IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received February 24, 2021, accepted March 9, 2021, date of publication March 10, 2021, date of current version March 18, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3065532

An Improved Particle Swarm Optimization Algorithm for Bayesian Network Structure Learning via Local Information Constraint

KUN LIU, YANI CUI, JIA REN, AND PEIRAN LI
School of Information and Communication Engineering, Hainan University, Haikou 570228, China

Corresponding author: Yani Cui (cyn0213@163.com)

This work was supported in part by the Hainan Natural Science Foundation Innovation Research Team Project under Grant 620CXTD434, in part by the Hainan Provincial Natural Science Foundation of China under Grant 519QN180, in part by the National Natural Science Foundation of China and Macau Science and Technology Development Joint Fund under Grant 61961160706 and Grant 0066/2019/AFJ, and in part by the Scientific Research Foundation of Hainan University under Grant KYQD(ZR)1859.

ABSTRACT At present, in the application of Bayesian network (BN) structure learning algorithm for structure learning, the network scale increases with the increase of number of nodes, resulting in a large scale of structure search space, which is difficult to calculate, and the existing learning algorithms are inefficient, making BN structure learning difficulty increase. To solve this problem, a BN structure optimization method based on local information is proposed. Firstly, it proposes to construct an initial network framework with local information and uses the Max-Min Parents and Children (MMPC) algorithm to construct an undirected network framework to reduce the search space. Then the particle swarm optimization (PSO) algorithm is used to strengthen the algorithm's optimization ability by constructing a new position and velocity update rule and improve the efficiency of the algorithm. Experimental results show that under the same sample data set, the algorithm can obtain a more accurate BN structure while converging quickly, which verifies the correctness and effectiveness of the algorithm.

INDEX TERMS Bayesian network, structure learning, local information, particle swarm optimization algorithm.

I. INTRODUCTION

Bayesian network (BN) is an important tool for uncertain knowledge expression, and it is a directed acyclic graph (DAG) of joint probability distribution of some nodes [1]. It consists of two parts: DAG and conditional probability table (CPT). Among them, the DAG qualitatively represents the independent relationship between variables, and the CPT quantitatively represents the degree of dependence between variables. Due to their powerful abilities in representing and reasoning, it is widely used in the field of biomedicine [2], prediction [3], classification [4], causal inference [5], etc.

The approaches of BN structure learning can be divided into three types: constraint-based methods [6], search-and-score methods [7] and hybrid methods [8], [9]. Since finding the optimal BN is a non-deterministic polynomial (NP) problem [10], In general, the number of network structures in DAG space is large, and the search space of the network structure will increase exponentially with the number of network nodes. In order to reduce the complexity of search, find the optimal network structure, the BN structure learning method mainly uses the heuristic search method.

At present, some scholars have proposed the use of ant colony algorithm [11], genetic algorithm [12], artificial bee colony algorithm [13], simulated annealing algorithm [14], PSO [15] and other algorithms for BN structure learning. PSO algorithm has been widely used in the learning process of BN structure because of its simple coding, strong global search ability and fast convergence speed.

At the same time, PSO is also applied to the following methods. Reference [16] showed a new multi-swarm cooperative multi-objective PSO algorithm to solve the multi-objective optimization problems. Reference [17] proposed a new MOPSO algorithm and applied successfully to solve the multi-objective environmental/economic dispatch problem with constraints. Similarly, PSO has certain advantages when dealing with high-dimensional data. Reference [18] proposes a new feature selection method

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

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

based on variable-size cooperative coevolutionary PSO, which divides the high-dimensional feature selections problem into several low-dimensional sub-problems. Reducing the calculation cost greatly improves the efficiency of the algorithm. Reference [19] showed the first study of multi-objective PSO for cost-based feature selection. It is a very competitive feature selection method to solve the cost-based feature selection problem.

However, in the process of using PSO to learn BN structure, most of the literature ignores the construction of node information for the initial structure when using PSO for structure learning. Reference [20] designed a discrete PSO to solve the learning problem of BN structure. Use mutation operator and nearest neighbor search operator to overcome the shortcomings of algorithm premature. Reference [21] proposed a network structure algorithm combining PSO and GA, using random mutation and crossover operations to redefine particle update rules, and achieved good results on small data sets. The above algorithms are all improved solutions for the PSO, but the initial particles are generated randomly, and there is no restriction on the initial particles, which will lead to excessive randomness of the algorithm. If there is no restriction, the result will be uncertain. Therefore, some scholars have proposed the use of mutual information to limit the initialization process of particles. Reference [22] proposed a PSO algorithm based on restriction technology, which uses mutual information to restrict the initialization of PSO and narrow the search space of the algorithm, but mutual information establishes global information between nodes. When the data dimension is high, the model construction efficiency is low. Reference [23] proposed a BN structure learning algorithm based on evolutionary strategy, and designed a new initial structure encoding method to avoid the generation of illegal structures cyclic graphs. However, due to the relatively complex coding, subsequent calculations require higher computer performance and are not easy to implement. Reference [24] proposed to use the maximum likelihood tree (MLT) to create an initial population, and at the same time apply the binary PSO based on immune strategy to BN structure learning, because the tree structure is used in the initial population construction process, the dependence on the root node is too large. When the root node error occurs, it will cause the failure of all network structures, thus affecting the accuracy of the algorithm.

