IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 17, 2020, accepted February 2, 2020, date of publication February 6, 2020, date of current version February 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2972123

Multiobjective Complex Network Clustering Based on Dynamical Decomposition Particle Swarm Optimization

TIAOKANG GAO[1,2,3], BIN CAO[1,2,3], (Member, IEEE), AND MENGXUAN ZHANG[2], (Member, IEEE)

[1]School of Artificial Intelligence, Hebei University of Technology, Tianjin 300401, China
[2]Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an 710071, China
[3]Hebei Provincial Key Laboratory of Big Data Calculation, Tianjin 300401, China

Corresponding authors: Tiaokang Gao (201722102013@stu.hebut.edu.cn) and Bin Cao (caobin@scse.hebut.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61976242, in part by the Opening Project of Guangdong Province Key Laboratory of Computational Science at the Sun Yat-sen University under Grant 2018002, in part by the Open Fund of Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, under Grant IPIU2019003, and in part by the State Key Program of National Natural Science of China under Grant 61836009.

ABSTRACT Clustering is a basic tool applied to complex networks. However, the clustering of complex networks is often based on a single objective function, which can obtain insufficient clustering effects. To address the insufficiencies of single objective complex network clustering, multiobjective complex network clustering was proposed. In this article, to improve multiobjective complex network clustering, we prove the superiority of dynamic decomposition mathematically and propose a parallel discrete particle swarm optimization algorithm based on dynamic decomposition (DDDPSO). First, solutions are obtained at different levels by optimizing the objective functions of parallel subpopulations. Second, the decomposition space is divided dynamically by the reference vector of dynamic decomposition. Particle swarms are used to search for optimal solutions in the partitioned dynamic spaces. Finally, the individuals in the particle swarm are sorted according to the obtained solutions to obtain individuals with good convergence and diversity. We conduct comparisons with many state-of-the-art algorithms on many widely used test datasets to test the DDDPSO. The experimental results show the effectiveness of the proposed approach for complex network clustering.

INDEX TERMS Multiobjective optimization, complex network clustering, discrete particle swarm, dynamic decomposition.

I. INTRODUCTION

In recent years, complex network clustering has attracted considerable interest from researchers [1]. Many problems can be abstracted into complex network clustering approaches, such as discovering extended social structures [2], investigating community networks [3], and analyzing protein networks [4]. A complex network can be represented by a graph in which the nodes represent objects in a complex network, and the edges represent the relationships between objects. The purpose of complex network clustering is to assign similar nodes to the same class and different nodes to different classes [5]. A good clustering method makes the nodes in a class dense and the connections between classes sparse. Via complex network clustering, people are better able to study, use, and understand the structures of complex networks.

Recent studies have proposed a large number of complex network clustering theories and techniques [6]–[8]. Among these, complex network clustering based on optimization has become the main research branch [9] by combining optimization with complex network clustering. The distribution of node locations is determined by optimizing the clustering objective. Evolutionary algorithms are widely used in the field of optimization problems [10]–[12]. An evolutionary algorithm is a heuristic search algorithm based on changes in populations that can obtain a solutions set in a single

The associate editor coordinating the review of this manuscript and approving it for publication was Sabu M. Thampi.

execution [13]. Therefore, evolutionary algorithms can be applied to the field of complex network clustering.

Evolutionary algorithms can be divided into single-objective and multiobjective evolutionary algorithms. A genetic algorithm optimizes complex network clustering by selecting an optimal solution using a quality measure function [14]. The clustering result is positively correlated with the value of the A function. For example, the Meme-Net algorithm optimizes clustering with a module density function [15].

As research progressed, people found that clustering objective functions are gradually increasing functions that often contradict each other. Multiobjective complex network clustering can cause the nodes to connect more closely within the class while also making the nodes between classes sparse. Hence, multiobjective evolutionary algorithms have been proposed to solve the complex network clustering problem. In the MOGA-net algorithm, two objective functions are optimized using a multiobjective genetic algorithm to identify networks with dense class connections and sparse connections between classes, and different solutions contain different numbers of classes [16]. A multiobjective optimization framework was used to solve artificial complex network clustering in [17]; the authors proposed a multiobjective evolutionary algorithm to efficiently identify structures in complex networks. To optimize multiple clustering functions, a clustering method based on decomposition of the discrete particle swarm algorithm was proposed in [18] that uses the Tchebycheff approach [19] to transform the multiobjective optimization problem into a scalar solution problem in a discrete particle swarm optimization framework. Label propagation was used to initialize the algorithm.

Many recent studies have also proposed new multiobjective complex network clustering algorithms. MOEA based on local information, termed LMOEA [20], improves the quality of clustering results by selecting the best individual. Mopso-net improves PSO algorithm and optimizes clustering results simultaneously with KKM function and RC function [21]. Zhang *et al.* proposed the idea of reducing the scale of complex networks in the process of multi-objective evolutionary algorithm clustering [22]. The idea Narrows the search space and improves the clustering accuracy.

At present, multiobjective complex network clustering is based on node degree when constructing the objective function, and it ignores the weight between nodes. However, in a real complex network, nodes have not only degrees but also weights. Thus, it is unsuitable to construct the objective function of clustering based solely on node degree the clustering result based only on node degree is not reasonable. The clustering results do not reflect high cohesion and low coupling. To address this problem, this paper proposes a multiobjective complex network clustering model composed of three objective functions that consider both node degree and node weight.

Many types of multiobjective evolutionary algorithms and multiobjective particle swarm optimization (MOPSO) methods exist [23]–[25]. MOPSO has the advantages of fast convergence and maneuverability and outstanding performance in the continuous field. Because of its outstanding performance on continuous problems, scholars began to explore discrete multiobjective particle swarm optimization (DMOPSO) [26]. In this paper, to improve the optimization of multiobjective complex network clustering models, MOPSO based on dynamic decomposition is proposed.

The main contributions of this paper are as follows:

1) To address complex network clustering problems, we construct a multiobjective complex network clustering model by improving the average clustering degree (ACD) function and apply it to complex network clustering. The ACD function complements the signed ratio correlation (SRA) function and the signed ratio cut (SRC) function, and these three functions constitute the multiobjective clustering model. These three objective functions consider not only node degree also node weight. Compared with previous models, the proposed model is more comprehensive, and the objective functions achieve a more reasonable balance.

2) To better optimize multiobjective complex network clustering, a DPSO based on dynamic decomposition is proposed, and the dynamic decomposition strategy of feasibility is proven mathematically. Instead of dividing the solution space of the objective function in advance, the dynamic decomposition algorithm divides the solution space dynamically based on the solution. To balance the convergence and diversity of the algorithm, the individuals are ranked based on their individual performances. This approach uses the top-ranked individuals to optimize the population.

3) To increase the applicability of the algorithm, a parallel method based on objective function and individual population is proposed. This parallel method has two phases. In the first stage, the population is divided into several subpopulations, and each subpopulation is assigned to a CPU. Then, the subpopulations are each optimized on individual computing platforms. The second stage constructs an information pool to store the best objective function value. When a subpopulation needs information, it signals the information pool. When the information pool receives such a request, it sends the objective function value to the requesting subpopulation and updates the new objective function value.

The remainder of this paper is organized as follows. Section II introduces the definition and background of complex networks and clustering objectives. In Section III, the superiority of dynamic decomposition is proved mathematically, and discrete particle swarm optimization based on dynamic decomposition is described. In Section IV, the proposed DDDPSO is compared with other algorithms, verifying the superiority of our algorithm. Finally, Section V summarizes the article and provides a conclusion.

II. BACKGROUND AND RELATED WORK

In this section, firstly, a brief review of complex networks' concepts, classification, and representation are presented; then, we introduce the clustering objectives used in this work,

including the role of each objective. By optimizing these three objective functions, better clustering results can be obtained.

