Physica A 449 (2016) 336–347

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

A multi-agent genetic algorithm for community detection in complex networks

Zhangtao Li, Jing Liu *

Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an 710071, China

HIGHLIGHTS

- The multi-agent system integrating with genetic algorithm is first used to detect communities in complex networks.
- A series of effective neighborhood-based operators are designed.
- The good performance of the new algorithm is validated by various networks and the systematic comparisons with two representative algorithms.
- The new algorithm can detect communities with high speed, accuracy and stability.

ARTICLE INFO

Article history:
Received 11 May 2015
Received in revised form 17 November 2015
Available online 7 January 2016

Keywords:
Community detection
Multi-agent systems
Neighborhood-based operators
Modularity
Genetic algorithm

ABSTRACT

Complex networks are popularly used to represent a lot of practical systems in the domains of biology and sociology, and the structure of community is one of the most important network attributes which has received an enormous amount of attention. Community detection is the process of discovering the community structure hidden in complex networks, and modularity Q is one of the best known quality functions measuring the quality of communities of networks. In this paper, a multi-agent genetic algorithm, named as MAGA-Net, is proposed to optimize modularity value for the community detection. An agent, coded by a division of a network, represents a candidate solution. All agents live in a lattice-like environment, with each agent fixed on a lattice point. A series of operators are designed, namely split and merging based neighborhood competition operator, hybrid neighborhood crossover, adaptive mutation and self-learning operator, to increase modularity value. In the experiments, the performance of MAGA-Net is validated on both well-known real-world benchmark networks and large-scale synthetic LFR networks with 5000 nodes. The systematic comparisons with GA-Net and Meme-Net show that MAGA-Net outperforms these two algorithms, and can detect communities with high speed, accuracy and stability.

1. Introduction

In recent years, with the development of information technique, complex networks have been used in many domains, such as web, power grids, sensor networks, biological networks and social networks [1]. In these applications, networks can be modeled as graphs where nodes represent objects and edges represent relationships between objects. The community structure, as one of the most important properties of complex networks, has received a lot of attention [2,3]. A network has

* Corresponding author. Tel.: +86 29 88202661.
E-mail addresses: neouma@mail.xidian.edu.cn, neouma@163.com (J. Liu).

http://dx.doi.org/10.1016/j.physa.2015.12.126
0378-4371/

clear community structure if the nodes inside communities are densely connected, while the nodes between communities are loosely connected.

A good community structure implies important messages about relationships between network function and topology. Many community detection algorithms have been proposed, where modularity-based algorithms are the most popular ones [1]. The modularity Q is a quality function measuring the quality of partitions of networks proposed by Newman and Girvan [3]. An effective way to solve the community detection problems is to find the optimal partitions with high modularity [4].

Community detection is a NP-hard problem that traditional optimization methods cannot solve it effectively [5]. Genetic algorithms (GAs) play a very important role in solving this kind of complex problems. But the major problem of GAs is that they may be trapped in local optima and they have difficulties in addressing large-scale problems effectively. In our previous work, a multi-agent genetic algorithm (MAGA) was proposed in Ref. [6] to solve large scale global numerical optimization problems, which achieved a good result with the dimensions increasing from 20 to 10 000. MAGA was a combination of multi-agent systems and GAs and the experimental results indicated that this kind of combination was effective in solving large-scale problems. Recently, multi-agent systems have been integrated with evolutionary algorithms to solve constraint satisfaction problems and combinatorial optimization problems with satisfactory results [7,8]. Moreover, an improved agent based model (iABM) is applied to dynamic airspace sectorization [9]. A multi-objective evolutionary algorithm is used to optimize this model with a good performance. All the above evidences show that MAGA is suitable for handling large-scale complex problems.

In order to cluster large-scale networks with high accuracy, in this paper, a multi-agent genetic algorithm, named as MAGA-Net, is proposed to optimize modularity value for the community detection. Based on the locus-based adjacency representation, a split and merging based neighborhood competition operator is designed. To make full use of the two-point and uniform crossover operators, we design a hybrid neighborhood crossover operator. Moreover, we use an adaptive mutation operator from Ref. [10] to effectively explore the search space when the number of generations grows without improvement. At last, we conduct the self-learning operator on the best sl number of agents in each generation to further increase their energy.

To validate the performance of MAGA-Net, in the experiments, both well-known real-world benchmark networks and large-scale synthetic LFR networks are used. The results show that MAGA-Net has the ability to find correct partitions of large-scale networks with 5000 nodes. The systematic comparisons with GA-Net and Meme-Net show that MAGA-Net outperforms these two algorithms, and can detect communities with high speed, accuracy, and stability.

The rest of the paper is organized as follows. We review related work on community detection in Section 2. The details of MAGA-Net are described in Section 3. The experiments on well-known real-world benchmark networks and large-scale synthetic LFR networks are performed in Section 4. Finally, conclusions are given in Section 5.

2. Related work

The key objective of community detection problems is to find the hidden communities of networks. Various methods have been proposed to give reasonable partitions of networks [11–14]. The method proposed in this paper is a kind of evolutionary algorithms (EAs). So, in this section, we will give a brief introduction of existing EAs for community detection.

EAs are effective methods to deal with problems in complex networks. In our previous work, a memetic algorithm for enhancing the robustness of scale-free networks against malicious attacks was proposed by Zhou et al. in Ref. [15]. For detecting communities, the evolution usually starts from a random set of individuals, and then every individual in the population is evaluated, next, a series of evolutionary operators are conducted to form a new population which will be used in the next generation. Repeat the above steps until termination has reached. As a result, only individuals with large fitness survive.

