A Novel Particle Swarm Optimization Algorithm for Network Clustering

Zhaoxing Li1, 2*, Lile, He1, Ze Li1, Yunrui Li3
1 School of Mechanical & Electrical Engineering
Xi’an University of Architecture and Technology
710055 Xi’an, China
2 College of Information Engineering
Yulin University, 719000 Yulin, China
3 Department of Electrical and Computer Engineering
Wayne State University,
MI 48202 Detroit. USA
39812767@qq.com

ABSTRACT: The use of complex network analysis has gathered momenta in both theoretical and empirical studies. Network clustering plays an important role in network analysis. This paper models the network-clustering task as an optimization problem. A novel discrete particle swarm optimization algorithm is introduced to solve the modeled optimization problem. Particle swarm optimization is a stochastic searching algorithm, and it cannot avoid prematurity. To improve the performance of the algorithm, a new particle status update principle is defined, a novel turbulence operation is proposed to improve exploration, and a novel local search strategy is developed to enhance exploitation. Extensive experiments on both synthetic and real-world networks are carried out. Several state-of-the-art network-clustering approaches are compared with the proposed method. The experiments demonstrate that the proposed method is effective and promising.

Categories and Subject Descriptors
C.2.1 [Network Architecture and Design]: Distributed networks; E.1 [Distributed data structures]

General Terms:
Optimization Algorithm, Network Clustering Design

Keywords: Complex network, Network clustering, Community structure, Particle swarm optimization

Received: 11 September 2014, Revised 28 October 2014, Accepted 4 November 2014

1. Introduction

The networks are ubiquitous. Many intricate real-world systems can be modeled as networks, for example, complex collaboration networks [1] and the World-Wide Web [2]. Networks provide us with the enormous convenience of better communication with others and have led to unprecedented changes in our daily life.

We live in a complex world, because complex systems can be represented as networks, it is worthwhile analyzing the properties of networks as a way of understanding complex systems. Networks have many notable properties, such as the small-world property [3] and the scale-free property [4]. In recent years, network community structure property [5] has received much attention in diverse fields. In the academic domain, a network is generally denoted by a graph composed of nodes and edges. A community of a network is regarded as a subset of a graph. Often, communities are required to satisfy the condition that the nodes of a given community share similar features, while different communities have different node properties. The network community structure provides a microscopic perspective for understanding the functionality of a complex system. In the past few decades, a variety of methods have been
proposed to detect community structures in networks. A survey can be found in [6].

The detection of communities in a network can be regarded as a clustering problem, i.e., one of clustering a network into different groups. The essence of clustering is an optimization problem. Evolutionary algorithms (EAs), inspired by principles from biology, ethology, etc., are a class of intelligent optimization algorithms. They have been successful in solving a wide range of optimization problems. Therefore, it is natural to apply EAs to solve the network-clustering problem. Based on this idea, many single-objective and multiobjective EA-based network-clustering methods have been proposed. Pizzuti [7] proposed a genetic algorithm to discover communities in networks, using a community evaluation criterion called the community score. Gong et al. [8] proposed a memetic algorithm-based approach in which a hill-climbing-based local search tactic was suggested for improving the search performance of the algorithm. Cai et al. [9] introduced a clonal-selection algorithm-based method in which an effective local search strategy based on vertex neighborhoods was developed.

Many optimization problems involve multiple objectives. For the network-clustering problem, many multiobjective optimization models have been proposed. Pizzuti [10] presented a multiobjective genetic algorithm to uncover communities in networks. Recently, Gong et al. [11] proposed a multiobjective discrete particle swarm optimization algorithm for network clustering. Experiments have demonstrated that the proposed algorithm is very effective.

The main contribution of the work described by Gong et al. [11] was the proposed discrete particle swarm optimization (PSO) framework. PSO was proposed by Eberhart and Kennedy [12]. It is an intelligent optimization algorithm and is well known for its fast convergence and concise framework. PSO is attracting attention and has found broad applications in diverse domains [13].