A. COMPLEX NETWORK DEFINITION

Complex networks are divided into signed and unsigned networks by their connecting properties [27]. Unsigned complex networks can be modeled as a graph G = (V,E). In graph, V represents the set of nodes and E represents the set of edges [28]. An A matrix represents the node interconnections. If node i is associated with node j, then $A_{ij} = 1$; otherwise, $A_{ij} = 0$ is obtained. In addition, we use a B matrix that references a similar weight in the unsigned complex network. The elements in B are assigned by the node links. If node i is associated with node j, then $B_{ij} = 1$ is obtained; Otherwise, $B_{ij} = 0$. Complex signed networks can be expressed as G = (V, E,W) [29], where V is the set of vertices, E represents the nodes of edges, and W is the set of weights of nodes. When the weights of nodes i and j are greater than zero, then $A_{ij} = 1$, and when the weight between node i and node j is less than zero, then $A_{ij} = -1$. When no weight exists, then $A_{ij} = 0$. If node i and node j are not connected, then $B_{ij} = 0$; otherwise, B_{ij} is the real weight of the two points. Node degree is calculated by $D(i) = \sum_{j=0}^{n-1} A_{ij}$, while node weight of nodes is calculated by $W(i) = \sum_{j=0}^{n-1} B_{ij}$.

After clustering the complex network, we suppose that there are n classes $\Omega = \{C_1, C_2, , , C_n\}$. In an unsigned complex network, if $\forall i \in C_m$, $K_i^{in} > K_i^{out}$ where $K_i^{in} = \sum_{i \in C_m, j \in C_m} A_{ij}$, $K_i^{out} = \sum_{i \in C_m, j \notin C_m} A_{ij}$ is a strongly connected class; otherwise, it is a weakly connected class. In a signed complex network, C_m is a strongly connected class if it satisfies $\forall i \in C_m$, $(K_i^+)^{in} > (K_i^-)^{in}$, $(K_i^+)^{in} = \sum_{i \in C_m, j \in C_m, C_{ij}=1} A_{ij}$, $(K_i^-)^{in} = \sum_{i \in C_m, j \in C_m, A_{ij}=-1} |A_{ij}|$. If (1) is satisfied, then C_m is a weakly connected class.

$$\begin{cases} \sum_{i \in C_m} (K_i^+)^{in} > \sum_{i \in C_m} (K_i^+)^{out} \\ \sum_{i \in C_m} (K_i^-)^{out} > \sum_{i \in C_m} (K_i^-)^{in} \end{cases} \tag{1}$$

where $(K_i^-)^{out} = \sum_{i \in C_m, j \notin C_m, A_{ij}=-1} |A_{ij}|$, $(K_i^+)^{out} = \sum_{i \in C_m, j \notin C_m, C_{ij}=1} A_{ij}$ [30].

B. CLUSTERING OBJECTIVE FUNCTION OPTIMIZATION

The multiobjective optimization problem (MOP) can be defined as follows:

$$\begin{aligned} &\min f(X) = (f_1(x), f_2(x), , , f_{M-1}(x), f_M(x)) \\ &s.t\ x \in X \end{aligned} \tag{2}$$

where $X \in R^n$ is a decision space, $x \in (x_1, x_2, , , , x_n)$ is a decision vector, and n represents the number of variables. M denotes the number of functions in the multiobjective optimization problem. F(X) is a solution vector composed by an objective function.

Definition 1 (Pareto Domination): In the solution space of the objective function, we will find the solution vector $r = (r_1, r_2, , , r_N)$ and other solution vectors $h = (h_1, h_2, , , h_N)$ when multiple solution vectors satisfy the result conditions. Among these, if

$$\begin{cases} \forall n \in \{1, 2, , , N\} : r_n \le h_n \\ \exists n \in \{1, 2, , , N\} : r_n < h_n \end{cases} \tag{3}$$

then vector r dominates vector h.

Definition 2 (Pareto Set): The elements in the Pareto set must meet Definition 1, which can be expressed as follows:

$$P_S = \{X \in D^n | \nexists X' \in D^n : y(X') \le y(X)\} \tag{4}$$

Definition 3 (Pareto Front): Each solution in the Pareto set corresponds to a vector of the objective function. Therefore, we can obtain the Pareto front (PF).

$$PF = \{y(X) | X \in P_S\} \tag{5}$$

Complex network clustering optimizes the node distribution according to the objective function. For unsigned network clustering, Girvan and Newman [31] proposed the module function Q:

$$Q = \frac{1}{2m}(a_{ij} - \frac{k_i k_j}{2m})\delta(i, j) \tag{6}$$

where m is the sum of the degree of all nodes; a_{ij} is the element in row i, column j of A; and k_i is the degree of node i. If node i and node j are in the same class, then we can obtain $\delta(i, j) = 1$; otherwise, we can obtain $\delta(i, j) = 0$. Gomez *et al.* [32] proposed the corresponding module function SQ for signed complex networks:

$$SQ = \frac{1}{2w^+ + 2w^-} \sum_{i,j} (w_{ij} - (\frac{w_i^+ w_j^+}{2w^+} - \frac{w_i^- w_j^-}{2w^-}))\delta(i, j) \tag{7}$$

where w^+ means the sum of all the nodes whose weights are greater than zero, w^- denotes the sum of all nodes with weights less than zero; and w_{ij} represents the elements in row i and column j of matrix B. The advantage of this function is that the topological structure of the class is unimportant as long as the clustering result satisfies low coupling and high cohesion and the function value is suitable. Intuitively, a higher function value indicates better clustering. Although the performance of complex network clustering is positively correlated with module functions, Fortunato and Barthelemy [33] found that when the class scale is smaller than a certain value, neither a signed module function nor an unsigned module function can effectively guide the clustering of complex networks. Therefore, to solve the problems of evaluation functions so that they are unaffected by the complex network structure as far as possible, to extend the single clustering result and ameliorate one-sided clustering bias, many papers have proposed new schemes.

For example, Li *et al.* [34] proposed the module density function. Lancichinetti *et al.* [35] proposed evaluation functions for complex network clustering. Pizzuti [16] proposed community scoring functions. This paper proposes multiobjective clustering for complex networks.

We abstract the clustering of complex networks into a multiobjective optimization problem with three objectives. The problem can be briefly expressed as follows:

$$\begin{aligned} &\min f(x) = (SRA(C_i), SRC(C_i), ACD(C_i)) \\ &s.t\ C_i \in \Omega \end{aligned} \tag{8}$$

where $\Omega = \{C_1, C_2, , , C_k\}$ means that the complex networks are divided into K classes. The specific function in f(x) is as follows [18]:

$$\begin{cases} SRA = -\sum_{i=1}^{n} \dfrac{L^+(v_i, v_i) - L^-(v_i, v_i)}{|v_i|} \\ SRC = -\sum_{i=1}^{n} \dfrac{L^+(v_i, \bar{v_i}) - L^-(v_i, \bar{v_i})}{|v_i|} \\ ACD = \dfrac{\sum_{j=i}^{SN} CD(i)}{n} \end{cases} \tag{9}$$

where SN represents the total number of nodes. In SRA, the numerator is $L^+(v_i, v_i) = \sum_{i\in C_m, j\in C_m} A_{ij}$, $L^-(v_i, v_i) = \sum_{i\in C_m, j\in C_m, A_{ij}=-1} |A_{ij}|$; and in SRC, the numerator is $L^+(v_i, \bar{v_i}) = \sum_{i\in C_m, j\notin C_m} A_{ij}$, $L^-(v_i, \bar{v_i}) = \sum_{i\in C_m, j\notin C_m, A_{ij}=-1} |A_{ij}|$. The numerator of the CD function [37] is