Bui et al. in Ref. [16] proposed a genetic algorithm for graph partition with a schema preprocessing phase to improve GAs' space searching capability. Talbi et al. in Ref. [17] proposed a parallel genetic algorithm for the graph partition problem which showed a superlinear speed-up. Tasgin et al. in Ref. [18] used a genetic algorithm to detect communities based on modularity. Pizzuti in Ref. [19] proposed a genetic algorithm for community detection named as GA-Net using the locus-based adjacency representation and uniform crossover. It was efficient in reducing the invalid search when only the actual correlations of all nodes were considered in each operator. A new collaborative evolutionary algorithm was proposed by Gog et al. in Ref. [20] which was based on information sharing mechanism between individuals in a population. Gong et al. in Ref. [21] proposed a memetic algorithm to optimize the modularity density for community detection. A local search procedure named as high-climbing strategy was added to genetic algorithm which performed better than traditional GAs. Gong et al. in Ref. [22] also proposed a multi-objective evolutionary algorithm based on decomposition which optimized two contradictory objectives, negative ratio association and ratio cut. Liu et al. in Ref. [23] designed a representation method which could represent separated and overlapping communities at the same time and proposed a multi-objective evolutionary algorithm to solve community detection problems under the framework of NSGA-II. Li et al. in Ref. [24] made a comparative analysis of evolutionary and memetic algorithms for community detection from signed social networks. Zeng et al. in Ref. [25] and Liu et al. in Ref. [26] both proposed a multi-objective evolutionary algorithm for community detection from signed social networks. While the algorithm in Ref. [26] was based on similarity and a direct and indirect combined representation was designed to detect both separated and overlapping communities.

3. MAGA-Net

3.1. Network community and modularity definition

A complex network *Ne* can be modeled as a graph $G = (V, E)$, where V represents all nodes in G while E represents all edges between nodes. M is an adjacency matrix and H represents an adjacency list. If there exists an edge between nodes i and j, $M_{ij} = 1$; otherwise, $M_{ij} = 0$. H is used to store the neighbors (node i is a neighbor of node j only when there is a link between them) of each node. As mentioned above, a network is deemed to have community structure only if the vertices have dense connections inside communities and sparse connections between communities. Let k_i represent the degree (the total number of links connected with i) of node i, C be a community of network *Ne* to which node i belongs. Then, for node i in community C, ${k_i}^{in} = \sum_{i,j\in C} M_{ij}$ and ${k_i}^{out} = \sum_{i\in C, j\notin C} M_{ij}$ stand for the internal and external degree. C has strong community structure when it satisfies:

$$\forall i \in C, \quad k_i^{in} > k_i^{out} \tag{1}$$

C has weak community structure if

$$\sum_{i\in C} k_i^{in} > \sum_{i\in C} k_i^{out}. \tag{2}$$

As pointed by Newman and Girvan in Ref. [3], modularity Q was defined as a criterion to measure the quality of a division of a network and it was widely used to describe the significance level of community structure. It is a quality assessment that reveals the difference between the detected communities and a random graph. MAGA-Net takes modularity optimization as its fitness function, and evaluates individuals in the population to increase the modularity values as much as possible. The modularity Q can be defined as follows:

$$Q = \sum_{k=1}^{s} \left[\frac{l_k}{L} - \left(\frac{d_k}{2L} \right)^2 \right] \tag{3}$$

where s is the total number of communities, L is the summation of all edges in the network, l_k is the number of edges inside community k and d_k is the summation of degrees of all nodes inside community k.

3.2. Agents for community detection

Agents have the ability to perceive and react against the environment. They have a very wide range of meanings depending on the problem we want to solve. In general, an agent has four properties [27,28]: (1) lives and acts in an environment; (2) able to sense its local surroundings; (3) driven by a specific purpose; (4) owns some reactive behaviors. All agents compete or work together to achieve their common goals. In MAGA-Net, an agent is defined as a division of a network.

Definition 1. An agent is a candidate solution of the community detection problem we need to solve, while the value of its energy equals the modularity defined in Eq. (3).

Definition 2. All agents live in a lattice-like environment L called agent lattice. Each agent is fixed on one point of this kind of lattice and can only exchange information with its neighbors. The number of agents is $L_{size} \times L_{size}$ and the agent lattice can be defined as the form in Fig. 1.

Suppose an agent locates at (m, n), $m, n = 1, 2, \ldots, L_{size}$, then its neighbors can be defined as $neighbors_{m,n}$ in Eqs. (4) and (5). We can see $L_{m,n}$'s neighbors clearly in Fig. 2.

$$neighbors_{m,n} = \left\{ L_{m',n}, L_{m,n'}, L_{m,n''}, L_{m'',n} \right\} \tag{4}$$

$$\begin{aligned} m' &= \begin{cases} m-1 & m \neq 1 \\ L_{size} & m = 1, \end{cases} \qquad n' = \begin{cases} n-1 & n \neq 1 \\ L_{size} & n = 1, \end{cases} \\ m'' &= \begin{cases} m+1 & m \neq L_{size} \\ 1 & m = L_{size}, \end{cases} \qquad n'' = \begin{cases} n+1 & n \neq L_{size} \\ 1 & n = L_{size}. \end{cases} \end{aligned} \tag{5}$$

3.3. Representation and initialization of agents

In MAGA-Net, we use the locus-based adjacency representation proposed in Ref. [29] to represent agents. In this graph based representation, an agent $L_{m,n}$ consists of N (number of nodes in a network) genes expressed as $L_{m,n}$ =

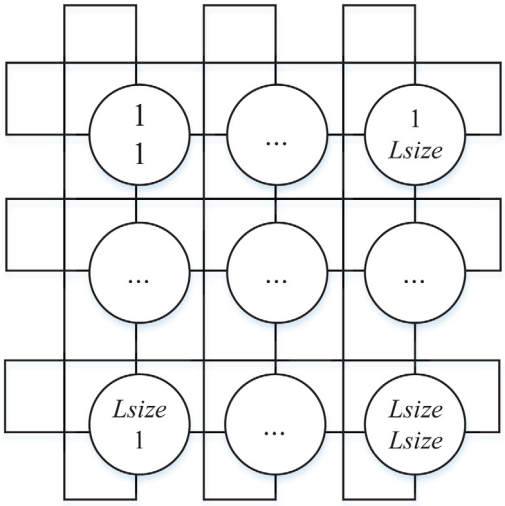


Fig. 1. Model of the agent lattice.