In this paper, based on the basic framework of PSO proposed in [11], we suggest a single-objective PSO algorithm to solve the network-clustering problem. The proposed algorithm maximizes a well-known community quality evaluation index called modularity. Modularity was proposed by Girvan and Newman [14] in their divisive hierarchical clustering algorithm, in which modularity was used as a criterion to stop the division of a network into subnetworks. Many methods have been developed to optimize modularity, such as the simulated annealing-based method [15], the extremal optimization-based approach [16], and the spectral optimization-based technique [17]. However, their performances are generally not satisfying. It has been proved that optimizing modularity is an NP-hard problem. In this paper, to improve the performance of the proposed algorithm, a specially designed particle turbulence operator is proposed to augment the exploration of the algorithm. A local search tactic is developed to enhance the exploitation of the algorithm. Extensive experiments on both synthetic and real-world networks prove that the proposed algorithm is efficient and much faster than several state-of-the-art approaches.

The rest of this paper is organized as follows. Section 2 gives the relevant background, including a statement of the network-clustering problem and a brief introduction to PSO. Section 3 presents the proposed algorithm for network clustering in detail. Section 4 describes the experimental studies of the proposed method, and finally, the conclusions are summarized in Section 5.

2. Related Background

2.1 Network Clustering

A network is normally represented by a graph that is composed of a set of nodes and edges. The task of network clustering is to divide a network into different clusters based on certain principles. Each cluster is called a community.

Given a network represented by a graph denoted by $G = (V, E)$, where $V$ is the set of nodes and $E$ is the set of edges, let $A$ be the adjacency matrix of $G$ and $a_{ij}$ the $ij$th element of $A$. Given that $S$ is a subgraph (i.e., $S \subseteq G$), let $d_{in} = \sum_{i \in S, j \in S} a_{ij}$ and $d_{out} = \sum_{i \in S, j \notin S} a_{ij}$ be respectively the internal and external degrees of node $i$. Then, according to the definition given by Radicchi et al. [12], $S$ can be regarded as a community in a strong sense if

$$\forall i \in S, d_{in} > d_{out}$$

and $S$ is a community in a weak sense if

$$\sum_{i \in S} d_{in} > \sum_{i \notin S} d_{out}$$

However, the above definition only prescribes the condition that a community should satisfy. For a network-clustering problem, there needs to be a quantitative index to measure the quality of a network partition. In respect of this, Girvan and Newman [14] have introduced a modularity index. Modularity (normally denoted by $Q$) can be written as

$$Q = \frac{1}{2m} \sum \left( A_{ij} - \frac{k_i k_j}{2m} \right) \sigma (i, j),$$

where $m$ is the number of edges and $\sigma (i, j) = 1$ if nodes $i$ and $j$ are in the same community, otherwise $\sigma (i, j) = 0$. Normally, by assumption, we take it that the larger the value of $Q$, the more accurate is the partition.

2.2 Particle Swarm Optimization

The PSO algorithm was first introduced in 1995 by Eberhart and Kennedy [12]. It is a population-based stochastic optimization technique originally designed for continuous optimization problems. The inspiration for PSO originated from social behavior such as fish schooling and bird flocking. PSO has now become one of the most
3. Proposed Method for Network Clustering

3.1 General Framework
In this paper, a novel PSO algorithm is proposed for network clustering. In the proposed algorithm, a novel particle status update principle is defined, and, to improve exploration of the algorithm, a novel particle turbulence operation is developed. To enhance the exploitation, a local search strategy is introduced. This section will present a detailed description of the proposed algorithm.

The proposed PSO-based network-clustering algorithm works as follows:

**Input:**
- \( \text{pop} \): Particle swarm size;
- \( \text{gmax} \): Number of iterations;
- \( \omega \): Inertia weight, learning factors \( c_1 \) and \( c_2 \);
- \( \text{A} \): Network adjacency matrix.

**Output:** The network community structure.

**Step 1** Initialization
**Step 1.1** Initialize the population, i.e., initialize the position vectors \( X_1, X_2, \ldots, X_{\text{pop}} \) and initialize the velocity vectors \( V_1, V_2, \ldots, V_{\text{pop}} \);
**Step 1.2** Initialize the personal best position vectors \( P_1, P_2, \ldots, P_{\text{pop}} \) and the global best position vector \( G \);

**Step 2** Update: For \( i = 1, \ldots, \text{pop} \), do
**Step 2.1** Update the particle velocity vectors according to Equation (6);
**Step 2.2** Update the particle position vectors according to Equation (7);

**Step 2.3** Turbulence operation on the position vectors;

**Step 3** Local search: implement the local search procedure on the global best vector \( G \);

**Step 4** Stopping criterion: If the stopping criterion is satisfied, then stop and output. Otherwise, go to Step 2).

3.2 Particle Representation
A particle represents a solution to the optimization problem. For the network-clustering problem, the position vector of a particle is an integer permutation.