$$CD(i) = WD(i) \times f(cw(i)) \tag{10}$$

where the f(cw(i)) function is $f(x) = \frac{1}{1+e^{-x}}$; and $CW(i) = \frac{1}{W(i)[D(i)-1]} \sum_{j,k} \frac{w(i,j)+w(j,k)}{2} a_{ij}a_{ij}a_{jk}$ and node i is connected to both nodes j and k, and W(i) is the sum of the weights of node i. The WD(i) function is $WD(i) = \alpha(D(i)) + (1-\alpha)W(i)$, and α is a random number. SRA and SRC are improved by kernel k-means (KKM) and ratio cut (RC), respectively. The reason for selecting two functions is that the KKM function mentioned in [36] increases the number of classes, while the RC function reduces the number of classes. In other words, the KKM function and the RC function are adversarial. The ACD function represents improvements to the CD function [37]. The ACD function balances these two attributes using node degree node weight to determine the average clustering ability of the nodes in the class. The stronger the average clustering ability in a class is, the more nodes that class will gather. However, as the number of nodes increases, the degree of association between the nodes will be reduced. Therefore, ACD is adversarial to the SRA and SRC functions. All three functions are minimized during the clustering process. Minimizing SRA and SRC ensures that the node connections within the class are as dense as possible and that the connections between classes are as sparse as possible. Minimizing ACD by considering weights ensures that the node connections within the class are more robust and reasonable and that the connections between classes are more reliable.

III. DISCRETE DYNAMIC DECOMPOSITION PARTICLE SWARM OPTIMIZATION

In this section, a DPSO based on dynamic decomposition is proposed for multiobjective complex network clustering. First, when dealing with a multiobjective problem, the dynamic decomposition mechanism benefits the algorithm, guiding it more accurately to find the Pareto solution set, and we prove this assertion mathematically. In the second part, we propose the DPSO based on dynamic decomposition and show the optimization of multiobjective complex network clustering in detail.

A. DYNAMIC DECOMPOSITION STRATEGY

Dynamic decomposition of the dynamic partition using reference vectors in the solution space guides the algorithm to search the Pareto solution set of clustering more accurately. To prove its superiority theoretically, first, the principle of dividing the solution space by the reference vector of dynamic decomposition is the same as that used to divide the solution space by the reference vector of angular distance, which works excellently in the field of multiobjective optimization [38]. Second, when the Pareto solution is convex, the dynamically decomposed reference vectors are proven to be more evenly distributed than are the angular distance reference vectors.

The basic theory of dynamic decomposition assumes that—without knowing the shape of the Pareto solution—all the solutions can be pulled towards the hyperplane in the direction of the reference vector. In other words, the reference vectors of the dynamic space pass through the same hyperplane. The dynamic decomposition partition solution space is given a population $P = \{p_1, p_2, p_{3,,,p_n}\}$, where $P_i = \{f_1(x), f_2(x), , , F_M(x)\}$. We normalize each solution as follows:

$$f_i'(x) = \frac{f_i(x) - z_i^{min}}{z_i^{max} - z_i^{min}} \tag{11}$$

where z_i^{max} represents the maximum function value of $f_i(x)$, and z_i^{min} represents the minimum function value. To avoid extreme cases, when $z_i^{max} = z_i^{min}$ we set $f_i'(x) = 1e-10$. The next step is to calculate the reference vector corresponding to each solution as follows:

$$\lambda(x) = \frac{F'(x)}{\sum_{j=1}^{m} f_j'(x)} \tag{12}$$

where $F'(x) = (f_1'(x), f_2'(x), , , f_{m-1}'(x), f_m'(x))^T$ is a standard set of objective value. From Formula (12), we know that

$$1 = \frac{f_1'(x) + f_2'(x) + \cdots + f_{j-1}'(m) + f_j'(m)}{\sum_{j=1}^{m} f_j'(x)}. \tag{13}$$

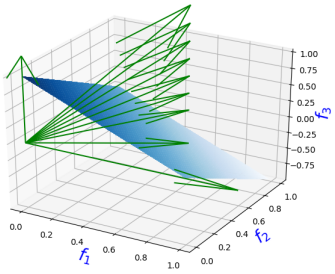


FIGURE 1. The dynamically decomposed reference vector passes through the hyperplane.

And based on Formula (13), the reference vectors of dynamic decomposition pass through a hyperplane. As shown in Figure 1, the green vectors are the reference vectors of dynamic decomposition, and the blue vectors are hyperplanes. The function normalization of angular distance is the same as that of dynamic decomposition; the difference lies is the generated relation of the reference vector. The reference vector of angular distance is as follows:

$$\lambda(x) = \frac{F'(x)}{\sqrt{\sum_{j=1}^{m} f_j'(x)^2}}. \tag{14}$$

where $F'(x) = (f_1'(x), f_2'(x), , , f_{m-1}'(x), f_m'(x))^T$. We sum the squares of $\lambda(x)$ for each element to obtain the following formula:

$$1 = \frac{f_1^{'}(x)^2 + f_2^{'}(x)^2 + \cdots + f_{j-1}^{'}(m)^2 + f_j^{'}(m)^2}{\sum_{j=1}^{m} f_j^{'}(x)^2}. \tag{15}$$

The reference vector for the angular distance also goes through a hypersphere. As shown in Figure 2, the red vectors are the reference vectors of dynamic decomposition, and the blue vectors are hyperplanes. Therefore, the angular distance reference vector and the dynamic decomposition of the reference vector principle are the same.

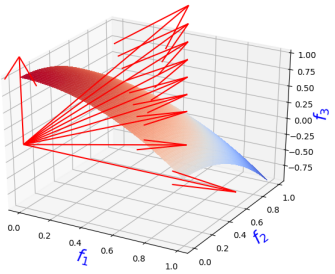


FIGURE 2. An angular distance reference vector passing through the hypersphere.

Furthermore, when the Pareto solution set is convex, the dynamically decomposed reference vectors are more evenly distributed than are the angular distance reference vectors. As shown in Figure 3, the reference vector in yellow is the Pareto solution set across the convex graph, and the hyperplane in blue is that of the angular distance in red.

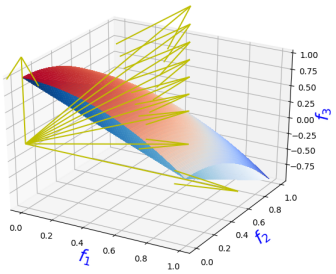


FIGURE 3. The angular distance reference vector and dynamic decomposition reference vector pass through the hyperplane.

As shown in the figure, the dynamic decomposition of the reference vector is more uniform.

It can also be proven that the dynamic decomposition of the reference vector distribution is more uniform when using the geometric method. Given three points $N = (x_1, x_2, x_3, , , x_m)$, $K = (y_1, y_2, y_3, , , y_m)$, $H = (z_1, z_2, z_3, , , z_m)$, on the hyperplane of the dynamic decomposition, assuming that $x_m = t$ and $x_1 = x_2 = x_3, , , = x_{m-1}$, $x_1 = x_2 = x_3, , , = x_{m-1} = \frac{1-t}{m-1}$ can be obtained from Formula (13). According to formula (13), when $y_m = t - u$ and $y_1 = y_2 = y_3, , , = y_{m-1}$, $y_1 = y_2 = y_3, , , = y_{m-1} = \frac{1-t+u}{m-1}$ can be obtained. If $z_m = t - 2u$ and $z_1 = z_2 = z_3, , , = z_{m-1}$, $z_1 = z_2 = z_3, , , = z_{m-1} = \frac{1-t+2u}{m-1}$ can be determined. The distance from point N to point K and the distance from point K to point H can be expressed as dist(N,K) and dist(K,H), which are calculated as follows:

$$\begin{aligned} &dist(N,K) \\ &= \sqrt{(m-1)(\frac{1-t}{m-1} - \frac{1-t+u}{m-1})^2 + (t-(t-u))^2} \\ &dist(N,K) \\ &= \sqrt{(\frac{u}{m-1})^2 + (u)^2} \end{aligned} \tag{16}$$

$$\begin{aligned} &dist(K,H) \\ &= \sqrt{(m-1)(\frac{1-t+u}{m-1} - \frac{1-t+2u}{m-1})^2 + (t-u-(t-2u))^2} \\ &dist(K,H) \\ &= \sqrt{(\frac{u}{m-1})^2 + (u)^2} \end{aligned} \tag{17}$$