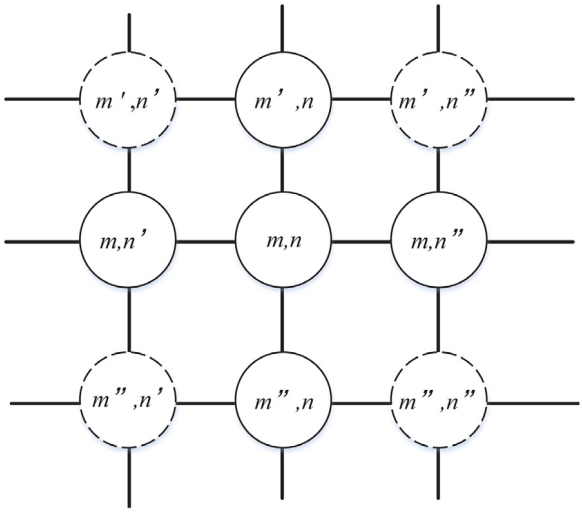


Fig. 2. The neighborhood of an agent.

$[g_1, g_2, g_3, \ldots, g_N]$ and each gene can adopt allele value k in the range $\{1, \ldots, N\}$. A gene and its allele all stand for nodes and k is an allele of g_i only when there is an edge between nodes i and k. In this representation, the ith gene's value $k\,(g_i = k)$ indicates nodes i and k are in the same community. A decoding process needs to identify all the components of a network and the nodes belonging to one component are assigned to the same community. The decoding step was proved to be done in linear time [30]. The main idea of this representation is vividly illustrated in Fig. 3.

We use locus-based adjacency representation for three reasons. One is that the number of communities are generated automatically without the need to specify a value beforehand; the second one is that locus-based adjacency representation can avoid isolated nodes in a partition by which we can initialize a relatively good population to speed up the convergence; the third one is that it restricts the possible solution space and reduces the invalid search.

The initialization procedure is quite simple. We only need to give an initial value to each gene by randomly selecting an allele from its adjacency of all agents. This operation is fast and makes full use of the connection relationships of all nodes. We can get a relatively good result from the beginning, but it is still far from being optimal.

3.4. Genetic operators of agents

In MAGA-Net, we have designed four genetic operators: split and merging based neighborhood competition operator, hybrid neighborhood crossover, adaptive mutation and self-learning operator. In order to gain more energy, on one hand, all agents compete or cooperate with their neighbors through the former two operators; on the other hand, each agent has

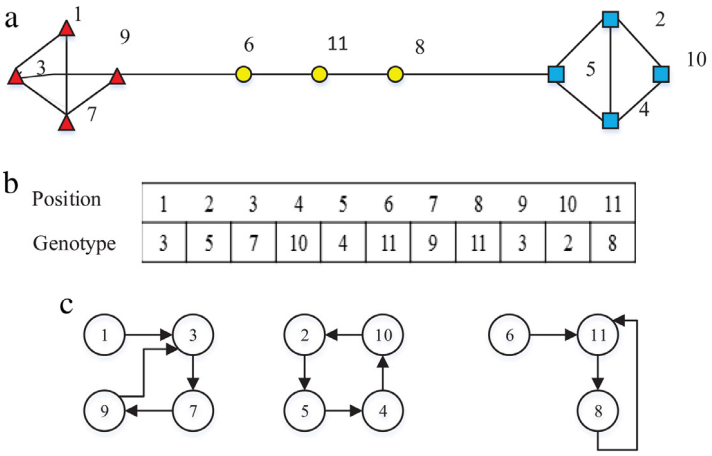


Fig. 3. (a) A graph with three communities; (b) the locus-based representation of an individual; (c) the partition of the graph based on decoding the individual in (b).

ability to use knowledge of its neighbors or itself to get a further study by using the latter two operators. All the operators are performed on $L_{m,n} = [g_1, g_2, g_3, \ldots, g_N]$. $Max_{m,n} = [h_1, h_2, h_3, \ldots, h_N]$ is $L_{m,n}$'s neighbor with maximum energy. The detailed descriptions of the four operators are shown as follows.

Split and merging based neighborhood competition operator: Considering the process of dividing a network using locus-based adjacency representation, we can clearly make out that each node has one and only one link towards its adjacency and there is one and only one loop in each community from Fig. 3.

Starting from this point, we use $u(0, 1)$ to represent a uniformly distributed random value in the interval [0, 1]. If $u(0, 1) \geq 0.5$, we randomly select a gene g_i of an agent as the target gene and transform its value to one of its alleles k in adjacency list when their community labels are different. We set s_1 as the community label of g_i and s_2 as the allele k's label, then $s_1 \neq s_2$. There are two possible outcomes, one is that communities s_1 and s_2 merge together to form a larger new community when the target gene selected is inside the loop of the community to which it belongs; the other is that community s_1 splits into two small communities and one of them without a loop combines with the community s_2, thus two communities are generated, one is smaller and the other is larger. This occurs when the target gene is out of the loop of the community to which it belongs; If $u(0, 1) < 0.5$, we just transform the value of the selected gene to an arbitrary allele of its adjacency list.

Using the above idea, we design a split and merging based neighborhood competition operator. For each agent $L_{m,n}$ on the lattice, do a comparison between agent $L_{m,n}$ and $Max_{m,n}$. If $Energy(L_{m,n}) > Energy(Max_{m,n})$, $L_{m,n}$ is the optimal one and can survive; otherwise, agent $L_{m,n}$ will be replaced by $Max_{m,n}$ with the strategy we proposed.

Strategy: Firstly, assign $Max_{m,n}$ to $New = [r_1, r_2, r_3, \ldots, r_N]$. Then, if $u(0, 1) \geq 0.5$, randomly select a gene r_i in New and replace it with one of its alleles k in adjacency list under the precondition of nodes i and k are in different communities; if $u(0, 1) < 0.5$, just select a gene r_i and change its value to one of its neighbors in adjacency list. Finally, $L_{m,n}$ is replaced by New.