It can be seen from the above references that in the process of learning BN using PSO, it plays a vital role in the construction of initial nodes, especially if the local information between nodes can be fully utilized to construct the initial network structure. Then the amount of calculation will be greatly reduced in the subsequent structure search process. Therefore, aiming at the above problems, we propose a BN structure learning algorithm based on local information. This method first uses the local search-based Max-Min Parents and Children (MMPC) algorithm to construct an undirected network framework to optimize the initial population and reduce the search space of the PSO. Then a PSO algorithm with local information is proposed, and the optimization ability of the algorithm is strengthened by constructing a new position and velocity update strategy. Compared with other previous BN structure learning algorithms, this article has the following contributions:

a) In the early stage of the algorithm, the construction of an undirected network framework based on local search was proposed to reduce the search space of the network structure. The algorithm limits the search space of PSO, simplifies the search mechanism, and improves the convergence speed of subsequent searches using PSO.
b) The concept of introducing local information in the constraint phase is put forward, and the related constraints of the information between nodes are increased to ensure the consistency of the selected local parent-child node information and global information. Improve the accuracy of network nodes in the early stage of the algorithm. The choice of the potential parents is very important for the success of structure learning, as a wrong initial choice will result in poor structure.
c) A discrete binary PSO algorithm is proposed. When applied to the BN structure learning problem, however, there are some difficulties for PSO to handle its two main elements (i.e. position and velocity) and associated updating rules. The velocity update strategy proposes a new improvement idea, which greatly simplifies the update process, reduces the algorithm complexity, and avoids PSO from falling into the local optimum.

Finally, taking the standard network model as an example, comparing with other algorithms, it demonstrates the specific and strong learning ability of this algorithm in BN structure learning.

The structure of the paper is as follows. Section II introduces the BN structure learning description and PSO algorithm. The BN structure learning method via local information in Section III. Simulation results are shown and analysis in Section IV. Section V draws the conclusions of the work.

II. BAYESIAN NETWORK STRUCTURE LEARNING

BN can be expressed as BN $= (G, \Theta)$, with $G = (X, E)$ for DAG. Where $X = \{x_1, x_2, x_3, \ldots, x_n\}$ is the set of nodes in the network, E is the set of directed edges in the network, and Θ is the set of conditional probability distributions of the nodes, $\Theta = \{P[x_i|\tau(x_i)], x_i \in X\}$, indicates the dependency relationship between a node and its parent node.

Therefore, the joint probability distribution of all nodes is:

$$\begin{aligned} P(x_1, x_2, \ldots, x_n) &= \prod_{i=1}^{n} P(x_i|x_1, x_2, \ldots, x_{i-1}) \\ &= \prod_{i=1}^{n} P(x_i|\tau(x_i)) \end{aligned} \tag{1}$$

The goal of BN structure learning is to use a training sample set D, $D = \{d_1, d_2, \ldots, d_m\}$ for a set of random variables $x_1, x_2, x_3, \ldots, x_n$ to determine the most suitable network topology S.

In the learning method based on score search, we need to find the BN structure model that makes the score the best, which is as follows:

$$S \leftarrow \arg bestScore\,(S, D) = P\,(S|D) \tag{2}$$

Commonly used scoring functions are Bayesian information criterion (BIC), Bayesian Dirichlet equivalent (BDE), minimum description length (MDL), K2, etc. We use BIC as the fitness function. A higher score indicates a higher degree of matching. The BIC scoring function is defined as follows:

$$Score_{BIC} = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{j=1}^{r_i} m_{ijk} \log\left(\frac{m_{ijk}}{m_{ij}}\right) - \sum_{i=1}^{n} q_i\,(r_i - 1)\frac{1}{2}\log(m) \tag{3}$$

where n represents the total number of nodes in the network, r_i represents that node x_i has r_i possible values, and q_i represents that the parent node of node x_i has q_i possible values. $m_{ij} = \sum_{k=1}^{r_i} m_{ijk}$ represents the number of x_i when the parent section set takes the value of j.

After the scoring function is determined, the next task is to select a suitable search algorithm to apply to the BN structure to find the network structure with the highest score.

When the PSO algorithm is applied to BN structure learning, the corresponding particle is the discrete structure of BN, and the structure is represented by a binary-coded square matrix, and PSO has some difficulties in processing its two main elements: position and velocity, also its updating rules. Therefore, the traditional PSO cannot meet this requirement, the Discrete Binary Particle Swarm Optimization (DBPSO) is needed to solve it.

The analysis of the particle trajectory shows that when the personal extreme value and the global extreme value are not equal, the particle will oscillate between the two extreme values. When the two extreme values are equal and the particle is not equal to the two extreme values, the particle will show a tendency to converge towards the extreme value, and gradually approach the two extreme value solutions, and the exploration ability is strong. Therefore, how to improve the local search ability of the algorithm near the optimal solution while maintaining the ability of the algorithm to escape from the local optimal solution is a problem that the traditional DBPSO algorithm needs to solve.

Different from the traditional PSO where the position of each particle is coded by real numbers, the position of each particle in the DBPSO algorithm is coded with binary 0 and 1. And the value of velocity is $\{-1, 0, 1\}$.

Therefore, the new PSO formula is redefined to obtain the velocity and position update formula of the DBPSO algorithm based on discrete space. The update formula is as follows:

$$\boldsymbol{V}_s^{t+1} = \boldsymbol{V}_s^{t} \oplus (\boldsymbol{P}_s^{t} \ominus \boldsymbol{X}_s^{t}) \oplus (\boldsymbol{P}_g^{t} \ominus \boldsymbol{X}_s^{t}) \tag{4}$$

$$\boldsymbol{X}_s^{t+1} = \boldsymbol{X}_s^{t} \boxplus \boldsymbol{V}_s^{t+1} \tag{5}$$

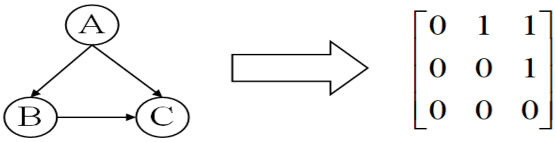


FIGURE 1. Network node coding schematic.

where, $\boldsymbol{V}_s^t$ and $\boldsymbol{X}_s^t$ respectively represent the velocity and position of the particle s in the t iteration; $\boldsymbol{P}_s^t$ represents personal best position of s-th particle in t-th iteration, namely $\boldsymbol{P}_s^t$ = optimal$\{\boldsymbol{X}_s^1, \boldsymbol{X}_s^2, \cdots, \boldsymbol{X}_s^t\}$; $\boldsymbol{P}_g^t$ represents global best position in t-th iteration $\boldsymbol{S} = \{1, 2, \cdots, n_S\}$ (population size is n_S), namely $\boldsymbol{P}_g^t$ = optimal$\{\boldsymbol{P}_1^t \cdots, \boldsymbol{P}_s^t, \cdots, \boldsymbol{P}_{n_S}^t\}$.