Dist(N,K) and dist(K,H) are equal. Considering the vectors $\overrightarrow{ON}$ and $\overrightarrow{OK}$ and $\overrightarrow{OK}$ and $\overrightarrow{OH}$ the cosine of the angle can be expressed as

$$\begin{aligned} &\cos < \overrightarrow{ON}, \overrightarrow{Ok} > \\ &= \frac{(m-1)\frac{1-t}{m-1}\frac{1-t+u}{m-1} + t(t-u)}{\sqrt{(\frac{1-t}{m-1})^2 + t^2}\sqrt{(\frac{1-t+u}{m-1})^2 + (t-u)^2}} \end{aligned} \tag{18}$$

$$\begin{aligned} &\cos < \overrightarrow{OK}, \overrightarrow{OH} > \\ &= \frac{(m-1)\frac{1-t+u}{m-1}\frac{1-t+2u}{m-1} + (t-u)(t-2u)}{\sqrt{(\frac{1-t+u}{m-1})^2 + (t-u)^2}\sqrt{(\frac{1-t+2u}{m-1})^2 + (t-2u)^2}} \end{aligned} \tag{19}$$

From (18) and (19), we know that $\frac{\cos<\overrightarrow{OK},\overrightarrow{OH}>}{\cos<\overrightarrow{ON},\overrightarrow{OK}>} \neq 1$. Based on the mathematical proof above, the reference vector passes through the Pareto solution set of the convex graph, and the angle between ON and OK is not the same as the angle between OK and OH. The distance in the hyperplane of the dynamic decomposition is dist(N,K) = dist(K,H). In summary, when the Pareto solution set is convex, the dynamic decomposition of the reference vector is more uniform than is the angular distance of the reference vector.

B. DPSO BASED ON DYNAMIC DECOMPOSITION

DPSO is widely used to solve optimization problems because of its fast convergence [39], [40]. However, DPSO can easily become trapped in a local optimum. To find the global optimum more accurately, we introduce a dynamic decomposition strategy [41]. Each iteration serves as a foundation that finds some clustering results with good balance between convergence and diversity. Dynamic decomposition references the ideas of quick selection [42]. First, dynamic decomposition divides the initial population into Q and W. Second, in population W, the individuals that divide set W into S_A and S_B are identified. Finally, the individuals with good balance and convergence are found in SA.

1) DYNAMIC DECOMPOSITION DIVIDES DPSO

In DPSO, each individual represents a clustering scheme. Each element in an individual represents a node, and each node has a value range of [1,N], where N represents the total number of nodes. Figure 4 shows the encoding mode of the discrete particle swarm in a complex network. When the integers of two elements are equal, the two nodes belong to the same class.

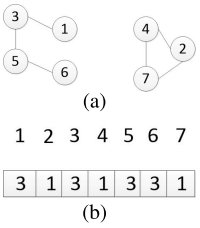


FIGURE 4. The encoding mode of discrete PSO for a complex network.

In DPSO, each particle has a velocity that determines its search range and the time that it takes to find the optimal solution. The particle velocity rule equation is as follows:

$$V_i = sig(\omega V_i + c_1 r_1 (Pbest_i \bigoplus X_i) + c_2 r_2 (Gbest \bigoplus X_i)) \tag{20}$$

where ω represents random numbers in the range [0,1], and c_1 and c_2 represent social and individual cognition, respectively. Both parameters are set to the value of 1.494 [18]. r_1 and r_2 represent two random numbers in the range [0,1]. In (26), the $\bigoplus$ symbol represents a logical XOR operation. The function f(x) = sig(x) is defined as follows:

$$\begin{cases} y_i = 0 & if\ rand(0,1) < sigmoid(x) \\ y_i = 1 & if\ rand(0,1) \geq sigmoid(x) \end{cases} \tag{21}$$

where the sigmoid(x) function can be described as $sigmoid(x) = \frac{1}{1+e^{-x}}$ [18]. Based on the velocity update rules, the velocity and position update rules are as follows:

$$X_i^t = X_i^t \bigotimes V_i^t \tag{22}$$

where $\bigotimes$ influences the position of the velocity at the next moment; therefore, the operator is important. Given $X_2 = (x_{21}, x_{22}, , , , x_{2n})$ and a velocity $v = (v_1, v_2, , , , , v_n)$, and the position $\bigotimes$, the velocity generates a new position. The elements in X_2 are defined as follows:

$$\begin{cases} X_{2i} = X_{2i} & v_i = 0 \\ X_{2i} = Nbest_i & v_i = 1 \end{cases} \tag{23}$$

Suppose the velocity element is equal to zero, and the corresponding position in X_2 does not change. If the position of the velocity is 1, then the corresponding position in X_2 is calculated in terms of $Nbest_i = \arg\max \sum_{j \in Nei} \varphi(X_{2j}, r)$. Nei is the set of neighbors of node j. When i = j, $\varphi(i,j)$ is 1; otherwise, $\varphi(i,j) = 0$. In other words, when the velocity element is 1, the corresponding position is the number that appears most frequently in the neighbor node.

After the particle swarm has completed two evolutions, the population is divided into two parts: one part is the sorted set Q, and the other part is the set W = $\{P-Q\}$, which must be sorted. Dynamic decomposition then iterates to assign rank values to the solution in the set W until all the solutions have been sorted. It is worth noting that when the individual in W is sorted, it will be removed from W and placed in Q. The resulting set in Q can be used to influence the next generation or the final solution. The formula for finding Q is as follows:

$$\begin{cases} e_j = arg\min g(x|w)\ x \in P \\ w_i = 1 \quad i = j \\ w_i = 1e-6 \quad i \neq j \end{cases} \tag{24}$$

where e_j s the population closest to the j-th axis, $w = (w_1, w_2, , , w_m)^T$ is the axis direction, and g is the same aggregation function described previously.

2) DYNAMIC DECOMPOSITION SELECTS INDIVIDUALS FROM THE DIVIDED POPULATIONS

In set W, the algorithm finds an individual that can divide the set into parts S_A and S_B. The formula for dividing W is as follows [41]:

$$\begin{cases} distance(x, Q) = mindst(x, y) \quad y \in Q \\ p = arg\max dist(x, Q) \end{cases} \tag{25}$$

where p is the individual for the division, and the distance formula is [41]

$$dist(x, y) = \|\lambda(x) - \lambda(y)\|_2 \tag{26}$$

Individual P divides set W into S_A and S_B as follows [41]:

$$S_A = \{f(x) \in R^m | dst(x, p) \leq distance(x, Q)\} \quad (27)$$

$$S_B = \{f(x) \in R^m | dst(x, p) > distance(x, Q)\} \quad (28)$$

3) DYNAMIC DECOMPOSITION SELECTS BEST INDIVIDUAL

It can be seen from the formula that S_B is closer to the Q set but that adding the S_B population to Q will be detrimental to the population diversity. Therefore, we consider the S_A population as a candidate population from which we select the best individual in each iteration. The formula to select the best individual s in S_A is as follows [41]:

$$S = arg \min g(x|\lambda(p)) \quad x \in S_A \quad (29)$$

where Lambda (p) is the reference vector for individual p, and the function g is an aggregate function. The individual in S_A is multiplied by the same reference vector, which ensures that the selected optimal individual will have good convergence. At the same time, diverse individuals are selected from the S_A population; thus, they have both good convergence and diversity.