Hybrid neighborhood crossover: Traditionally, a two-point crossover operator is conducted on two chromosomes, randomly select two crossing points, and then exchange the genes of the two chromosomes between the two points. Being different from the two-point crossover operator, the uniform crossover operator exchanges each gene of the two chromosomes with the same probability. These two crossover operators are both easy to implement and can be applied to MAGA-Net because of the flexibility of the locus-based adjacency representation. But the main disadvantage of these two operators is that their recombination is random and we cannot guarantee to generate offspring better than the parents.

In our algorithm, we design a hybrid neighborhood crossover operator. An agent $L_{m,n}$ on the lattice will only cross with $Max_{m,n}$ so as to obtain useful information from its neighbors and avoid random recombination. Moreover, we mix two-point crossover and uniform crossover operators together to provide more possibility of changes. In order to protect good patterns, we will not change $Max_{m,n}$.

For each agent $L_{m,n}$, if $u(0, 1) < p_c$ and $Energy(L_{m,n}) < Energy(Max_{m,n})$, we will replace $L_{m,n}$ by mixing $L_{m,n}$ and $Max_{m,n}$ with two strategies decided by p_s. That is, if $u(0, 1) < p_s$, strategy 1 is performed; otherwise, strategy 2 is selected.

Strategy 1: Randomly select two points k_1 and k_2, if $u(0, 1) < 0.5$, assign genes between position k_1 and k_2 of $Max_{m,n}$ to $L_{m,n}$; otherwise, the rest genes are assigned to $L_{m,n}$.

Strategy 2: The general uniform crossover operator is conducted on $Max_{m,n}$ and $L_{m,n}$, then the newly generated agent will replace $L_{m,n}$.

Algorithm 1: Self-learning Operator

Input:
sL_t: the agent lattice at the tth generation of sL;
sN_s: the maximum number of generations without improvement;
$L_{m,n}$: an agent in L to conduct the self-learning operator;
$sBest^t$: the best agent in sL_0, sL_1, ..., sL_t;
$sCBest^t$: the best agent in sL_t;
Mutation probability: ${}_sP_m$;
Output:
$L_{m,n} \leftarrow sBest^t$;
Learning($L_{m,n}$)$\leftarrow$ *False*;

$t \leftarrow 0$;
$n \leftarrow 0$;
$sL_0 \leftarrow$ initialize sL using the neighbor-based-mutation-operator according to Eq. (7) with probability ${}_sP_m$ and update $sBest^0$;
while $(n < sN_s)$ **do**
  $t \leftarrow t + 1$;
  $sL_t \leftarrow$ conduct the split-and-merging-based-neighborhood-competition-operator on sL_t;
  Update $sCBest^t$;
  if $(Energy(sCBest^t) > Energy(sBest^{t-1}))$ **then**
    $n \leftarrow 0$;
    $sBest^t \leftarrow sCBest^t$;
  else
    $n \leftarrow n + 1$;
    $sBest^t \leftarrow sBest^{t-1}$;
    $sCBest^t \leftarrow sBest^t$;
  end
end

Adaptive mutation: In order to avoid the useless searches over the search space and keep the diversity of the population, we apply neighbor-based mutation operator proposed in Ref. [31] to our algorithm. In this operator, for each gene of an agent, if $u(0, 1) < p_m$, then the gene's value is changed to the allele of one of its neighbors.

An adaptive mutation operator changing p_m to find a better result proposed in Ref. [10] is also adopted in this paper. The mutation probability will increase with the growing number of generations when no improvement has reached. Higher p_m represents more changes of genes in each agent which makes it more likely to jump out of local optima. N_s is the termination criterion and our algorithm will stop when it has run N_s generations without improvement. We set t as the number of generations without improvement, then $p_m{}'$ is calculated as follows:

$$p_m{}' = (t/N_S + 1)p_m. \tag{6}$$

Self-learning operator: An agent can use its knowledge to get more energy, so a local search method is essentially required to conduct on excellent agents. We generate a small-scale agent lattice sL with size $sL_{size} \times sL_{size}$ based on Eq. (7). $L_{m,n}$ is the agent that will be conducted the self-learning operator.

$$sL = \begin{cases} L_{m,n} & m' = 1, \quad n' = 1 \\ sL_{m',n'} & \text{otherwise} \end{cases} \tag{7}$$

where $sL_{m',n'}$ is generated by conducting neighbor-based mutation operator on $L_{m,n}$ with probability ${}_sp_m$. The split and merging based neighborhood competition operator is conducted on sL in each generation. sN_s is the termination criterion which indicates that the self-learning operator will stop when it runs sN_s generations without improvement. At last, the best agent with maximum energy in this process will replace $L_{m,n}$. We conduct the self-learning operator on the best sl number of agents in L to further increase their energy. Algorithm 1 summarizes self-learning operator more clearly.

3.5. Implementation of MAGA-Net

In the above sections, we have introduced the definition of network community, the multi-agent systems and all the operators we designed and adopted in detail. In this section, we give the framework of the proposed MAGA-Net in Algorithm 2. The split and merging based neighborhood competition operator is used to eliminate agents with lower energy (smaller modularity value) from the lattice firstly. Then, hybrid neighborhood crossover and adaptive mutation operators