III. BN STRUCTURE LEARNING ALGORITHM VIA LOCAL INFORMATION CONSTRAINT

The BN structure learning algorithm proposed in this paper accelerates the efficiency of finding the optimal BN structure algorithm by reducing the search space. First, use the MMPC based on the restriction technology to construct the framework of the undirected network, thereby solving the problem of the large randomness of the initial particles in the later optimization process using the PSO algorithm. Then, the DPSO-based scoring search algorithm is executed to perform iterative optimization and finally determine the connection edges and directions of the network structure.

A. PARTICLE CODING METHOD

First, define the network structure of BN as a DAG, and each particle corresponds to a candidate bn structure. The position of the particle is represented by a matrix $G\,(n, n)$, n is the number of nodes in the network, and the number of nodes in each DAG is the dimension of the particle. As shown in Figure 1. The definition of each element in the matrix is:

$$g_{ij} = \begin{cases} 1, & i \text{ is the parent node of node } j \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

In the PSO algorithm, the particles move at a certain velocity. The velocity is used to measure the difference between two particles and indicate the possibility of the position state. Therefore, in order to be consistent with the definition of particle ‘‘position’’, the velocity is also defined as a $n \times n$ order matrix representation. Since each dimension in the model is restricted to 1 or 0 in DBPSO, the value of velocity is $\{-1, 0, 1\}$. That is, the velocity of particle i at step t is V_{ij}.

$$V_{ij} = \begin{cases} 1, & \text{add edges between nodes} \\ 0, & \text{constant} \\ -1, & \text{delete edges between nodes} \end{cases} \tag{7}$$

where, 1 means adding a directed edge between nodes i and j, 0 means the relationship between node i and node j remains unchanged, and -1 means deleting the edge between node i and node j. The element in the i row and j column is $V_{ij} \in \{-1, 0, 1\}$.

B. LOCAL INFORMATION CONSTRUCTION

In this paper, we use the local search capability of the MMPC algorithm to obtain the initial structure undirected graph of BN with local information, and then use conditional relative average entropy (CRAE) to connect the undirected structure graph determination of the side direction. Finally, an initial population satisfying the constraint conditions is generated based on the initial structure.

The accuracy of BN structure learning depends to a certain extent on the selection of the initial network structure. The initial structure diagram construction method based on local information not only greatly reduces the search space of the algorithm, but also accelerates the convergence speed of the algorithm. It also uses the initial population. It carries the parents and children set information of the node, which improves the ability of the subsequent iterative optimization process of the PSO to jump out of the local extreme value. The specific implementation process is described as follows:

Step 1: First use the MMPC algorithm to get the candidate parents and children (CPC) nodes of each node;

Step 2: Use mutual information (MI) to constrain the connection edges of nodes in the parents-children nodes set; Determine the two connecting edges with the largest MI between each node and the parents-children nodes;

$$I\left(X_i, X_j\right) = \sum_{X_i, X_j} p\left(X_i, X_j\right) \log \frac{p\left(X_i, X_j\right)}{p(X_i)p(X_j)} \tag{8}$$

The greater the $I(X_i, X_j)$, the stronger the dependence between the two nodes. Because of the symmetry of MI, that is, $I(X_i, X_j) = I(X_j, X_i)$, the connection edge corresponding to each node is undirected. In this way, an initial DAG is obtained. So far, the initial DAG has carried the local information between nodes.

Step 3: Determine the direction of the connecting edge. According to the known DAG, CRAE is used to determine the direction of each connecting edge. CRAE is defined as:

$$CRAE\left(X_j \rightarrow X_i\right) = \frac{H\left(X_i|X_j\right)}{H\left(X_i\right) \cdot |X_i|} \tag{9}$$

where $|X_i|$ is the number of values of variable X_i, $H\left(X_i\right)$ is the entropy of discrete random variable X_i.

$$H\left(X_i\right) = -\sum_{i=1}^{N} p\left(x_i\right) \log p\left(x_i\right) \tag{10}$$

$H(X_i|X_j)$ is the conditional entropy of discrete random variable X_i.

$$H\left(X_i|X_j\right) = \sum_{i=1}^{N}\sum_{j=1}^{M} p\left(x_i, x_j\right) \log p\left(x_i|x_j\right) \tag{11}$$

If $CRAE(X_i \rightarrow X_j) \geq CRAE(X_j \rightarrow X_i)$ the orientation should be $i \rightarrow j$

Step 4: Obtain the initial population. Due to the need to construct a large number of initial particle structures, two random operations are set: adding edges and reversing edges to generate the initial population of the PSO algorithm. In order to obtain more connected edge information, more local information is obtained by increasing the number of connected edges between some nodes, and for each edge in G, that is, when $g_{ij} = 1$, we change node i and node j between the directions to increase the diversity of the initial population particles.

Step 5: To achieve de-ring operation. Because the BN must be a DAG, no matter in the process of generating the initial population or applying the PSO algorithm cyclic search process, the new position of the particle update needs to be de-ringed. The new location is the generated network structure. We must first determine whether there is a loop in the network structure. If it does, then we need to remove the loop.

So far, we have generated the initial structure graph $G = (X, E)$ through the local information algorithm, where n is the total number of nodes in the BN structure, $G = (g_{ij})_{n*n}, i, j = (1, 2, \ldots, n)$ and G is used to represent the adjacency matrix of BN.

