of signed graphs for clustering, prediction and visualization," in *Proc. SIAM Int. Conf. Data Mining*, Apr. 2010, pp. 559–570.

[20] K.-Y. Chiang, C.-J. Hsieh, N. Natarajan, I. S. Dhillon, and A. Tewari, "Prediction and clustering in signed networks: A local to global perspective," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1177–1213, 2014.

[21] S. Gómez, P. Jensen, and A. Arenas, "Analysis of community structure in networks of correlated data," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 80, no. 1, Jul. 2009, Art. no. 016114.

[22] V. A. Traag and J. Bruggeman, "Community detection in networks with positive and negative links," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 80, no. 3, Sep. 2009, Art. no. 036115.

[23] H. W. Shen, *Community Structure: An Introduction*. Berlin, Germany: Springer, 2013.

[24] R. Figueiredo and G. Moura, "Mixed integer programming formulations for clustering problems related to structural balance," *Social Netw.*, vol. 35, no. 4, pp. 639–651, Oct. 2013.

[25] P. Anchuri and M. Magdon-Ismail, "Communities and balance in signed networks: A spectral approach," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, Aug. 2012, pp. 235–242.

[26] K. Y. Chiang, J. J. Whang, and I. S. Dhillon, "Scalable clustering of signed networks using balance normalized cut," in *Proc. 21st ACM Int. Conf. Inf. Knowl. Manage.*, vol. 2012, pp. 615–624.