Algorithm 2: MAGA-Net

Input:
L_t: the agent lattice at the tth generation of L;
sl: the number of agents carried out self-learning operator;
$Best^t$: the best agent in $L_0, L_1, \ldots, L_t$;
$CBest^t[sl]$: the best sl agents in L_t;
$CBest^t$: the best agent in L_t;
Crossover probability: P_c;
Mutation probability: P_m;
P_s: the probability of deciding which strategy will be chosen in hybrid neighborhood crossover operator;
N_s: the maximum number of generations without improvement;
Output:
Transform the optimal agent in L_t into a partition solution and output;

```
t ← 0;
n ← 0;
L_0 ← initialize the population by locus based adjacency representation, assign the Learning labels of L_0 as True and
update Best^0;
while (n < N_s) do
    t ← t + 1;
    L_t ← conduct the split-and-merging-based-neighborhood-competition-operator on L_t and update Learning
    labels of L_t;
    L_t ← conduct the hybrid-neighborhood-crossover-operator on L_t with probabilities P_c and P_s and update
    Learning labels of L_t;
    L_t ← conduct adaptive-mutation-operator on L_t with probability P_m and update Learning labels of L_t;
    CBest^t[sl] ← Finding the best sl agents in L_t;
    for i ← 1 to sl do
        if Learning(CBest^t[i]==True) then
            Conduct self-learning-operator on CBest^t[i]
        end
    end
    Update CBest^t;
    if (Energy(CBest^t) > Energy(Best^{t-1})) then
        n ← 0;
        Best^t ← CBest^t;
    else
        n ← n + 1;
        Best^t ← Best^{t-1};
        CBest^t ← Best^t;
    end
end
```

are conducted on L to effectively explore the search space with probabilities p_c and p_m and maintain the diversity of the population. Next, self-learning operator is performed on the best sl number of agents in L, which plays an irreplaceable role in improving the performance of MAGA-Net. Learning property of an agent decides that whether the self-learning operator can conduct on it. If the learning label of an agent is *True*, it means the self-learning operator can conduct on this agent; otherwise, not. When there is a change of an agent caused by the genetic operators, the learning label of this agent will be assigned as *True*, by which it can regain the opportunity of self-learning.

4. Experimental results

In this section, four well-known real-world networks and large-scale synthetic LFR networks are used to validate the performance of MAGA-Net. We make systematic comparisons between MAGA-Net and two representative algorithms, namely GA-Net [19] and Meme-Net [21]. All the experiments are performed on a machine with 3.2 GHz CPU and 4 GB of memory. Both MAGA-Net and GA-Net are implemented in Microsoft Visual Studio 2012, while Meme-Net is run in MATLAB. All experiments of MAGA-Net are implemented under the same parameter setting shown in Table 1.

Normalized mutual information (NMI) is used to evaluate the results [32]. NMI is a similarity measure estimating the similarity between the detected partitions and the true ones. Suppose A and B are partitions of a network, and c_A represents

Table 1
Parameter setting.

L_{size}	sL_{size}	P_c	P_m	P_s	${}_sP_m$	sl	N_s	sN_s
5	3	0.6	0.05	0.5	0.02	3	10	50

Table 2
The number of evaluations used by MAGA-Net, Meme-Net and GA-Net.

Graph	Karate	Dolphins	Polbooks	Football
Evaluation	3000	5000	6000	8000

Table 3
The comparisons in terms of modularity Q (Q_{Avg}, Q_{Max}, Q_{Std}) and p-value on the four real-world networks.

Network	BKR	GA-Net				Meme-Net				MAGA-Net			Times
		Q_{Avg}	Q_{Max}	Q_{Std}	p-value	Q_{Avg}	Q_{Max}	Q_{Std}	p-value	Q_{Avg}	Q_{Max}	Q_{Std}	(s)
Karate	0.420	0.3741	0.4198	0.0767	0.0020	0.4083	0.4198	0.0134	0.0000	0.4194	**0.4198**	0.0019	0.0202
Dolphins	0.529	0.4928	0.5227	0.0119	0.0000	0.4273	0.5025	0.3050	0.0000	0.5271	**0.5286**	0.0007	0.0733
Polbooks	0.527	0.4871	0.5212	0.0369	0.0000	0.4436	0.5139	0.0216	0.0000	0.5270	**0.5273**	0.0001	0.2680
Football	0.605	0.5020	0.5561	0.0237	0.0000	0.4904	0.5492	0.0233	0.0000	0.6020	**0.6046**	0.0026	0.3776

the number of communities in A while c_B denotes that of B. D is a confusion matrix, and $D_{i,j}$ stands for the number of nodes in community i of A that also appear in community j of B. N is the number of elements. $D_{i.}$ is the sum over row i of D while $D_{.j}$ is the sum of elements in column j. The definition of $NMI(A, B)$ is shown as:

$$NMI(A,B) = \frac{-2\sum_{i=1}^{c_A}\sum_{j=1}^{c_B} D_{ij}\log\left(\frac{D_{ij}N}{D_{i.}D_{.j}}\right)}{\sum_{i=1}^{c_A} D_{i.}\log\left(\frac{D_{i.}}{N}\right) + \sum_{j=1}^{c_B} D_{.j}\log\left(\frac{D_{.j}}{N}\right)}. \tag{8}$$

4.1. Experiments on real-world networks

In this section, we apply MAGA-Net on four well-known real-world networks: the Karate club network [33], Dolphins network [34], Political books network [35], and College football network [2]. In order to make reasonable comparisons of MAGA-Net with GA-Net and Meme-Net, we set the same number of evaluations for them in Table 2.

In addition, a statistical test (Welch's t-test) is performed over the experimental results to make comparisons from a statistical standpoint. Welch's t-test is an adaptation of Student's t-test and is reliable when the two samples are independent and have unequal variances. Two modularity sets X and Y are used as input and the significance level α is set as 0.05. The output p-value is defined as the probability of obtaining a result equal to or more extreme than what was actually observed under the premise that the null hypothesis is true. The null hypothesis can be rejected at the 5% significance level when $\alpha \geq p$.