C. PARTICLE UPDATE RULES

Due to the redefinition of particle position and velocity, the update operation rules for particles have also been redefined.

Position and position subtraction operation $\boldsymbol{X}_1 \ominus \boldsymbol{X}_2$: represents the change of particle position, that is, velocity, so $\boldsymbol{X}_1 \ominus \boldsymbol{X}_2$ is the velocity obtained by subtracting the personal extreme position and the current position, or the global extreme position and the current position. The velocity is a set of several new replacement sequences.

$$V = \begin{cases} 1 & x_{ij} = 1,\ x'_{ij} = 0 \\ 0 & x_{ij} = x'_{ij} \\ -1 & x_{ij} = 0,\ x'_{ij} = 1 \end{cases} \tag{12}$$

where x_{ij} is the element in the i row and j column in the position matrix P^t_s or P^t_g, and x'_{ij} is the element in the i row and j column in the position matrix X^t_s.

The new velocity after two position calculations is obtained by subtraction operation, as shown in the Figure 2.

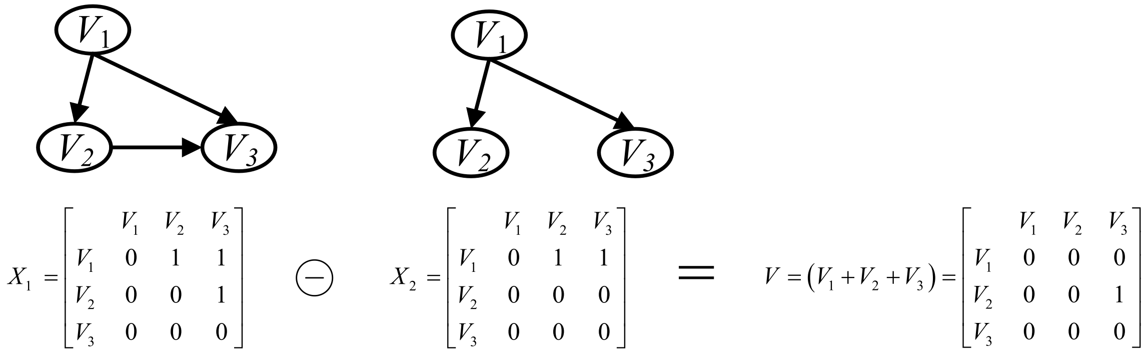
Position and velocity addition operation ($\boldsymbol{X} \boxplus \boldsymbol{V}$): represents an addition operation based on a replacement operation, and the result is a new position.

For example, the following velocity:

$$V = (V_1 + V_2 + V_3) = \left[\begin{array}{cccc} & V_1 & V_2 & V_3 \\ V_1 & 0 & 1 & -1 \\ V_2 & 0 & 0 & -1 \\ V_3 & -1 & 1 & 0 \end{array}\right] \tag{13}$$

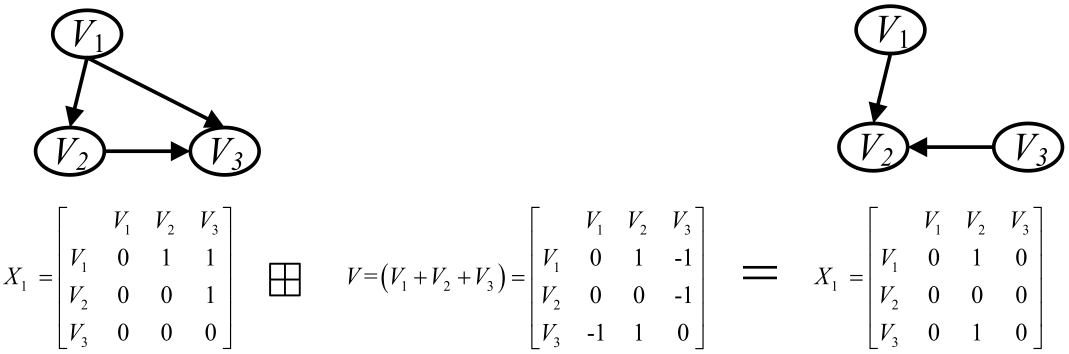
When the following conditions occur in the particle position and velocity, the particle position is removed.

$$G_p = \begin{cases} 1 & x_{ij} = 0, v_{ij} = 1 \; or \; x_{ij} = 1, \\ & v_{ij} = 0 \; or \; x_{ij} = 1, v_{ij} = 1 \\ 0 & x_{ij} = 0, v_{ij} = 0 \; or \; x_{ij} = 1, \\ & v_{ij} = -1 \; or \; x_{ij} = 0, v_{ij} = -1 \end{cases} \tag{14}$$



V_1
V_2
V_3
$X_1 = \begin{array}{c|ccc} & V_1 & V_2 & V_3 \\ V_1 & 0 & 1 & 1 \\ V_2 & 0 & 0 & 1 \\ V_3 & 0 & 0 & 0 \end{array}$
$\ominus$
$X_2 = \begin{array}{c|ccc} & V_1 & V_2 & V_3 \\ V_1 & 0 & 1 & 1 \\ V_2 & 0 & 0 & 0 \\ V_3 & 0 & 0 & 0 \end{array}$
$=$
$V = (V_1 + V_2 + V_3) = \begin{array}{c|ccc} & V_1 & V_2 & V_3 \\ V_1 & 0 & 0 & 0 \\ V_2 & 0 & 0 & 1 \\ V_3 & 0 & 0 & 0 \end{array}$

FIGURE 2. Velocity update indication.