[27] A. Amelio and C. Pizzuti, "Community mining in signed networks: A multiobjective approach," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining*, 2013, pp. 95–99.
[28] Y. Li, J. Liu, and C. Liu, "A comparative analysis of evolutionary and memetic algorithms for community detection from signed social networks," *Soft Comput.*, vol. 18, no. 2, pp. 329–348, Feb. 2014.
[29] B. Yang, W. Cheung, and J. Liu, "Community mining from signed social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 10, pp. 1333–1348, Oct. 2007.
[30] Q. Cai, M. Gong, S. Ruan, Q. Miao, and H. Du, "Network structural balance based on evolutionary multiobjective optimization: A two-step approach," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 903–916, Dec. 2015.
[31] C. Liu, J. Liu, and Z. Jiang, "A multiobjective evolutionary algorithm based on similarity for community detection from signed social networks," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2274–2287, Dec. 2014.
[32] J. Huang, H. Sun, Y. Liu, Q. Song, and T. Weninger, "Towards online multiresolution community detection in large-scale networks," *PLoS ONE*, vol. 6, no. 8, Aug. 2011, Art. no. e23829.
[33] J. Q. Jiang, "Stochastic block model and exploratory analysis in signed networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 91, no. 6, Jun. 2015, Art. no. 062805.
[34] R. Harakawa, T. Ogawa, and M. Haseyama, "Extracting hierarchical structure of Web video groups based on sentiment-aware signed network analysis," *IEEE Access*, vol. 5, pp. 16963–16973, 2017.
[35] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA, USA: Addison-Wesley, 1989.
[36] B. Yang, X. Liu, Y. Li, and X. Zhao, "Stochastic blockmodeling and variational bayes learning for signed network analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 9, pp. 2026–2039, Sep. 2017.
[37] X. Zhao, B. Yang, X. Liu, and H. Chen, "Statistical inference for community detection in signed networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 95, no. 4, Apr. 2017, Art. no. 042313.
[38] S. Kropivnik and A. Mrvar, "An analysis of the slovene parliamentary parties network," *Dev. Stat. Methodol*, vol. 12, pp. 209–216, Oct. 1996.
[39] K. E. Read, "Cultures of the central highlands, new guinea," *Southwestern J. Anthropology*, vol. 10, no. 1, pp. 1–43, Apr. 1954.
[40] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed networks in social media," in *Proc. 28th Int. Conf. Hum. Factors Comput. Syst. (CHI)*, 2010, pp. 1361–1370.
[41] P. Doreian and A. Mrvar, "Structural balance and signed international relations," *J. Social Struct.*, vol. 16, no. 1, pp. 1–49, 2019.
[42] J. Hua, J. Yu, and M.-S. Yang, "Fast clustering for signed graphs based on random walk gap," *Social Netw.*, vol. 60, pp. 113–128, Jan. 2020.
[43] M. J. Brusco and P. Doreian, "Partitioning signed networks using relocation heuristics, tabu search, and variable neighborhood search," *Social Netw.*, vol. 56, pp. 70–80, Jan. 2019.
[44] B. A. Attea, H. M. Rada, M. N. Abbas, and S. Özdemir, "A new evolutionary multi-objective community mining algorithm for signed networks," *Appl. Soft Comput.*, vol. 85, Dec. 2019, Art. no. 105817.
[45] X. Zhu, Y. Ma, and Z. Liu, "A novel evolutionary algorithm on communities detection in signed networks," *Phys. A, Stat. Mech. Appl.*, vol. 503, pp. 938–946, Aug. 2018.
[46] S. Ping, D. Liu, B. Yang, Y. Zhu, H. Chen, and Z. Wang, "Community detection in signed networks based on the signed stochastic block model and exact ICL," *IEEE Access*, vol. 7, pp. 53667–53676, 2019.
[47] J. Chen, U. Liji, H. Wang, and Z. Yan, "Community mining in signed networks based on dynamic mechanism," *IEEE Syst. J.*, vol. 13, no. 1, pp. 447–455, Mar. 2019.
[48] C. Yan and Z. Chang, "Modularized convex nonnegative matrix factorization for community detection in signed and unsigned networks," *Phys. A, Stat. Mech. Appl.*, vol. 539, Feb. 2020, Art. no. 122904.
[49] X. Liu, W. Song, K. Musial, X. Zhao, W. Zuo, and B. Yang, "Semi-supervised stochastic blockmodel for structure analysis of signed networks," *Knowl.-Based Syst.*, vol. 195, May 2020, Art. no. 105714.
[50] D. Liu, Y. Zhang, R. Liang, B. Li, and Z. Xia, "Signed network community mining based on fine-grained signed stochastic block model," in *Proc. 2nd Int. Conf. Artif. Intell. Big Data (ICAIBD)*, May 2019, pp. 329–333.
[51] F. Bonchi, E. Galimberti, A. Gionis, B. Ordozgoiti, and G. Ruffo, "Discovering polarized communities in signed networks," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Beijing, China, Nov. 2019, pp. 961–970.
[52] B. Hu, H. Wang, X. Yu, W. Yuan, and T. He, "Sparse network embedding for community detection and sign prediction in signed social networks," *J. Ambient Intell. Hum. Comput.*, vol. 10, no. 1, pp. 175–186, Jan. 2019.
[53] J. Chen, D. Liu, F. Hao, and H. Wang, "Community detection in dynamic signed network: An intimacy evolutionary clustering algorithm," *J. Ambient Intell. Hum. Comput.*, vol. 11, no. 2, pp. 891–900, Feb. 2020.
[54] X. He, H. Du, W. Du, and M. Feldman, "A community structure in fully signed static networks," *Hsi-An Chiao Tung Ta Hsueh/J. Xi'an Jiaotong Univ.*, vol. 52, no. 2, pp. 45–51, 2018.
[55] S. Wang, G. Hu, Z. Pan, J. Zhang, and D. Li, "A game-theoretic approach for community detection in signed networks," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E102.A, no. 6, pp. 796–807, 2019.
[56] Z. Li, J. Chen, Y. Fu, G. Hu, Z. Pan, and L. Zhang, "Community detection based on regularized semi-nonnegative matrix tri-factorization in signed networks," *Mobile Netw. Appl.*, vol. 23, no. 1, pp. 71–79, Feb. 2018.
[57] Y. Zhang, Y. Liu, X. Ma, and J. Song, "Community detection in signed networks by relaxing modularity optimization with orthogonal and nonnegative constraints," *Neural Comput. Appl.*, vol. 2019, pp. 1–10, Nov. 2019.
[58] B. Hu, H. Wang, and Y. Zheng, "Sign prediction and community detection in directed signed networks based on random walk theory," *Int. J. Embedded Syst.*, vol. 11, no. 2, pp. 200–209, 2019.
[59] N. Girdhar and K. K. Bharadwaj, "Community detection in signed social networks using multiobjective genetic algorithm," *J. Assoc. Inf. Sci. Technol.*, vol. 70, no. 8, pp. 788–804, Aug. 2019.
[60] E. Zahedinejad, D. Crawford, C. Adolphs, and J. Oberoi, "Multiple global community detection in signed graphs," in *Proc. 4th Future Technol. Conf.*, San Francisco, CA, USA, 2019, pp. 688–707.
[61] Y. Wu, P. Chao, W. Ying, L. He, and S. Chen, "A conical area evolutionary algorithm based on modularity q for community detection from signed networks," in *Proc. IEEE Int. Conf. Embedded Ubiquitous Comput. (EUC)*, Guangzhou, China, Jul. 2017, pp. 57–62.
[62] Q. Cai, M. Gong, B. Shen, L. Ma, and L. Jiao, "Discrete particle swarm optimization for identifying community structures in signed social networks," *Neural Netw.*, vol. 58, pp. 4–13, Oct. 2014.
[63] Ruby and I. Kaur, "An advanced automated approach for community mining in signed social networks," in *Proc. Int. Conf. Energy, Commun., Data Anal. Soft Comput. (ICECDS)*, Chennai, India, Aug. 2017, pp. 665–670.
[64] Y. Ma, X. Zhu, and Q. Yu, "Clusters detection based leading eigenvector in signed networks," *Phys. A, Stat. Mech. Appl.*, vol. 523, pp. 1263–1275, Jun. 2019.
[65] Z. Liu, Y. Ma, and X. Wang, "A compression-based multi-objective evolutionary algorithm for community detection in social networks," *IEEE Access*, vol. 8, pp. 62137–62150, 2020.

SHIWEI CHE received the M.E. degree from the Department of Computer Science and Technology, Xinjiang University, Xinjiang, China, in 2010. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Technology, Harbin Engineering University. His main research interests include social networks and community detection.

WU YANG received the Ph.D. degree in computer system architecture specialty from the Computer Science and Technology School, Harbin Institute of Technology. He is currently a Professor and a Doctoral Supervisor with Harbin Engineering University. His main research interests include wireless sensor networks, peer-to-peer networks, and information security. He is a member of ACM and a Senior Member of CCF.

WEI WANG received the Ph.D. degree in computer system architecture specialty from the Computer Science and Technology School, Harbin Institute of Technology. He is currently a Professor with Harbin Engineering University. His main research interests include social networks and community detection.