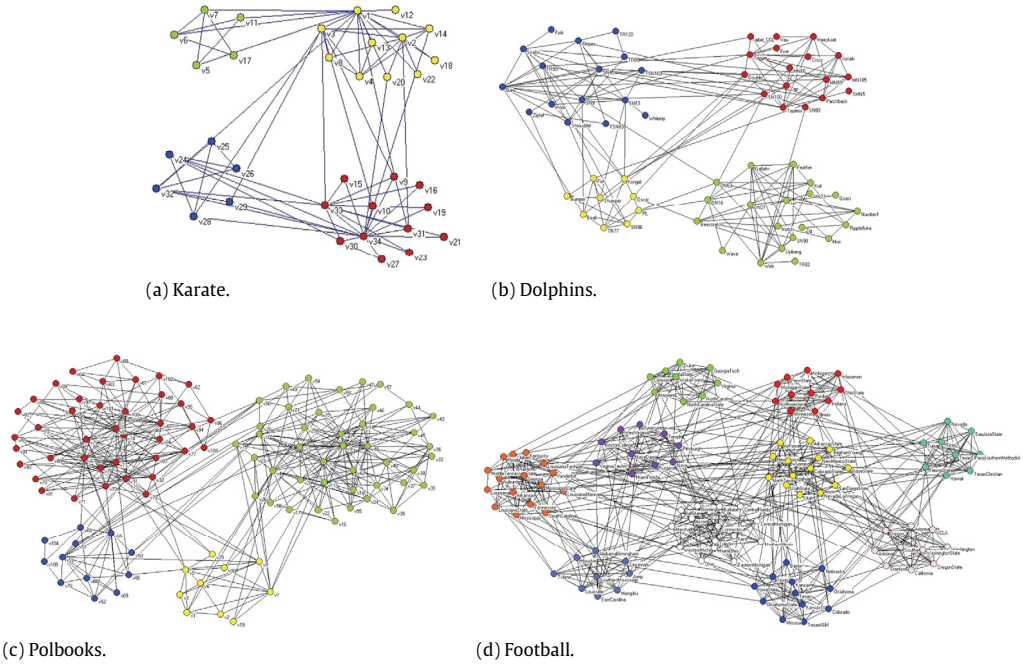
We run the three algorithms for 30 times independently on each network, and the results are shown in Table 3. BKR is the best known results of the four networks collected from published algorithms which achieved the best results and the sources of BKR are in Refs. [36–40]. We record the average, maximum and standard deviations (Q_{Avg}, Q_{Max}, Q_{Std}) of modularity Q, the p-value and the times taken by MAGA-Net.

As we can see from Table 3, MAGA-Net outperforms the two other algorithms in all networks. Under the same number of evaluations, MAGA-Net can quickly converge to the optima with a good stability. The small p-values listed in Table 3 indicate that MAGA-Net is better than Meme-Net and GA-Net on the four graphs with known ground truths. Moreover, compared to BKR, MAGA-Net has the ability to reach all the best known results with approximating as for the values of BKR are approximations themselves. Fig. 4 presents the clear community structure found by MAGA-Net for the four famous real-world networks.

To further test the performance of MAGA-Net, we calculate the NMI to measure the similarity of our partitions and the real ones. We still make comparisons with the two other methods and the results are shown in Fig. 5. The partitions obtained by MAGA-Net are more similar to the real ones than those obtained by the two other algorithms.

4.2. Experiments on synthetic networks

In order to verify the superior performance of MAGA-Net, we apply MAGA-Net on synthetic LFR benchmark graphs with scales of 1000 and 5000 nodes. LFR networks was proposed by Lancichinetti et al. [41], which are suitable for systematically



(a) Karate. (b) Dolphins.

(c) Polbooks. (d) Football.

Fig. 4. Community structure found by MAGA-Net of the four well-known benchmark networks. (a) Karate club network, $Q = 0.4198$. (b) Dolphins network, $Q = 0.5286$. (c) Political books network, $Q = 0.5273$. (d) College football network, Q=0.6046.

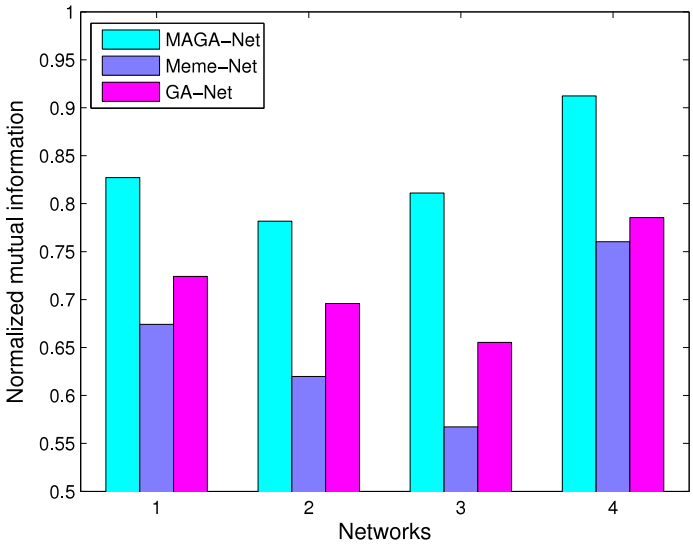


Fig. 5. The comparisons of average values of *NMI* obtained by MAGA-Net, Meme-Net and GA-Net on the four benchmark networks. (1) Karate club network. (2) Dolphins network. (3) Political books network. (4) College football network.

measuring the property of an algorithm. The average degree k and the maximum degree $k_{\max}$ are set to 20 and 50 in networks with 1000 nodes, while in networks with 5000 nodes, k and $k_{\max}$ are set to 40 and 100, respectively. The variable u is the mixing parameter ranging from 0.1 to 0.6, which is the proportion of edges besides communities. The communities become harder to detect as the value of u grows.

We generate 6 different networks as u ranging from 0.1 to 0.6 with sizes 1000 and 5000, respectively. *NMI* is used to estimate the similarity between the detected communities and the true ones. We run the three algorithms for 10 times independently, and the comparisons of average values of *NMI* are shown in Figs. 6 and 7.