The adopted particle representation scheme is direct and easy to realize; besides, it no longer needs to prescribe the number of communities of a network, because it can automatically determine this unique interest.

![Figure 1. Graphical illustration of the particle representation scheme](image)

3.3 Particle Update Rules
Because the particle velocity and position vectors are all integer coded, the update rules in Eqs. (4) and (5) no longer fit the requirement of the recommendation problem. In this paper, we redefine the update rules in a discrete form. The redefined rules read

\[
V_i \leftarrow \text{sig}(\omega V_i + c_1 r_1 (P_i \oplus X_i) + c_2 r_2 (G \oplus X_i))
\]

\[
X_i \leftarrow X_i \oplus V_i
\]

where the operator \( \oplus \) is the XOR operation, and the \( \text{sig}(x) \) function works as follows:

\[
\text{sig}(x) = \begin{cases} 
1 & \text{if } \text{rand}(0, 1) < 1 / (1 + e^{-x}) \\
0 & \text{if } \text{rand}(0, 1) \geq 1 / (1 + e^{-x})
\end{cases}
\]

The \( \text{sig}(x) \) function maps the velocity vector into a binary code sequence; consequently, the operator \( \oplus \) is the key operation of the newly defined particle update rules. In this paper, considering the network topology and the features of the optimization objective function, we define the operation of \( \ominus \) in the following way:

\[
X_i \ominus V_i = \{x_i', x_{i2}', \ldots, x_{in}'\}
\]

In this equation, if an element of \( V_i \) is zero, \( x_i' = x_i \); otherwise, \( x_i' \) is calculated as follows:

...
\[ x_{ij}' = \arg \max Q(x_{ij} \leftarrow x_{ik} \mid k \in N_j) \]  

(10)

where \( N_j \) is the neighbor set of node \( j \).

Equation (10) returns the value \( x_{ij}' \) that can generate the largest modularity increment when the current element \( x_{ij} \) is replaced by this value. By defining the position update rule in this form, we can obtain a large value for the objective function, because each step of the updating always finds the community identifier that can lead to an increase in the modularity. It should be pointed out that when there are multiple \( x_{ik} \) that can generate the greatest increase in modularity, then we randomly choose one of them.

3.4 Particle Turbulence Operation

To preserve population diversity after the particle position has been updated, we have designed a turbulence operator.

Inspired by the social phenomenon that if the majority of a person’s friends share a common feature, then it is likely that the person will join them, the turbulence operator is designed to work in the following way.

We first decide whether it will undergo the operation with a possibility of 0.5; then, in the turbulence step, we update the position element with the dominated identifiers from its neighbors (the dominated identifiers are the identifiers that are in the majority). If there is more than one dominated identifier, we randomly choose one of them. The pseudocode of the operation is given in Algorithm 1.

Algorithm 1 Pseudocode of the turbulence operation.

for \( i = 0; i < \text{vertex}; i++ \)

(a) if rand(0, 1) < 0.5

\[
\begin{align*}
NL & [i] = \text{find} \left( \text{node}[i].\text{NeighborLabel} \right); \\
DL & [i] = \text{DominatedLabel} (NL);  \\
\text{node} [i]. \text{label} = DL (\text{floor} (\text{rand} (0, 1) * \text{size} (DL))); \\
\end{align*}
\]

(b) end if

end for

3.5 Particle Turbulence Operation

Algorithm 2 Pseudocode of the local search operation.

for \( i = 0; i < \text{vertex}; i++ \)

a) \( NL [i] = \text{find} \left( \text{node}[i].\text{NeighborLabel} \right); \)

b) \( \text{DiffL} [i] = \text{DifferntLabel} (NL); \)

c) for \( j = 0; j < \text{size} (\text{DiffL}); j++ \)

\[
\begin{align*}
\text{DQ} [i] & = \text{deltaQ} \left( \text{node}[i]. \text{label} \leftarrow \text{DiffL} [j] \right); \\
\end{align*}
\]

d) end for

e) if \( \text{max} (\text{DQ} []) > 0 \)

i. \( \text{node} [i]. \text{label} = \text{DiffL} [\text{max}]; \)

ii. break;

f) end if

e) end for

Because the global best particle plays the role of the leader, the local search step in our algorithm works only on this particle to guide the swarm to a promising region.