V_1
V_2
V_3
$X_1 = \begin{array}{c|ccc} & V_1 & V_2 & V_3 \\ V_1 & 0 & 1 & 1 \\ V_2 & 0 & 0 & 1 \\ V_3 & 0 & 0 & 0 \end{array}$
$\boxplus$
$V = (V_1 + V_2 + V_3) = \begin{array}{c|ccc} & V_1 & V_2 & V_3 \\ V_1 & 0 & 1 & -1 \\ V_2 & 0 & 0 & -1 \\ V_3 & -1 & 1 & 0 \end{array}$
$=$
$X_1 = \begin{array}{c|ccc} & V_1 & V_2 & V_3 \\ V_1 & 0 & 1 & 0 \\ V_2 & 0 & 0 & 0 \\ V_3 & 0 & 1 & 0 \end{array}$

FIGURE 3. Position update indication.

The position update of the X particles after the velocity change is shown in Figure 3.

Velocity and velocity addition operation ($\boldsymbol{V}_1 \oplus \boldsymbol{V}_2 \oplus \boldsymbol{V}_3$): the union operation of sets, which means that three velocity sets are merged into one velocity set.

Where, the initial velocity of the particle is

$$V_1 = [V_{1,1}, V_{1,2}, \ldots V_{1,n}] \tag{15}$$

The velocity obtained by the subtraction operation, the personal extremum value update velocity is

$$V_2 = [V_{2,1}, V_{2,2}, \ldots V_{2,n}] \tag{16}$$

The velocity obtained by the subtraction operation, the global extremum update velocity is

$$V_3 = [V_{3,1}, V_{3,2}, \ldots V_{3,n}] \tag{17}$$

The new velocity update formula obtained from the addition operation.

$$V_{add} = \begin{cases} V_{1,i} & rand < P_{r1} \\ V_{2,i} & rand < P_{r2} \\ V_{3,i} & rand < p_{r3} \end{cases} \tag{18}$$

where $P_{r1} + P_{r2} + P_{r3} = 1$.

It can be seen from the definition of P_{r1}, P_{r2}, P_{r3} that in the initial stage of the algorithm, in order to improve the global search ability of particles, P_{r1} should be set to a larger value. As the search process proceeds, the local search ability of particles needs to be strengthened, and the value of P_{r2}, P_{r3} gradually increases. Therefore, the definition of P_{r1}, P_{r2}, P_{r3} is as follows:

$$P_{ri} = \frac{f(r_i)}{\sum_{j=1}^{N} f(r_i)} \tag{19}$$

where $f(r_i)$ is the fitness value of each individual.

Then calculate the cumulative probability of each individual

$$q_i = \sum_{j=1}^{i} f(r_i) \tag{20}$$

Randomly generate $rand \in [0, 1]$, if $P_{ri} > rand$, select individual V_i.

D. ALGORITHM IMPLEMENTATION STEPS

The specific implementation process of the entire algorithm is given below.

E. THE TIME COMPLEXITY OF ALGORITHM

In this section, we will analyze the time complexity of the algorithm, and use the worst-case time complexity as the

```
    Inputs: Training data $D = \{x_1, \cdots x_n\}$ N is the number
of the particles, Maximum number of iterations T
    Outputs: BN structure
    Local information construction
$CPC = \varnothing$
    repeat
        Fill the CPC of the target node
        Check independence between nodes
    until CPC has not changed
    return CPC
    for $i = 1 : node\_size.$ do
        $MI = I(X_i, X_j)$
    end for
    return DAG
    for $i = 1 : node\_size$ do
        calculate $CRAE(X_i \to X_j)$ and $CRAE(X_j \to X_i)$
        if $CRAE(X_i \to X_j) \geq CRAE(X_j \to X_i)$ then
     $X_{ij} = 1$
        end if
    end for
Adding edges and reversing edges
    return $G = (g_{ij})_{n*n}$
Initialize entire particle parameters, velocity matrix V and
position matrix P.
    while $t \leq \max iteration$ do
     for $i = 1 : pop\_size$ do
         Update the velocity
         Randomly generate $V_1$
         $V_2 = (\boldsymbol{P}_s^t \ominus \boldsymbol{X}_i^t)$, $V_3 = (\boldsymbol{P}_g^t \ominus \boldsymbol{X}_i^t)$
         $V_{add} = \boldsymbol{V}_1 \oplus \boldsymbol{V}_2 \oplus \boldsymbol{V}_3$
         Update the position
         $\boldsymbol{X}_i^{t+1} = \boldsymbol{X}_i^t \boldsymbol{V}_i^{t+1}$
         if $score(\boldsymbol{X}_i^{t+1}) > score(\boldsymbol{X}_i^t)$
             $\boldsymbol{P}_s^{t+1} = \boldsymbol{X}_i^{t+1}$
         end if
         if $score(\boldsymbol{P}_s^{t+1}) > score(\boldsymbol{P}_g^t)$
             $\boldsymbol{P}_g^{t+1} = \boldsymbol{P}_s^{t+1}$
         end if
         $t = t + 1$
        end for
    end while
    return $\boldsymbol{X}_g^t$
```

basis for judgment. In the algorithm mentioned above, the algorithm is divided into 5 steps in total, and then we will discuss the complexity of each step separately. Assume that the maximum possible number of states for any node is r, the number of data samples is N, the number of particles is p, the number of iterations is g and the number of nodes is n.

In step 1, we first calculate the set of parents and children nodes of each node. According to the steps of the proposed algorithm, the number of cases in this step is bounded by $O\left(n \cdot 2^{|CPC|}\right)$. In step 2, we calculate the MI value between each pair of nodes. Due to the symmetry of MI, we only need to calculate $n\,(n-1)\,/2$ MI values. Since r is usually much smaller than N, the complexity of calculating each MI value $O\left(4r^2 + N\right)$ can be simplified to $O\,(N)$, so the complexity of calculating the MI value of all node pairs is $O\left(Nn^2\right)$. In step 3, we use CRAE to confirm the direction of the generated DAG, so from the previous calculation content, we can see that the algorithm responsibility degree of this step is $O\left(n^2\right)$. Since the step 4 is to initialize the particles, the time complexity is $O\,(p)$. The number of particles is much smaller and constant compared to other data, so it can be ignored. In step 5, the updated PSO update rule iterates the generated particles, so the update complexity of p particles under g iterations is $O\left(gpn^2\right)$.