As shown in Fig. 6, MAGA-Net has a good performance in the LFR networks with 1000 nodes. The value of *NMI* changes from 1 to 0.8 with u ranging from 0.1 to 0.6, which implies that the communities detected by MAGA-Net have a high sim-

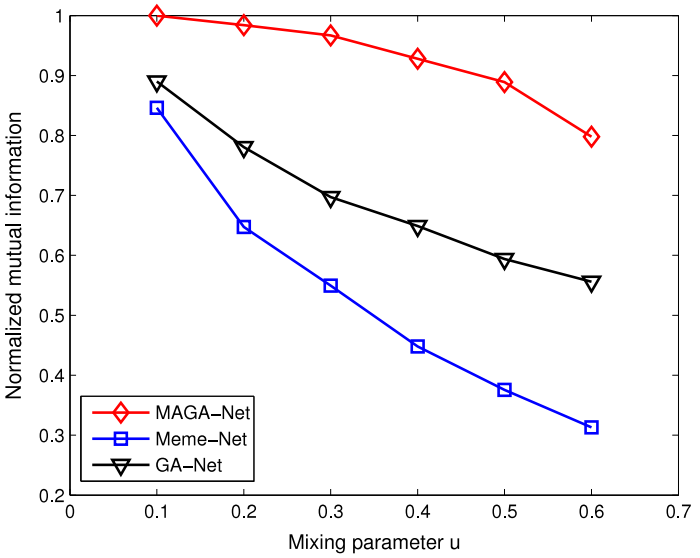


Fig. 6. The comparisons of average values of *NMI* obtained by MAGA-Net, Meme-Net and GA-Net on synthetic LFR networks as u ranging from 0.1 to 0.6. The number of nodes is now $N = 1000$.

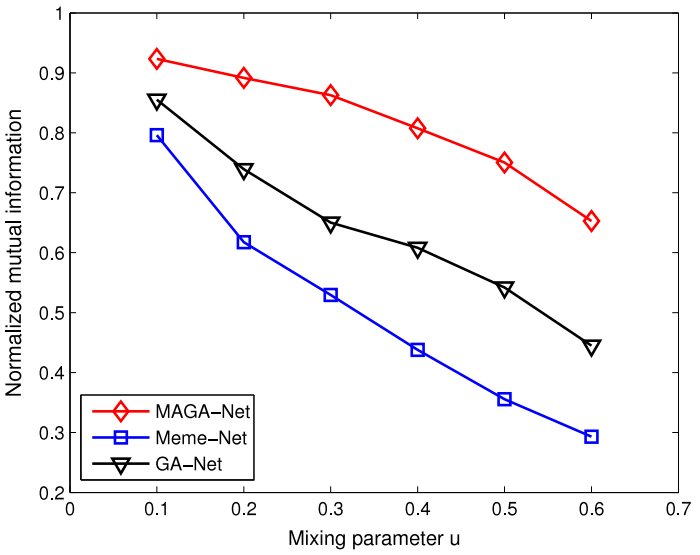


Fig. 7. The comparisons of average values of *NMI* obtained by MAGA-Net, Meme-Net and GA-Net on synthetic LFR networks as u ranging from 0.1 to 0.6. The number of nodes is now $N = 5000$.

ilarity with the real ones. However, the limitations of Meme-Net and GA-Net are exposed as the size of networks and the value of u increases.

The experimental results in Fig. 7 show that Meme-Net and GA-Net cannot handle large networks with 5000 nodes, while MAGA-Net obtains good results with the size 5000. Changing the value of u from 0.1 to 0.6 makes it much harder to detect communities, but MAGA-Net overcomes the difficulties and still has a good performance which comprehensively verifies the effectiveness of MAGA-Net.

All the above results show that MAGA-Net has a good performance. In terms of speed, MAGA-Net can converge to the global optima with a small number of evaluations, while Meme-Net and GA-Net are far from convergence with the same number of evaluations. According to the common sense, Meme-Net with a local search should be better than GA-Net, however, all the results show that GA-Net has a relatively good performance. The main reason is that the local search method in Meme-Net wastes a lot of computational resources which result in no improvement of the algorithm. This also verifies the superiority of MAGA-Net from the opposite side. Moreover, MAGA-Net has a good stability according to the standard deviations and can handle large-scale networks with 5000 nodes.

5. Conclusions

In this paper, we propose an algorithm named as MAGA-Net to optimize the modularity value for community detection. Traditional GAs had been used to solve this kind of problems, but they were apt to trap in local optima and were not suitable for large-scale networks. In MAGA-Net, a series of operators are designed, namely split and merging based neighborhood competition operator, hybrid neighborhood crossover, adaptive mutation, and self-learning operator, to increase modularity value as much as possible. The experiments on the four well-known real-world benchmark networks and the synthetic LFR graphs show that our method has the ability to find the global optima and can be used to solve large-scale networks with 5000 nodes. The systematic comparisons with GA-Net and Meme-Net indicate that MAGA-Net outperforms these two methods and can find the best partitions of networks with high speed, accuracy and stability. MAGA-Net can also achieve the optimal values found by the state-of-the-art algorithms compared with BKR. In addition, it is confirmed that MAGA-Net can handle networks with different densities of edges besides communities by turning the parameter u in LFR networks. However, there are still limitations of our proposed algorithm. We only consider the topological structure in network clustering, while in many real graphs, each vertex usually has one or more attributes describing its properties which are often homogeneous in a community. These attributes sometimes are equally important to topological structure in graph clustering. Thus, we are planning to design algorithms considering topological structure and vertex properties meanwhile in our future work.

Acknowledgments

This work is partially supported by the Outstanding Young Scholar Program of National Natural Science Foundation of China (NSFC) under Grant 61522311, the General Program of NSFC under Grant 61271301, the Overseas, Hong Kong & Macao Scholars Collaborated Research Program of NSFC under Grant 61528205, the Research Fund for the Doctoral Program of Higher Education of China under Grant 20130203110010, and the Fundamental Research Funds for the Central Universities under Grant K5051202052.