4. Experimental Studies

In this section, the proposed algorithm for network clustering will be tested on real-world networks. Several state-of-the-art network-clustering methods have been selected for comparison with the proposed approach. All the experiments have been performed on an Intel® Celeron® CPU 550 machine (3.2 GHz, 4 GB memory). The operating system is MS Windows 7 and the program compiler is VC++ 6.0.

4.1 Experimental settings

The proposed algorithm is denoted as PSO-net. The comparison methods are the genetic algorithm-based method GA-net [7], the multiobjective PSO-based method MODPSO [11], and the normalized cut (Ncut) approach proposed in [18]. For fair comparison, the population size \( \text{pop} \), the maximal algorithm iteration number \( \text{gmax} \), the crossover possibility \( \text{pc} \), and the mutation possibility \( \text{pm} \) are set to 50, 250, 0.9, and 0.1, respectively. The objective function used in GA-net has been changed to be the modularity. The learning factors \( c_1 \) and \( c_2 \) used in PSO-net are both set to 1.494, and the inertia weight \( \omega \) is set to 0.729.

To estimate the similarity between the true network partition and the detected one, the widely used normalized mutual information (NMI) metric [19] is adopted. Given two partitions \( A \) and \( B \) of a network into communities, let \( C \) be the confusion matrix, whose element \( C_{ij} \) is the number of nodes shared in common by community \( i \) in partition \( A \) and community \( j \) in partition \( B \). \( \text{NMI} (A, B) \) is then defined as

\[
\text{NMI} = \frac{-2 \sum_{i=1}^{CA} \sum_{j=1}^{CB} C_{ij} \log(C_{ij}/C_{i.}C_{.j})}{\sum_{i=1}^{CA} C_{i.} \log(C_{i.}/N) + \sum_{j=1}^{CB} C_{.j} \log(C_{.j}/N)}
\]

(11)

where \( CA \) and \( CB \) are the numbers of clusters in partitions \( A \) and \( B \), respectively; \( C_i \) and \( C_j \) are the sums of the elements of \( C \) in row \( i \) and in column \( j \), respectively; and \( N \) is the number of nodes. If \( A = B \), then \( \text{NMI} (A, B) = 1 \); if \( A \) and \( B \) are completely different, then \( \text{NMI} (A, B) = 0 \).

4.2 Experiments on Synthetic Networks

We first test our proposed algorithm on two kinds of synthetic networks whose true partitions are known. The first kind of synthetic networks are the so-called GN extended benchmark network proposed by Lancichinetti et al. [20]. A GN extended benchmark network is divided into four communities, with each community having 32 nodes. The average degree of each node is 16. Every node shares a fraction \( 1 - \gamma \) of links with the other nodes in its community and \( \gamma \) with the other nodes of the network. Here, \( \gamma \) is called the mixing parameter. The mixing parameter controls the portion of links within a community.
and outside it. In the experiments, we have generated 11 networks with values of γ ranging from 0.0 to 0.5. The second kind of synthetic networks are the LFR networks proposed in [20]. We run each of the algorithms 30 times and record the averaged NMI.

Figure 2 shows the experimental results for the GN extended benchmark networks. As can be seen, when γ is no bigger than 0.15, all the methods can obtain \( NMI = 1 \), i.e., they can all correctly figure out the true partitions of the networks. As γ increases, GA-net fails to partition the networks. We can see from the curves that the proposed algorithm outperforms the other methods in terms of NMI values. The GN extended benchmark networks are small. In the next step, we test the methods on the large LFR benchmark networks.