IV. EXPERIMENTAL

A. PARAMETER SETTING AND EVALUATION INDEX

The experimental platform used in our paper is a personal computer with Intel Core i5-5300U, 2.30 GHz, 32-bit architecture, 4 GB RAM memory and under Windows 7. The programs are all compiled using the MATLAB software release R2014a.

In order to evaluate the quality of the learned network structure, this paper uses the more intuitive Hamming distance and BIC score value, algorithm running time, and number of iterations as the standard. The Hamming distance is defined as $H(G) = M(G)+A(G)+I(G)$, which is the number of missing edges $M(G)$, added edges $A(G)$, and inversion edges $I(G)$ compared with the original network. Usually a good search algorithm can learn from the data set as similar to the original network as possible, that is, the number of missing edges, redundant edges and edges in the opposite direction is the lowest. The standard BIC scoring criterion indicates the BIC score value when the algorithm learns the optimal structure. The higher the score, the better the result.

The algorithm performance evaluation indicators are as follows:

BIC: The average BIC score of the final individuals.

SHD: The average structure Hamming distance between the best individual and the original BN structure including the number of mistakenly added, detected and reversed directed edges.

ET: The average execution time over all trials in seconds.

IT: The average number of iterations needed to find an optimal result.

B. EXPERIMENTAL RESULTS AND ANALYSIS

In order to test the performance of the algorithm, this article uses the general Benchmark data set ASIA network, CHILD network and ALARM network to complete the experiment. Each algorithm is run 100 times, the maximum number of iterations is 300. After obtaining the results of each time, statistics are performed to obtain the average value of 100 independent repeated experiments for each algorithm. Use the same network data set every time. The specific data composition is shown in Table 1. According to the standard

TABLE 1. Standard test network data.

Data set	Samples	Nodes	Edges	Max in/out degree	Domain
ASIA-500	500	8	8	2	2
ASIA-1000	1000	8	8	2	2
ASIA-5000	5000	8	8	2	2
CHILD-500	500	20	25	2/7	2~6
CHILD-1000	1000	20	25	2/7	2~6
CHILD-5000	5000	20	25	2/7	2~6
ALARM-500	500	37	46	4/5	2~4
ALARM-1000	1000	37	46	4/5	2~4
ALARM-5000	5000	37	46	4/5	2~4

TABLE 2. Experimental results of LIPSO performance for constructing the ASIA network using different number of samples and different algorithms.

Data set		LIPSO	MMHC	BNC-PSO	PSO
ASIA-500	BIC	-23654.24	-24723.55	-24798.23	-24579.84
	SHD	1.466	1.901	1.598	2.109
	ET	40.115	43.807	41.370	49.087
	IT	35.233	-	36.475	45.583
ASIA-1000	BIC	-25372.81	-26086.12	-25937.63	-26237.12
	SHD	1.233	1.823	1.304	1.897
	ET	19.532	22.908	19.847	24.343
	IT	28.557	-	30.438	37.495
ASIA-5000	BIC	-34742.74	-35593.48	-35028.52	-35673.68
	SHD	1.100	1.272	1.172	1.673
	ET	14.233	16.394	14.359	17.309
	IT	21.935	-	22.494	29.485

network structure, first use the BNT tool to generate data sets with sample sizes of 500, 1000, and 5000 according to the standard probability table, and then use the algorithm in this paper to learn the structure of the generated multiple sets of data.

The results of the operation are compared with the MMHC algorithm [25], BNC-PSO algorithm [21], and PSO algorithm [22] to verify the effect of this algorithm. We named the algorithm local information PSO algorithm (LIPSO). Each algorithm is run 100 times, the maximum number of iterations is 300.Table 2 shows the experimental results for constructing the ASIA network using different number of samples and different algorithms. Table 3 shows the experimental results in terms of the BIC score, Hamming distance, the number of iterations and execution time for CHILD network. Table 4 shows the experimental results of LIPOS performance for constructing the ALARM network.

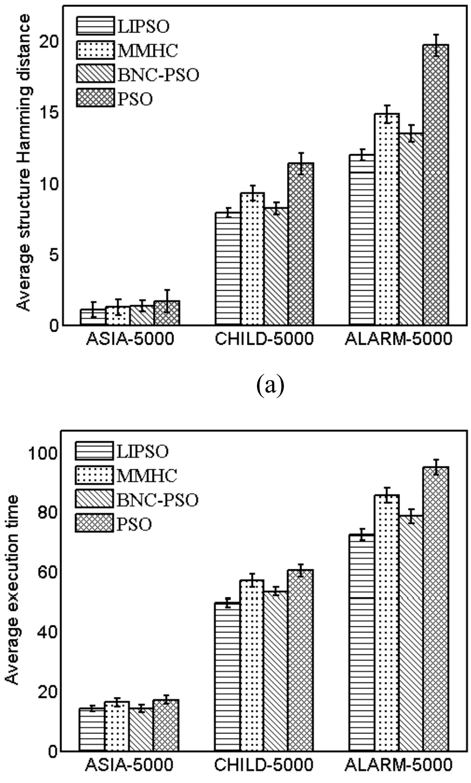
TABLE 3. Experimental results of LIPSO performance for constructing the CHILD network using different number of samples and different algorithms.