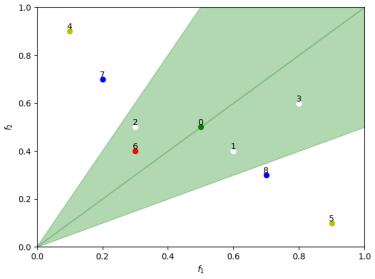


FIGURE 5. The dynamic decomposition selection process.

To clarify the dynamic decomposition optimization DPSO, we provide an example. Figure 5 shows a schematic diagram of the dynamic decomposition selection process of a bi-objective function in which the number of populations is 9, and each individual is assigned a unique number. According to Formula (20), we first divide P into S_A and S_B. The yellow individuals (4 and 5) represent the individuals in $Q = \{4, 5\}$. The remaining individuals are selected from the set $W = \{0, 1, 2, 3, 6, 7, 8\}$. First, we select p individuals using (21) and (22), which are the individuals marked in green. Next, using Formulas (22) and (23), we divide W into set $S_A = \{0, 1, 2, 3, 6\}$ and set $S_B = \{7, 8\}$. The area occupied by the points in S_A are marked in light green, and the individuals in S_B are marked in blue. Then, in the solution space of S_A, according to Formula (25), we can conclude that the red individual 6 is the closest to the green individual 0. Therefore, number six is selected as the best individual. We repeat the above process until the number of selected individuals meets the requirements.

It is worth noting that in the last iteration, the final non-dominated set is selected in an unsupervised way. First, one individual selected by Formulas (25)-(28) is divided into S_A and S_B. Then, the real clustering results are selected from S_A by Formula (29).

C. PARALLELIZATION METHOD BASED ON OBJECTIVE FUNCTION AND POPULATION

As the number of objective functions and the number of nodes in the network increase, the running time of the algorithm also increases. Therefore, to improve the computational efficiency, we propose a parallel method based on the objective function and population consisting of three parts. First, within a neighborhood, individuals communicate with each other. Second, individuals receive neighborhood information and use their own computing resources to update the individual state. Finally, the optimal objective function of the individual information is stored to update the subpopulation. When a subpopulation needs the optimal objective function, the program sends that information to the subpopulation.

Assuming that there are N CPUs to execute parallel processes and NP individuals, the number of populations assigned to each CPU is $\frac{NP}{N}$. Thus, each CPU is responsible for complete sets of individuals. The MOP can be optimized by all the CPUs simultaneously, which greatly reduces the running time. The CPU exchanges information in order according to the physical structure of the computer where the distribution of CPU nodes is abstracted into a network. In this network, each crossover node represents a CPU. To identify the CPU neighbor node, the CPU step size should be calculated to determine which CPU should send information to the current CPU and which CPU should receive current CPU information. The calculation formula for the step size and neighbor nodes is as follows:

$$\begin{cases} step = \sqrt{N} \\ Node_{itop} = (node_i - step + N)\%N \\ Node_{ibottom} = (node_i + step)\%N \\ Node_{ileft} = (node_i - 1 + N)\%N \\ Node_{iright} = (node_i + 1 + N)\%N \end{cases} \quad (30)$$

where step is the step length, $Node_i$ is the rank of MPI process i, and $Node_{iup}$,$Node_{idown}$, $Node_{idown}$ and $Node_{idown}$ are the ranks of the four neighboring nodes (top, bottom, left and right).

The running time of the algorithm primarily involves the population evolution and the objective function calculations; therefore, the time consumption of the serial algorithm is

$$T_1 \approx t_1 + t_2 \quad (31)$$

where t1 is the time evolution of the population and t2 is the time required to calculate the objective function. When the algorithm is combined, the time consumption formula becomes

$$T_2 \approx \frac{t_1}{N} + \frac{t_2}{N} \quad (32)$$

where N is the number of CPUs. Consequently, the parallel form of the algorithm is approximately N times more efficient than is the serialized form.

D. ALGORITHM FRAMEWORK

The overall framework of the DDDPSO is shown in Table 1. In step 3.4, DDDPSO applies the mutation operator [18] whose algorithm framework is shown in Table 2. After two DPSO iterations, dynamic decomposition is used to guide the DPSO to select individuals with both convergence and diversity as the parent of the next generation.

TABLE 1. Algorithm framework.

DDDPSO algorithm framework based on adaptive grouping
Parameter Settings: maximum iteration number Maxgen, population size, population P, mutation probability probability=0.1, select the results New_p, weight w in particle swarm and learning factors, c1 and c2 input A matrix B and A output real clustering results, P_{best}, g_{best}
Step 1): Initialization
Step 1.1): Initialize the position of each particle $P = \{x_1, x_2, , , x_{pop}\}^T$
Step 1.2): Initialize the velocity of each particle $V = \{v_1, v_2, , , v_{pop}\}^T$
Step 2): Set t = 0
Step 3): while t < Maxgen do
Step 3.1): Calculate the velocity and of position the i-th particle
Step 3.2): If rand(0,1)>Pm, then the i-th particle performs mutation according to algorithm 2
Step 3.3): If func(P_i)<f(P_{best}), then P_i=P_{best}
Step 3.4): Store the current population P=P+P
Step 4): If (t%2==0)
Step 4.1): Using Eq. (24), divide the population from two iterations into W and Q
Step 4.2):for i=1,2,,,pop, do
Step 4.3): Using Eq. (25) and Eq. (28), find individual that divides W into S_A and S_B
Step 4.4):Use (29) to select s and add s to New_p
step 4.5):If i==1, then g_{best}=s
step 4.6):Update W=W-s
step 4.7):end for
step 4.8):Update $P = New_p$
step 5): If t==Maxgen-1 by Formulas (25)–(29) select the real clustering results
Step 6):t++
Step 7):end while

TABLE 2. Individual mutation algorithm.

Algorithm 2: Individual mutation algorithm
nodesize is the number of neighbors
1: for i=1;i<nodesize;i++
2: if rand(0,1)< probability
3: for k=1;j≤Node[i].neighsize;k++
4: m=Node[i].neig[j];
5: X[m]=X[i];
6: end for
7: end for

E. COMPLEXITY ANALYSIS

1) The storage space of the algorithm is mainly composed of two parts.The first part is used to store experimental data.The second part is used to store individuals in a population.The space needed to store data is O(n^2). N is node of number. The space needed to store individual is O(mn), where m is size of the population. Therefore, the total storage space required by the algorithm is O(n^2).

2) Time Complexity: Step 3 and step 4 take up most of the calculation time of the algorithm because the other steps can be ignored. In the process of analyzing the time complexity of the algorithm, the number of nodes in the complex network is represented by n, and the degree is represented by m. Steps 3.4, 3.5, 4.5, 4.6, 4.7 and 4.8 require O(1) basic operations. Steps 3.1, 3.2, 4.1, 4.2, 4.4 and 5 of time need for execution is O(n), and step 3.3 of tome need O(K^2). K is the average degree of the network. Finally, step 4.3 requires O(2n) basic operations. According to the abstract rules of time complexity, The worst-case calculation time for DDDPSO is O($iterations \cdot population \cdot (m+n)$). In the time complexity of DDDPSO, iterations is the number of iteration and population is the number of individuals in a population.

IV. EXPERIMENTAL STUDIES

A. COMPARED ALGORITHMS

We compare our algorithm with seven similar algorithms: GA-net [14], MOEA/D-net [43], MOCD [17], MOGA-net [16], Informap [44], MOPSO-r2 [45], LMOEA [20] and CCLPA [46]. proposed the GA-net algorithm. The main purpose of GA-net is to apply EA to network clustering. Pizzuti defined a community score to evaluate the clustering results. MOCD, MOGA-net, and MOEA/D-net are network clustering algorithms based on MOEA. To compare the derived algorithms based on MOEA and the derived algorithm based on DPSO, we chose these three algorithms and our proposed DRVPSO for comparison. Decomposition strategies are used in our algorithm and these three algorithms. The biggest difference between MOCD and our algorithm lies in the optimization mechanism. MOCD uses a genetic algorithm for optimization, while our algorithm uses a reference vector. Informap is a complex network clustering algorithm based on informatics. MOPSO-r2 is an extension of MOPSO that uses heuristic search methods. CCLPA uses a propagation algorithm to cluster nodes based on clustering parameters.