<table>
<thead>
<tr>
<th>Network</th>
<th>Ref.</th>
<th>Nodes</th>
<th>Edges</th>
<th>Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Karate</td>
<td>[21]</td>
<td>34</td>
<td>78</td>
<td>2</td>
</tr>
<tr>
<td>dolphin</td>
<td>[22]</td>
<td>62</td>
<td>159</td>
<td>2</td>
</tr>
<tr>
<td>SFI</td>
<td>[5]</td>
<td>118</td>
<td>200</td>
<td>—</td>
</tr>
<tr>
<td>netscience</td>
<td>[23]</td>
<td>1589</td>
<td>2742</td>
<td>—</td>
</tr>
</tbody>
</table>

Table 1. Parameters of the four real-world networks

Figure 2. Experimental results on the GN extended benchmark networks

Figure 3. Experimental results on the LFR benchmark networks

Figure 3 records the averaged NMI values over 30 runs of the algorithms on the 17 computer-generated networks. It is obvious that GA-net performs poorly on the LFR networks. The proposed PSO-net algorithm can still obtain the true network partitions when γ = 0.6. As γ increases, the network structures become chaotic, and none of the methods can correctly partition the networks.

The experiments on the two kinds of benchmark networks have demonstrated the effectiveness of the proposed PSO-net algorithm for network clustering. In the next step, we test the algorithms on some real-world networks.

4.3 Experiments on Real-world Networks

Four real-world networks have been chosen to test the proposed algorithm and the comparison algorithms. The parameters of the networks are shown in Table 1.

We run each of the algorithms 30 independent times and record the statistical results. In the experiments, the maximum and averaged values of the modularity and the NMI indices are recorded. We also use Wilcoxon's rank sum hypothesis test to perform a statistical analysis of the results. In the hypothesis test, the significance level is set as 0.05. The null hypothesis is that the means of two independent samples are the same. Tables 2 and 3 show the experimental results.

It can clearly be seen that PSO-net is very effective. Not only does PSO-net give generally higher modularity values, but it also takes less computational time. The small \( p \)-values indicate that our proposed algorithm PSO-net is significantly better than the comparison methods. Ncut is the fastest among the four algorithms, but it needs to preassign the cluster numbers of the networks, which makes it inconvenient to use.

In our experiments, we have observed that PSO-net not only gives better modularity values, but also yields meaningful community structures. Figures 4 and 5 show the ground truths and the obtained clustering results for the karate and dolphin networks.
<table>
<thead>
<tr>
<th>Data</th>
<th>Method</th>
<th>$Q_{\text{max}}$</th>
<th>$Q_{\text{avg}}$</th>
<th>$NMI_{\text{max}}$</th>
<th>$NMI_{\text{avg}}$</th>
<th>Time (s)</th>
<th>$p$-value</th>
<th>Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>karate</td>
<td>PSO-net</td>
<td>0.4198</td>
<td>0.4198</td>
<td>1</td>
<td>0.6881</td>
<td>2.1203E-2</td>
<td>—</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>GA-net</td>
<td>0.4198</td>
<td>0.4198</td>
<td>1</td>
<td>0.6873</td>
<td>3.7883E-1</td>
<td>0.5000</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>MODPSO</td>
<td>0.4198</td>
<td>0.4198</td>
<td>1</td>
<td>0.6873</td>
<td>1.3975</td>
<td>0.5000</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Ncut</td>
<td>0.4198</td>
<td>0.4198</td>
<td>1</td>
<td>0.6873</td>
<td>3.8282E-2</td>
<td>0.5000</td>
<td>4</td>
</tr>
<tr>
<td>dolphin</td>
<td>PSO-net</td>
<td>0.5285</td>
<td>0.5284</td>
<td>1</td>
<td>0.8935</td>
<td>1.7670E-1</td>
<td>—</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>GA-net</td>
<td>0.5285</td>
<td>0.5280</td>
<td>1</td>
<td>0.5892</td>
<td>3.7883E-1</td>
<td>0.1591</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>MODPSO</td>
<td>0.5268</td>
<td>0.5248</td>
<td>1</td>
<td>1</td>
<td>3.1713</td>
<td>0.0000</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Ncut</td>
<td>0.5068</td>
<td>0.5047</td>
<td>1</td>
<td>0.5048</td>
<td>9.3715E-3</td>
<td>0.0000</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 2. Experimental results on the karate and dolphin networks. For the Ncut method, the preassigned cluster numbers for the karate and dolphin networks are set to 4 and 2, respectively.