Data set		LIPSO	MMHC	BNC-PSO	PSO
CHILD-500	BIC	-37047.42	-37893.23	-37394.42	-37348.47
	SHD	9.800	13.943	11.383	15.730
	ET	141.788	183.394	151.949	200.487
	IT	51.856	-	55.821	59.840
CHILD-1000	BIC	-36903.23	-37184.83	-37027.62	-37234.94
	SHD	8.769	11.384	10.472	13.472
	ET	68.362	75.204	71.390	78.389
	IT	47.853	-	49.134	52.921
lCHILD-5000	BIC	-35294.49	-35828.42	-35693.47	-36942.36
	SHD	7.938	9.337	8.047	11.394
	ET	49.815	57.138	53.490	60.479
	IT	42.803	-	44.532	46.721

TABLE 4. Experimental results of LIPSO performance for constructing the ALARM network using different number of samples and different algorithms.

Data set		LIPSO	MMHC	BNC-PSO	PSO
ALARM-500	BIC	-42948.47	-43389.48	-43137.48	-43793.94
	SHD	21.838	28.214	24.899	35.345
	ET	204.791	274.871	220.011	290.125
	IT	65.370	-	68.347	74.943
ALARM-1000	BIC	-43690.80	-42947.47	-43484.74	-44959.33
	SHD	15.8679	24.825	17.823	29.834
	ET	99.613	125.221	105.633	137.721
	IT	58.790	-	61.247	72.421
ALARM-5000	BIC	-49993.32	-50930.32	-50230.58	-51048.58
	SHD	11.964	14.845	13.478	19.733
	ET	72.588	85.836	78.834	95.477
	IT	51.590	-	53.735	61.781

From the data in Table 2 to Table 4 , it can be seen that with the same sample dataset, the network structure learned by the LIPSO algorithm is closer to the real network. On the ASIA dataset, PSO has the worst learning effect, and the algorithm in this paper has the best learning effect, followed by BNC-PSO. At the same time, on the CHILD dataset and ALARM dataset, the LIPSO algorithm in this paper can achieve the highest BIC score and the smallest SHD. In 100 independent simulation experiments, the BIC score value of our algorithm is higher than the score value of other algorithms in different networks, and the BIC score value represents the degree of similarity matching with the real network structure. Because LIPSO uses local information to better restore the network structure in the initial constraint stage of the algorithm, it is easier to obtain the optimal result when the BIC score is used for calculation at the end. Therefore, it can be seen from the table that the algorithm in this paper has more advantages in BIC score value than other algorithms. At the same time, the index of SHD can also explain this problem. Our algorithm has achieved the



(a)

(b)

FIGURE 4. **Error bar graph of two measures for four algorithms on different data set. (a) Average structure Hamming distance of the four algorithms; (b) Average execution time of the four algorithms.**

smallest SHD under several different networks, which proves the effectiveness of the algorithm. The reason for this result is obtained in the early stage of the algorithm. The local information of the network is closer to the real network, so that our algorithm obtains more accurate information from the beginning, while the initial structure of other algorithms is mostly generated by random generation. The acquisition of local information is effective in the final search stage. The result of the improvement has helped.

In order to verify the convergence performance of the algorithm, the algorithm in this paper is the optimal algorithm in terms of convergence time and operating efficiency under different network dataset. Because the MMHC algorithm is not a swarm intelligence algorithm, it does not compare iteration time. As the algorithm itself improves the PSO and proposes a simpler update method, the two experimental results, the running time and the number of iterations, also reflect better results. For different experimental networks, due to the difference in the number of nodes, it takes more time to calculate when there are networks with a large number of nodes to learn, because the algorithm avoids the problem of local optimization similar to the previous traditional PSO network. So, our algorithm has a faster number of iterations, while simplifying the update operation, it can better reflect the efficiency of the algorithm.

In order to clearly compare the performance of these four algorithms, Figure 4 (a) (b) respectively describes the average Hamming distance and average execution time under 5000 data sets. At the same time, Figure 4 also reflects the standard of each group of experiments. Difference error graph. It can also be seen from Figure 4 that compared with other algorithms, when the network structure is relatively simple, the gap between several algorithms is not obvious, but LIPSO can use less execution time to obtain the best BN structure, and at the same time. Compared with other algorithms, it has a smaller standard deviation value, which reflects the stability of the algorithm. Because other algorithms mostly use random generation methods in the initial structure stage, the calculation results are unstable. However, as the network scale increases. There is also an increase in the error rate, which is mainly reflected in the increase in the selection of nodes and edges in the calculation, resulting in learning difficulty in the final learning, but the final result of this paper is significantly better than other algorithms in the learning structure.

V. CONCLUSION

The BN structure learning algorithm based on local information proposed in this paper, by constructing a new initial structure diagram with local information, enables the use of PSO to obtain more node information later. By constructing the framework of the initial undirected network, the search space is narrowed. Then use the DBPSO optimization algorithm to build a new position and velocity update strategy to strengthen the algorithm's optimization ability and improve the algorithm's convergence efficiency. It can be seen from the experimental results that in the case of the same sample size, the learned network structure is closer to the real network, and in the shortest effective time, the network structure with higher score is learned, and the convergence effect is the best. As the number of nodes increases, there are still challenges in terms of time and accuracy when processing networks with higher complexity and more node information. In future work, we consider studying large-scale networks and algorithms containing a kind of node ordering information.

REFERENCES

[1] M. Bartlett and J. Cussens, "Integer linear programming for the Bayesian network structure learning problem," *Artif. Intell.*, vol. 244, pp. 258–271, Mar. 2017.

[2] Y. Cao, M. Raoof, E. Szabo, J. Ottosson, and I. Näslund, "Using Bayesian networks to predict long-term health-related quality of life and comorbidity after bariatric surgery: A study based on the scandinavian obesity surgery registry," *J. Clin. Med.*, vol. 9, no. 6, p. 1895, Jun. 2020.

[3] W. Li, P. Zhang, H. Leung, and S. Ji, "A novel QoS prediction approach for cloud services using Bayesian network model," *IEEE Access*, vol. 6, pp. 1391–1406, 2018.

[4] S. Li, S. Chen, and Y. Liu, "A method of emergent event evolution reasoning based on ontology cluster and Bayesian network," *IEEE Access*, vol. 7, pp. 15230–15238, 2019.