B. EXPERIMENTAL SETTINGS

The parallel technique used in the algorithm is MPI. MPI is a parallel communication scheme based on the C++ framework. The computer used in the experiments was equipped with 2 Intel e5-2665 CPUs running at 2.4 GHz with 128 GB of memory: each CPU has 8 cores. The development environment of the experiment was Visual Studio 2017 under a Windows 10 operating system, and the language used in the experiment was C++. The program code is available upon email request.

When the real clustering results are known, the measure we use is normalized mutual information (NMI) [47]. The NMI effectively measures the similarity between the clustering results of the algorithms and the true clustering. Suppose the clustering result of an algorithm is set A but the real clustering result is set B. We define a fuzzy-proof C such that the elements in matrix C are the number of nodes jointly owned by class i in set A and class j in set B. The NMI(A,B) is then defined as follows [47]:

$$NMI = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} C_{ij} log(C_{ij}N/C_{i\cdot}C_{\cdot j})}{\sum_{i=1}^{C_A} C_{i\cdot} log(C_{i\cdot}) + \sum_{i=1}^{C_B} C_{\cdot j} log(C_{\cdot j})} \quad (33)$$

where C_A and C_B are the number of classes in sets A and B, N is the number of nodes, and $C_{i\cdot}$ and $C_{\cdot j}$ are the sums of the elements of row i and column j, respectively. The value

TABLE 3. Average rankings of the algorithms on the GN dataset(Aligned Friedman).

Algorithm	Ranking
GA-net	38.1818
MOCD	16.6364
MOGA-net	31.2727
MOPSO-r2	46.8182
CCPLA	55.3636
DDDPSO	12.7273

range of NMI is [0, 1]. When NMI is zero, A and B are completely different, and when NMI is equal to 1, A and B are identical. Danon *et al.* [48] previously proved the feasibility and rationality of NMI.

1) GN NETWORK EXPANSION EXPERIMENT

The GN benchmark network [31] consists of 128 nodes divided into 4 categories. The average degree of nodes is 16. In the GN benchmark network, r represents the proportion of nodes connected to nodes that are not in the same class, and 1-r represents the proportion of nodes connected to nodes within the same class. When r<0.5, we say that the network structure is strong and vice versa. In the experiment, the *r* value ranged from 0 to 0.5.

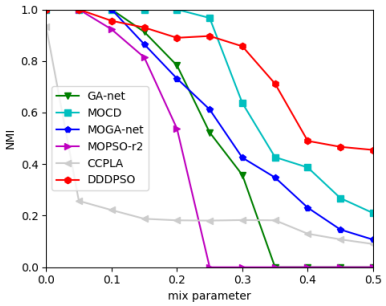


FIGURE 6. On the GN extended base dataset, the results of different algorithms have an average of 30 NMI values.

In Figure 6, the abscissa shows r increasing from 0 to 0.5 with a step size of 0.05, while the ordinate represents the average value of NMI obtained by each algorithm over 30 runs. From Figure 6, before r reaches 0.2, DDDPSO is not significantly different from the other algorithms. Between 0.2 and 0.3, DDDPSO moves toward second place. After 03, DDDPSO clearly begins to show its superiority. From 03 to 0.5, DDDPV ranks first. This shows that DDDPSO is more effective on networks with complex structures. From the perspective of the overall curve, our algorithm is more promising than are the other algorithms at clustering networks that possess more complex node structures. To convincingly demonstrate DDDSO's significant advantages over the other algorithms, we calculated the aligned Friedman score of all the algorithms. As shown in Table 3, DDDPSO achieves first place. In addition, we calculated the variance of the NMI obtained by the algorithms running on the GN dataset. As shown in Table 6, our algorithm is generally stable.

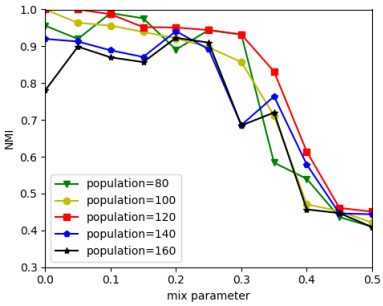


FIGURE 7. On the GN extended base dataset, the results of different population num have an average of 30 NMI values.

TABLE 4. Average rankings of the population (aligned Friedman).

Population	Ranking
80	26.4545
100	26.9545
120	11.1364
140	33.2727
160	42.1818

The population number has a large influence on the outcome of the algorithm; therefore, we performed some experiments. Figure 7 shows representations of different populations on the datasets. We varied the population number from 80 to 160 and observed which population performed well on the datasets. The population increases by 20 individuals at a time. Between 0 and 0.2, the results (the red line) does not clearly show advantages over the other cases. However, when r>0.2, the red line indicates a substantially superior performance over other cases. The red line represents a population of 120. Populations 140 and 160 also performed well in the 0.45 and 0.5 datasets. However, according to the Friedman population in Table 4, we fixed the population at 120.

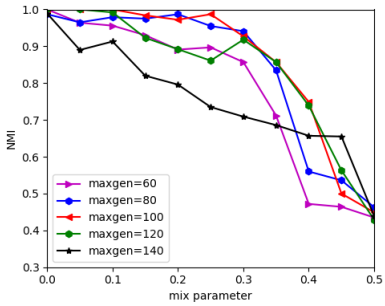


FIGURE 8. On the GN extended base dataset, the results of different maxgen as an average of 30 NMI values.

Apart from the population parameter, we also explored the effect of the number of iterations on clustering by fixing the other parameters while varying the number of iterations. The iteration parameter was varied from 60 to 140 at a step size of 20. Figure 8 shows the NMI obtained by the different numbers of iterations: when the number of iterations exceeds a certain value, the NMI decreases. An iteration number between 60 and 100 increases the value of NMI; however, after the number of iterations reaches 100, the NMI value

TABLE 5. Average rankings of the maxgen (aligned Friedman).

Maxgen	Ranking
60	34.3182
80	32.0909
100	14.1818
120	20.9545
140	38.4545

TABLE 6. Variance of the algorithms on GN datasets.

Algorithm	variance
GA-net	0.2020
MOCD	0.1120
MOGA-net	0.0578
MOPSO-r2	0.2142
CCPLA	0.0548
DDDPSO	0.0473

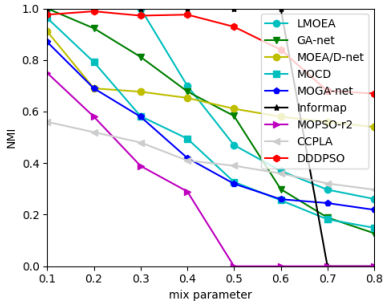


FIGURE 9. On the LFR extended base dataset, the results of different algorithms obtain an average of 30 NMI values.

decreases. According to Table 5, based on the maxgen of a Friedman test, we fixed the number of iterations at 100.

2) LFR COMPLEX NETWORK

The degree of each node in the GN-based network is similar. The number of nodes in each class is approximately the same, and the total number of nodes is relatively small. These properties do not reflect complex networks in real life. Lancichinetti proposed that the number of LFR data aggregation points should be 1000. The LFR dataset belongs to a complex large-scale network. In the dataset, the parameters μ_1 and μ_2 should be set to control the node degree and the number of nodes in the class after clustering [49]. A node's connections to nodes in its own class occupies a ratio of μ to all the connections for that node, while its connections to nodes in another class constitute the ratio 1-μ to all connections. The range of μ is $[0, 1]$. We varied μ from 0.1 to 0.8 with a step size of 0.1. After the LFR data aggregation class, the number of classes varied from 10 to 50. We set the parameters $\mu_1 = 2$ and $\mu_2 = 1$ so that the average degree of all the nodes was 20 and the maximum node degree was 50. The experimental results of 30 runs are shown in Figure 9. On the LFR data, the population and iteration parameters of DDDPSO are 120 and 100, respectively.