<table>
<thead>
<tr>
<th>Data</th>
<th>Method</th>
<th>$Q_{\text{max}}$</th>
<th>$Q_{\text{avg}}$</th>
<th>Time(s)</th>
<th>$p$-value</th>
<th>Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>SFI</td>
<td>PSO-net</td>
<td>0.7506</td>
<td>0.7506</td>
<td>3.3221E-2</td>
<td>—</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>GA-net</td>
<td>0.7506</td>
<td>0.7506</td>
<td>4.5563</td>
<td>0.3735</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>MODPSO</td>
<td>0.7484</td>
<td>0.7481</td>
<td>1.9742</td>
<td>0.0000</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Ncut</td>
<td>0.7484</td>
<td>0.7480</td>
<td>7.3621E-3</td>
<td>0.0000</td>
<td>7</td>
</tr>
<tr>
<td>netscience</td>
<td>PSO-net</td>
<td>0.9540</td>
<td>0.9512</td>
<td>5.3275E+1</td>
<td>—</td>
<td>331</td>
</tr>
<tr>
<td></td>
<td>GA-net</td>
<td>0.9086</td>
<td>0.9003</td>
<td>1.2165E+2</td>
<td>0.0000</td>
<td>279</td>
</tr>
<tr>
<td></td>
<td>MODPSO</td>
<td>0.9503</td>
<td>0.9493</td>
<td>6.3725E+1</td>
<td>0.0000</td>
<td>286</td>
</tr>
<tr>
<td></td>
<td>Ncut</td>
<td>0.9293</td>
<td>0.9268</td>
<td>5.5670E-1</td>
<td>0.0000</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 3. Experimental results of the SFI and netscience networks. For the Ncut method, the preassigned cluster numbers for the SFI and netscience networks are set to 7 and 100, respectively.

Figure 4. Clustering result for the karate network. (a) Ground truth. (b) Detected result
Figure 5. Clustering result of the dolphin network. (a) Ground truth. (b) Detected one

Figure 6. Clustering result of the SFI network
The karate network represents the relationships between the 34 members of a karate club. This network was originally divided into two parts. The proposed algorithm PSO-net subdivides one of the original two clusters into three smaller clusters; thus, PSO-net gives four clusters, as shown in Figure 4(b).

The dolphin network represents the social behavior among 62 bottlenose dolphins living in Doubtful Sound, New Zealand. This network was originally divided into two groups: the female group and the male group. For this network, PSO-net obtains five clusters, as shown in Figure 5(b). It can be seen from Figure 5 that the clustering result of PSO-net is just a subdivision of the real network partition. For the karate and dolphin networks, the proposed algorithm never misplaces any nodes.

The SFI network represents the relationships between 118 scientists in residence at the Santa Fe Institute. The real partition of this network is unknown. PSO-net has divided this network into eight clusters, as shown in Figure 6. The clustering result for this network makes sense. Each group is centered around the research interests of leading members.

On the big netscience network, PSO-net has obtained a modularity value of 0.9540. The corresponding cluster number is 331. Technically, the clustering result is valid in terms of the modularity index.

The above experiments have proved the effectiveness of the proposed PSO-net algorithm. The turbulence operator and the local search strategy account for the good performance of the algorithm. To show their effects, we have performed an experiment on the netscience network. We ran the proposed algorithm with arbitrary combinations of the two tactics and recorded the convergence curves. The result is displayed in Figure 7, from which it is clear that the proposed turbulence and local search strategies help to improve the performance of the algorithm. The turbulence operator preserves population diversity, which enhances the exploration of the algorithm, and the local search operator augments the exploitation of the algorithm, which benefits the search for optimal solutions.

5. Conclusion

In this paper, a novel discrete PSO algorithm has been proposed for complex network clustering. The proposed algorithm maximizes a widely used index called modularity. In the algorithm, the particle position update rule has been redesigned so that a position label is updated with the neighbor label that generates the largest increase in modularity. The newly defined rule drives the particles to a more promising region. A novel turbulence operation is suggested for improving the exploration of the algorithm. This operation makes full use of the network linkage relationships to direct the search process. A local search strategy is developed to enhance the exploitation of the algorithm. The local search procedure is carried out on the leader particle. To validate the performance of the proposed algorithm, extensive experiments have been performed on both synthetic and real-world networks. We have compared the proposed algorithm with three state-of-the-art methods. All the experiments demonstrate that the proposed algorithm is effective and promising.

Acknowledgements

The authors are grateful for the support provided by the Scientific Research Program of the Department of Education of Shaanxi Province (Grant No. 14JK1859) and the Science Project of Yulin City (Grant Nos. Gy13-15 and Ny13-10), the Department of Science & Technology of
References