[5] L. Wang, G. Wang, Z. Duan, H. Lou, and M. Sun, "Optimizing the topology of Bayesian network classifiers by applying conditional entropy to mine causal relationships between attributes," *IEEE Access*, vol. 7, pp. 134271–134279, 2019.

[6] Y. Jiang, Z. Liang, H. Gao, Y. Guo, Z. Zhong, C. Yang, and J. Liu, "An improved constraint-based Bayesian network learning method using Gaussian kernel probability density estimator," *Expert Syst. Appl.*, vol. 113, pp. 544–554, Dec. 2018.

[7] B. Andrews, J. Ramsey, and G. F. Cooper, "Scoring Bayesian networks of mixed variables," *Int. J. Data Sci. Analytics*, vol. 6, no. 1, pp. 3–18, Aug. 2018.

[8] A. L. Madsen, F. Jensen, A. Salmerón, H. Langseth, and T. D. Nielsen, "A parallel algorithm for Bayesian network structure learning from large data sets," *Knowl.-Based Syst.*, vol. 117, pp. 46–55, Feb. 2017.

[9] X. Sun, C. Chen, L. Wang, H. Kang, Y. Shen, and Q. Chen, "Hybrid optimization algorithm for Bayesian network structure learning," *Information*, vol. 10, no. 10, p. 294, Sep. 2019.

[10] H. Li and H. Guo, "A hybrid structure learning algorithm for Bayesian network using experts' knowledge," *Entropy*, vol. 20, no. 8, p. 620, Aug. 2018.

[11] X. Zhang, Y. Xue, X. Lu, and S. Jia, "Differential-evolution-based coevolution ant colony optimization algorithm for Bayesian network structure learning," *Algorithms*, vol. 11, no. 11, p. 188, Nov. 2018.

[12] C. Contaldi, F. Vafaee, and P. C. Nelson, "Bayesian network hybrid learning using an elite-guided genetic algorithm," *Artif. Intell. Rev.*, vol. 52, no. 1, pp. 245–272, Jun. 2019.

[13] C. Yang, H. Gao, X. Yang, S. Huang, Y. Kan, and J. Liu, "BnBeeEpi: An approach of epistasis mining based on artificial bee colony algorithm optimizing Bayesian network," in *Proc. IEEE Int. Conf. Bioinf. Biomed. (BIBM)*, Nov. 2019, pp. 232–239.

[14] S. Lee and S. B. Kim, "Parallel simulated annealing with a greedy algorithm for Bayesian network structure learning," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 6, pp. 1157–1166, Jun. 2020.

[15] X. Liu and X. Liu, "Structure learning of Bayesian networks by continuous particle swarm optimization algorithms," *J. Stat. Comput. Simul.*, vol. 88, no. 8, pp. 1528–1556, May 2018.

[16] Y. Zhang, D.-W. Gong, and Z.-H. Ding, "Handling multi-objective optimization problems with a multi-swarm cooperative particle swarm optimizer," *Expert Syst. Appl.*, pp. 13933–13941, May 2011.

[17] Y. Zhang, D.-W. Gong, and Z. Ding, "A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch," *Inf. Sci.*, vol. 192, pp. 213–227, Jun. 2012.

[18] X.-F. Song, Y. Zhang, Y.-N. Guo, X.-Y. Sun, and Y.-L. Wang, "Variable-size cooperative coevolutionary particle swarm optimization for feature selection on high-dimensional data," *IEEE Trans. Evol. Comput.*, vol. 24, no. 5, pp. 882–895, Oct. 2020.

[19] Y. Zhang, D.-W. Gong, and J. Cheng, "Multi-objective particle swarm optimization approach for cost-based feature selection in classification," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 14, no. 1, pp. 64–75, Jan. 2017.

[20] J. Wang and S. Liu, "A novel discrete particle swarm optimization algorithm for solving Bayesian network structures learning problem," *Int. J. Comput. Math.*, vol. 96, no. 12, pp. 2423–2440, Dec. 2019.

[21] S. Gheisari and M. R. Meybodi, "BNC-PSO: Structure learning of Bayesian networks by particle swarm optimization," *Inf. Sci.*, vol. 348, pp. 272–289, Jun. 2016.

[22] G. Xiaoguang and G. Z. Di Ruohai, "Bayesian network structure learning based on improved particle swarm optimization," *J. Northwestern Polytech. Univ.*, vol. 32, no. 5, pp. 749–755, 2014.

[23] Y. Zhu, D. Liu, and H. Jia, "A new evolutionary computation based approach for learning Bayesian network," *Procedia Eng.*, vol. 15, pp. 4026–4030, 2011, doi: 10.1016/j.proeng.2011.08.755.

[24] X.-L. Li and X.-D. He, "A hybrid particle swarm optimization method for structure learning of probabilistic relational models," *Inf. Sci.*, vol. 283, pp. 258–266, Nov. 2014.

[25] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," *Mach. Learn.*, vol. 65, no. 1, pp. 31–78, Oct. 2006.

KUN LIU received the master's degree from Hainan University, Haikou, China, in 2016, where he is currently pursuing the Ph.D. degree with the School of Information and Communication Engineering. His research interests include Bayesian networks and intelligent optimization.

YANI CUI received the Ph.D. degree from Hainan University, Haikou, China, in 2017. She is currently a Lecturer with the School of Information and Communication Engineering, Hainan University. Her research interests include intelligent systems, collaborative control, and Bayesian networks.

JIA REN received the Ph.D. degree from Northwestern Polytechnical University, Xi'an, China, in 2011. He is currently a Professor with the School of Information and Communication Engineering, Hainan University. His research interests include intelligent control, machine learning, and Bayesian networks.

PEIRAN LI is currently pursuing the master's degree with the School of Information and Communication Engineering, Hainan University. Her research interests include Bayesian networks and intelligent optimization.