As shown by the curves in Figure 9, our algorithm performs stably across different datasets and achieves excellent clustering. Between 0 and 0.5, our algorithm consistently produces NMI values above 0.9. In other words, even when applied to large-scale complex network clustering, DDDPSO can accurately identify the locations of nodes and obtain good clustering results. Between dataset values of 0.4 and 0.6, our algorithm performed consistently and ranked second. Although DDDPSO fluctuates somewhat between 0.5 and 0.6, it still ranks second and is well ahead of third place and well behind first place. Surprisingly, after 0.6, the performance of Informap changed substantially: its NMI value suddenly drops from 1 to 0. Thus, DDPSO, after some fluctuations, moved from second-place to first place. From the datasets with values of 0.6 to 0.8, the algorithm structure becomes very complex, and the number of nodes becomes very large. Nevertheless, our algorithm still performs well, indicating the universality of DDDPSO. Furthermore, according to the Friedman test results in Table 7, our algorithm has promising potential for addressing large-scale complex networks.

TABLE 7. Average rankings of the algorithms (aligned Friedman).

Algorithm	Ranking
GA-net	43.625
MOCD	45.25
MOGA-net	43.125
MOPSO-r2	65.375
CCPLA	48.875
DDDPSO	9.5
MOEA/D-net	24.625
Informap	22.9375
LMOEA	25.1875

3) REAL-WORLD DATASET

We also applied algorithms to an American College Football dataset [31]. This dataset contains a total of 115 nodes and 613 edges representing the Iowa college football grouping in the fall of 2000. The dataset constitutes 12 subclasses after clustering. Due to the complex connections and the unbalanced classification of nodes in this dataset, no algorithm can comprehensively find the correct structure. On the American college football data, the population and iteration parameters of DDDPSO were set to 120 and 100, respectively. The NMI data of each algorithm for this dataset are shown in Table 8, which shows that DDDPSO performs better than the other algorithms.

The dolphin network includes a network of 62 bottlenose dolphins. According to the gender, the network naturally is separated into the female group and the male one. The NMI data of each algorithm for this dataset are shown in Table 9, which shows that DDDPSO performs better than the other algorithms.

C. THE DIVERSITY AND CONVERGENCE OF DDDPSO

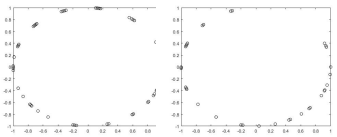
Algorithm diversity and convergence are two important performance indicators for multiobjective complex network clustering algorithms. To improve the convergence and diversity of DPSO, we proposed creating a DPSO based on dynamic decomposition. To evaluate the influence on the clustering effect achieved by dynamic decomposition, we visualized and quantified the clustering solution set of DPSO both with and without dynamic decomposition.

TABLE 8. NMI values of different algorithms on the American college football datasets.

algorithm	GA-net	MOEA/D-net	MOCD	MOGA-net	Informap	DDDVPSO
NMI	0.906	0.861	0.804	0.840	0.762	0.913

TABLE 9. NMI values of different algorithms on the dolphin datasets.

algorithm	GN	CNM	Informap	Mopso-r2	DDDVPSO
NMI	0.4304	0.554	0.5730	0.5622	0.6243



(a) score=0.8622 (b) score=0.5524

FIGURE 10. The encoding mode of discrete PSO for a complex network.

As shown in Figure 10, the solution sets obtained by DPSO with dynamic decomposition are close to circular in shape. The more the solution set of a the multiobjective algorithm tends toward circularity, the better the convergence and balance of the algorithm are. Clearly, the DPSO-based dynamic decomposition algorithm is better. To form a more convincing demonstration, the complex network clustering results obtained by DDDPSO are not only visualized but also quantified. As shown in Figure 10, the DDDPSO scores 0.8622, while DPSO scores 0.5524 (the score range of this value is [0, 1], and the closer the value is to 1, the better the convergence and balance of the algorithm are). DDDPSO with dynamic decomposition performs better than does DPSO without dynamic decomposition. There are two main reasons why DDDPSO is better. First, the reference vector used in dynamic decomposition divides the solution space more evenly. The particle swarm perturbations in these subspaces yield clustering results closer to the Pareto front. Second, based on their convergence and diversity, the dynamic decomposition sorts the individuals in the population. The ranking results directly affect the survival of individuals.

V. CONCLUSION

In this paper, we proposed using ACD objective functions in combination with SRC and SRA functions to form a multiobjective complex network clustering method. The ACD function considers both the weight and degree of nodes, resulting in more reasonable clustering results. Multiobjective complex network clustering is executed to obtain several different solutions, each of which has a different emphasis. To better solve multiobjective complex network clustering problems, DPSO has been widely used for clustering. DPSO has the advantages of simple operation and rapid convergence; however, is also has the disadvantage of becoming trapped in local optima. To balance the diversity and convergence of algorithms, we proposed a DPSO variant based on dynamic decomposition. We first proved that the principle of dividing the solution space by dynamic decomposition is the same as that of dividing the solution space by distance. The, on that basis, we proved mathematically that when the square Pareto solution set is convex, the dynamically decomposed reference vectors are more evenly distributed than are the angular distance reference vectors. To increase the universality of the algorithm, we proposed a parallelization method centered on the population and the objective functions. This parallel method calculates the top, bottom, left and right neighbors according to von Neumann's system.

REFERENCES

[1] H. Su, Z. Rong, M. Z. Q. Chen, X. Wang, G. Chen, and H. Wang, "Decentralized adaptive pinning control for cluster synchronization of complex dynamical networks," *IEEE Trans. Cybern.*, vol. 43, no. 1, pp. 394–399, Feb. 2013.

[2] H. Meyerhenke, P. Sanders, and C. Schulz, "Parallel graph partitioning for complex networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 9, pp. 2625–2638, Sep. 2017.

[3] N. Tremblay and P. Borgnat, "Graph wavelets for multiscale community mining," *IEEE Trans. Signal Process.*, vol. 62, no. 20, pp. 5227–5239, Oct. 2014.

[4] L. Ma, J. Li, Q. Lin, M. Gong, C. A. C. Coello, and Z. Ming, "Reliable link inference for network data with community structures," *IEEE Trans. Cybern.*, vol. 49, no. 9, pp. 3347–3361, Sep. 2019.

[5] C. Gao, M. Liang, X. Li, Z. Zhang, Z. Wang, and Z. Zhou, "Network community detection based on the physarum-inspired computational framework," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 15, no. 6, pp. 1916–1928, Nov. 2018.

[6] S. Qiao, N. Han, Y. Gao, R.-H. Li, J. Huang, J. Guo, L. A. Gutierrez, and X. Wu, "A fast parallel community discovery model on complex networks through approximate optimization," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1638–1651, Sep. 2018.

[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. Comput. Soc. Syst.*, vol. 5, no. 4, pp. 1021–1033, Dec. 2018.

[8] M. Feng, H. Qu, and Z. Yi, "Highest degree likelihood search algorithm using a state transition matrix for complex networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 61, no. 10, pp. 2941–2950, Oct. 2014.

[9] L. Bai, X. Cheng, J. Liang, and H. Shen, "An optimization model for clustering categorical data streams with drifting concepts," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 11, pp. 2871–2883, Nov. 2016.

[10] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 5, pp. 773–791, Oct. 2016.

[11] Q. Kang, S. Feng, M. Zhou, A. C. Ammari, and K. Sedraoui, "Optimal load scheduling of plug-in hybrid electric vehicles via weight-aggregation multi-objective evolutionary algorithms," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2557–2568, Sep. 2017.