References

[1] S. Fortunato, C. Castellano, Community structure in graphs, in: Computational Complexity, Springer, 2012, pp. 490–512.
[2] M. Girvan, M.E. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. 99 (12) (2002) 7821–7826.
[3] M.E. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.
[4] M.E. Newman, Modularity and community structure in networks, Proc. Natl. Acad. Sci. 103 (23) (2006) 8577–8582.
[5] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, D. Wagner, On modularity clustering, IEEE Trans. Knowl. Data Eng. 20 (2) (2008) 172–188.
[6] W. Zhong, J. Liu, M. Xue, L. Jiao, A multiagent genetic algorithm for global numerical optimization, IEEE Trans. Syst. Man Cybern. B 34 (2) (2004) 1128–1141.
[7] J. Liu, W. Zhong, L. Jiao, A multiagent evolutionary algorithm for combinatorial optimization problems, IEEE Trans. Syst. Man Cybern. B 40 (1) (2010) 229–240.
[8] J. Liu, W. Zhong, L. Jiao, A multiagent evolutionary algorithm for constraint satisfaction problems, IEEE Trans. Syst. Man Cybern. B 36 (1) (2006) 54–73.
[9] J. Tang, S. Alam, C. Lokan, H.A. Abbass, A multi-objective approach for dynamic airspace sectorization using agent based and geometric models, Transp. Res. 21 (1) (2012) 89–121.
[10] L.M. Naeni, R. Berretta, P. Moscato, MA-Net: A reliable memetic algorithm for community detection by modularity optimization, in: Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems, Vol. 1, Springer, 2015, pp. 311–323.
[11] M.E. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6) (2004) 066133.
[12] A. Capocci, V.D. Servedio, G. Caldarelli, F. Colaiori, Detecting communities in large networks, Phys. A 352 (2) (2005) 669–676.
[13] V.D. Blondel, J.-L. Guillaume, R. Lambiotte, E. Lefebvre, Fast unfolding of communities in large networks, J. Stat. Mech. Theory Exp. 2008 (10) (2008) P10008.
[14] A. Clauset, M.E. Newman, C. Moore, Finding community structure in very large networks, Phys. Rev. E 70 (6) (2004) 066111.
[15] M. Zhou, J. Liu, A memetic algorithm for enhancing the robustness of scale-free networks against malicious attacks, Phys. A 410 (2014) 131–143.
[16] T.N. Bui, B.R. Moon, Genetic algorithm and graph partitioning, IEEE Trans. Comput. 45 (7) (1996) 841–855.
[17] E.-G. Talbi, P. Bessiere, A parallel genetic algorithm for the graph partitioning problem, in: Proceedings of the 5th International Conference on Supercomputing, ACM, 1991, pp. 312–320.
[18] M. Tasgin, A. Herdagdelen, H. Bingol, Community detection in complex networks using genetic algorithms. ArXiv preprint arXiv:0711.0491.
[19] C. Pizzuti, Ga-net: A genetic algorithm for community detection in social networks, in: Parallel Problem Solving from Nature–PPSN X, Springer, 2008, pp. 1081–1090.
[20] A. Gog, D. Dumitrescu, B. Hirsbrunner, Community detection in complex networks using collaborative evolutionary algorithms, in: Advances in Artificial Life, Springer, 2007, pp. 886–894.
[21] M. Gong, B. Fu, L. Jiao, H. Du, Memetic algorithm for community detection in networks, Phys. Rev. E 84 (5) (2011) 056101.
[22] M. Gong, L. Ma, Q. Zhang, L. Jiao, Community detection in networks by using multiobjective evolutionary algorithm with decomposition, Phys. A 391 (15) (2012) 4050–4060.
[23] J. Liu, W. Zhong, H. Abbass, D.G. Green, Separated and overlapping community detection in complex networks using multiobjective evolutionary algorithms, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–7.
[24] Y. Li, J. Liu, C. Liu, A comparative analysis of evolutionary and memetic algorithms for community detection from signed social networks, Soft Comput. 18 (2) (2014) 329–348.
[25] Y. Zeng, J. Liu, Community detection from signed social networks using a multi-objective evolutionary algorithm, in: Proceedings of the 18th Asia Pacific Symposium on Intelligent and Evolutionary Systems, Vol. 1, Springer, 2015, pp. 259–270.
[26] C. Liu, J. Liu, Z. Jiang, A multiobjective evolutionary algorithm based on similarity for community detection from signed social networks, IEEE Trans. Cybernet. 44 (12) (2014) 2274–2287.
[27] J. Liu, H. Jing, Y.Y. Tang, Multi-agent oriented constraint satisfaction, Artificial Intelligence 136 (1) (2002) 101–144.
[28] J. Liu, Autonomous Agents and Multi-Agent Systems: Explorations in Learning, Self-Organization, and Adaptive Computation, World Scientific, 2001.
[29] Y. Park, M. Song, A genetic algorithm for clustering problems, in: Proceedings of the Third Annual Conference on Genetic Programming, 1998, pp. 568–575.

[30] J. Handl, J. Knowles, An evolutionary approach to multiobjective clustering, IEEE Trans. Evol. Comput. 11 (1) (2007) 56–76.
[31] C. Pizzuti, A multiobjective genetic algorithm to find communities in complex networks, IEEE Trans. Evol. Comput. 16 (3) (2012) 418–430.
[32] L. Danon, A. Diaz-Guilera, J. Duch, A. Arenas, Comparing community structure identification, J. Stat. Mech. Theory Exp. 2005 (09) (2005) P09008.
[33] W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. (1977) 452–473.
[34] D. Lusseau, K. Schneider, O.J. Boisseau, P. Haase, E. Slooten, S.M. Dawson, The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations, Behav. Ecol. Sociobiol. 54 (4) (2003) 396–405.
[35] V. Krebs, Books about us politics, 2004, http://www.orgnet.com.
[36] G. Agarwal, D. Kempe, Modularity-maximizing graph communities via mathematical programming, Eur. Phys. J. B 66 (3) (2008) 409–418.
[37] A. Noack, R. Rotta, Multi-level algorithms for modularity clustering, in: Experimental Algorithms, Springer, 2009, pp. 257–268.
[38] Z. Ye, S. Hu, J. Yu, Adaptive clustering algorithm for community detection in complex networks, Phys. Rev. E 78 (4) (2008) 046115.
[39] A. Medus, G. Acuna, C. Dorso, Detection of community structures in networks via global optimization, Phys. A 358 (2) (2005) 593–604.
[40] G. Xu, S. Tsoka, L.G. Papageorgiou, Finding community structures in complex networks using mixed integer optimisation, Eur. Phys. J. B 60 (2) (2007) 231–239.
[41] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (4) (2008) 046110.