[12] B. Xu, Y. Zhang, D. Gong, Y. Guo, and M. Rong, "Environment sensitivity-based cooperative co-evolutionary algorithms for dynamic multi-objective optimization," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 15, no. 6, pp. 1877–1890, Nov. 2018.

[13] F. Lin, J. Zeng, J. Xiahou, B. Wang, W. Zeng, and H. Lv, "Multiobjective evolutionary algorithm based on nondominated sorting and bidirectional local search for big data," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 1979–1988, Aug. 2017.

[14] C. Pizzuti, "GA-Net: A genetic algorithm for community detection insocial networks," in *Proc. Int. Conf. Parallel Problem Solving Nature*, vol. 5199, 2008, pp. 1081–1090.

[15] M. Gong, B. Fu, L. Jiao, and H. Du, "Memetic algorithm for communitydetection in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 84, no. 5, Nov. 2011, Art. no. 056101.

[16] 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.
[17] C. Shi, Z. Y. Yan, Y. N. Cai, and B. Wu, "Multi-objective community detection in the complex networks," *Appl. Soft Comput.*, vol. 12, no. 2, pp. 850–859, Feb. 2012.
[18] 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.
[19] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.
[20] F. Cheng, T. Cui, Y. Su, Y. Niu, and X. Zhang, "A local information based multi-objective evolutionary algorithm for community detection in complex networks," *Appl. Soft Comput.*, vol. 69, pp. 357–367, Aug. 2018.
[21] S. Rahimi, A. Abdollahpouri, and P. Moradi, "A multi-objective particle swarm optimization algorithm for community detection in complex networks," *Swarm Evol. Comput.*, vol. 39, pp. 297–309, Apr. 2018.
[22] X. Zhang, K. Zhou, H. Pan, L. Zhang, X. Zeng, and Y. Jin, "A network reduction-based multiobjective evolutionary algorithm for community detection in large-scale complex networks," *IEEE Trans. Cybern.*, vol. 50, no. 2, pp. 703–716, Feb. 2020.
[23] H. Han, W. Lu, L. Zhang, and J. Qiao, "Adaptive gradient multiobjective particle swarm optimization," *IEEE Trans. Cybern.*, vol. 48, no. 11, pp. 3067–3079, Nov. 2018.
[24] Q. Zhu, Q. Lin, W. Chen, K.-C. Wong, C. A. C. Coello, J. Li, J. Chen, and J. Zhang, "An external archive-guided multiobjective particle swarm optimization algorithm," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2794–2808, Sep. 2017, doi: 10.1109/tcyb.2017.2710133.
[25] X. Yu, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, S. Kwong, and J. Zhang, "Set-based discrete particle swarm optimization based on decomposition for permutation-based multiobjective combinatorial optimization problems," *IEEE Trans. Cybern.*, vol. 48, no. 7, pp. 2139–2153, Jul. 2018.
[26] L. Tong, B. Du, R. Liu, and L. Zhang, "An improved multiobjective discrete particle swarm optimization for hyperspectral endmember extraction," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 10, pp. 7872–7882, Oct. 2019.
[27] 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.
[28] X. Zheng, Y. Y. Tang, and J. Zhou, "A framework of adaptive multiscale wavelet decomposition for signals on undirected graphs," *IEEE Trans. Signal Process.*, vol. 67, no. 7, pp. 1696–1711, Apr. 2019.
[29] C. J. Quinn, N. Kiyavash, and T. P. Coleman, "Directed information graphs," *IEEE Trans. Inf. Theory*, vol. 61, no. 12, pp. 6887–6909, Dec. 2015.
[30] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Dening and identifying communities in networks," in *Proc. Nat. Acad. Sci. USA*, vol. 101, no. 9, pp. 2658–2663, Mar. 2004.
[31] 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.
[32] S. Gomez, 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, Jul. 2009, Art. no. 016114.
[33] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," *Proc. Nat. Acad. Sci. USA*, vol. 104, no. 1, pp. 36–41, 2007.
[34] Z. Li, S. Zhang, R. Wang, X. Zhang, and L. Chen, "Quantitative function for community detection," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 77, no. 3, 2008, Art. no. 036109.
[35] A. Lancichinetti, S. Fortunato, and K. Kertesz, "Detecting the overlapping and hierarchical community structure in the complex networks," *New J. Phys.*, vol. 11, no. 3, 2009, Art. no. 033015.
[36] L. Angelini, S. Boccaletti, D. Marinazzo, M. Pellicoro, and S. Stramaglia, "Identification of network modules by optimization of ratio association," *Chaos*, vol. 17, no. 2, Jun. 2007, Art. no. 023114.
[37] Q. Wang, J. Ren, Y. Wang, B. Zhang, Y. Cheng, and X. Zhao, "CDA: A clustering degree based influential spreader identification algorithm in weighted complex network," *IEEE Access*, vol. 6, pp. 19550–19559, 2018.
[38] X. Bi and C. Wang, "A many-objective evolutionary algorithm based on hyperplane projection and penalty distance selection," *Natural Comput.*, vol. 17, no. 4, pp. 877–899, Dec. 2018.
[39] J. Huang, M. Gong, and L. Ma, "A global network alignment method using discrete particle swarm optimization," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 15, no. 3, pp. 705–718, May 2018.
[40] J. Smoczek and J. Szpytko, "Particle swarm optimization-based multivariable generalized predictive control for an overhead crane," *IEEE/ASME Trans. Mechatronics*, vol. 22, no. 1, pp. 258–268, Feb. 2017.
[41] X. He, Y. Zhou, Z. Chen, and Q. Zhang, "Evolutionary many-objective optimization based on dynamical decomposition," *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 361–375, Jun. 2019.
[42] C. A. R. Hoare, "Algorithm 65: Find," *Commun. ACM*, vol. 4, no. 7, pp. 321–322, Jul. 1961.
[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, Aug. 2012.
[44] 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, Jan. 2008.
[45] G. Palermo, C. Silvano, and V. Zaccaria, "Discrete particle swarm optimization for multi-objective design space exploration," in *Proc. 11th EUROMICRO Conf. Digit. Syst. Design Archit. Methods Tools*, 2008, pp. 641–644.
[46] E. Raju, Y. R. Devi, and K. Sravanthi, "CCLPA: A clustering coefficient based label propagation algorithm for unfolding communities in the complex networks," in *Proc. 2nd Int. Conf. Commun. Electron. Syst. (ICCES)*, Coimbatore, India, 2017, pp. 240–245.
[47] F. Wu and B. A. Huberman, "Finding communities in linear time: A physics approach," *Eur. Phys. J. B-Condensed Matter*, vol. 38, no. 2, pp. 331–338, Mar. 2004.
[48] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identifcation," *J. Stat. Phys.*, vol. 2005, no. 9, 2005, Art. no. P09008.
[49] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 78, no. 4, 2008, Art. no. 046110.

TIAOKANG GAO received the bachelor's degree in computer science and technology from Hebei Agricultural University, Hebei, China, in 2017. He is currently pursuing the master's degree in computer science and technology with the School of Artificial Intelligence, Hebei University of Technology, Tianjin, China. His research interests multiobjective evolution and complex network clustering.

BIN CAO (Member, IEEE) received the Ph.D. degree in computer application technology from Jilin University, in 2012. From 2012 to 2014, he was a Postdoctoral Researcher with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He is currently a Professor with the Hebei University of Technology, Tianjin, China. His research interests include intelligent computation with its applications to cyber-physical systems, big data, graphics and visual media, high-performance computing, and cloud computing.

MENGXUAN ZHANG (Member, IEEE) received the B.S. and Ph.D. degrees in circuits and systems from Xidian University, Xi'an, China, in 2010 and 2017, respectively. She is currently a Lecturer with the School of Artificial Intelligence, Xidian University. Her research interests include computational intelligence, data mining, and image processing.