Neurocomputing 151 (2015) 1216–1226

Contents lists available at ScienceDirect

Neurocomputing

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

A cellular learning automata based algorithm for detecting community structure in complex networks

Yuxin Zhao [a], Wen Jiang [a], Shenghong Li [a,*], Yinghua Ma [b], Guiyang Su [b], Xiang Lin [b]

[a] Department of Electronic Engineering, Shanghai Jiao Tong University, 800 Dong Chuan Road, Shanghai 200240, China
[b] School of Information Security Engineering, Shanghai Jiao Tong University, 800 Dong Chuan Road, Shanghai 200240, China

ARTICLE INFO

Article history:
Received 26 February 2014
Received in revised form
24 April 2014
Accepted 28 April 2014
Communicated by D.-S. Huang
Available online 18 October 2014

Keywords:
Community detection
Complex network
Modularity optimization
Resolution limit
Cellular learning automata

ABSTRACT

Community structure is a common and important property of complex networks. The detection of communities has great significance for understanding the function and organization of networks. Generally, community detection can be formulated as a modularity optimization problem. However, traditional modularity optimization based algorithms have the resolution limit that they may fail to find communities which are smaller than a certain size. In this paper, we propose a cellular learning automata based algorithm for detecting communities in complex networks. Our algorithm models the whole network as an irregular cellular learning automata (ICLA) and reveals the optimal community structure through the evolution of the cellular learning automata. By interacting with both the global and local environments, our algorithm effectively solves the resolution limit problem of modularity optimization. The experiments on both synthetic and real-world networks demonstrate that our algorithm is effective and efficient at detecting community structure in complex networks.

1. Introduction

Many real-world systems can be represented by complex networks, such as social networks, information networks, and biological networks [1,2]. Community structure is considered to be a common and important property of complex networks. A community can be generally described as a group of nodes with dense internal connections and relatively sparse connections to other groups [3–5]. Since communities in complex networks correspond to the functional entities or units of the underlying systems, detection of community structure can provide useful information and important insights to understand the organization and function of complex systems [6].

Modularity is an important measure function used for evaluating the significance of community structure in the network, which was introduced by Newman and Girvan [7]. On the basis of modularity, community detection can be modeled as a modularity optimization problem, which has been proven to be NP-hard. In recent years, numerous optimization methods have been proposed to solve the problem of community detection in complex networks, including FN [8], CNM [9], extremal optimization [10], group search optimization [11] and genetic algorithm [12].

However, the researches by Fortunato and Barthélemy [13] have shown that the optimization of modularity may fail to find communities which are smaller than a certain size. This is famously known as the resolution limit of modularity optimization. To overcome the resolution limit, many other quality metrics [14–17] have been proposed to discover community structure at different scales. Most of these metrics need a tunable parameter to determine the resolution level of community structure. Another approach to solve the resolution limit is to formulate the community detection as a multi-objective optimization problem. The multi-objective optimization based algorithms [18–21] achieve the Pareto optimal solutions by simultaneously optimizing multiple objective functions that evaluate the community structure from different perspectives.

Cellular learning automata (CLA) is a powerful mathematical model for many decentralized problems and dynamic phenomena by combining cellular automata (CA) and learning automata (LA) [22]. Cellular learning automata can be defined as a class of cellular automata where each cell is assigned with a learning automaton. The basic idea of cellular learning automata is to adjust the state transition probabilities of stochastic cellular automata using learning automata. Cellular learning automata is superior to cellular automata because the learning automata residing in the cells provides the ability to learn. It is also superior to single learning automaton because the learning automata residing in different cells can interact with each other to produce complicated

* Corresponding author.
E-mail address: shli@sjtu.edu.cn (S. Li).

http://dx.doi.org/10.1016/j.neucom.2014.04.087
0925-2312/

patterns [23]. Cellular learning automata has been successfully applied in many fields, such as image processing [24,25], sensor networks [26–28], numerical optimization [29–31] and sociological analysis [32,33].

In this paper, we propose a cellular learning automata based algorithm called CLA-net for detecting community structure in complex networks. In our algorithm, the whole network is modeled as an irregular cellular learning automata (ICLA), where each node is equipped with a cell of cellular learning automata and each cell is assigned with a learning automaton. Through the interactions with both the global and local environments, the cellular learning automata evolves and gradually reveals the optimal community structure in the network. Our main contributions are summarized as follows:

1. We successfully introduce the cellular learning automata to solve the problem of community detection in complex networks.
2. Our algorithm solves the resolution limit of modularity optimization through the interactions with both global and local environments.
3. The tests on both synthetic and real-world networks demonstrate the effectiveness and efficiency of our algorithm.

The rest of the paper is organized as follows. In Section 2, we introduce the related background of community detection in complex networks. Section 3 gives a brief review of the theory of cellular learning automata. A detailed description of our proposed cellular learning automata based algorithm is presented in Section 4. The experimental results and discussions are reported in Section 5. Finally, Section 6 gives the conclusion of this paper.

2. Related background

2.1. Definition of community structure

Community structure is a common and important property of complex networks. Generally, a community can be defined as a group of nodes with dense internal connections and relatively sparse connections to the rest of the network. This notion is a kind of ambiguous. A more explicit definition of community structure is required when tackling the problem of community detection. Here, we introduce several quantitative definitions of community structure which are widely adopted in the community detection literature.

Generally, a network can be represented by a simple graph $G=(V,E)$, where V is the set of nodes and $E=\{(i,j)|i,j\in V\}$ is the set of edges between the nodes. The topology of the network is fully specified by the adjacency matrix A, where $A_{ij}=1$ if node i and node j are directly connected and $A_{ij}=0$ otherwise.

Radicchi et al. [34] proposed two local definitions of community in a strong sense and a weak sense. Considering a subnetwork $C\subset G$, to which node i belongs, the total degree of node i is split into two contributions: $k_i=k_i^{in}(C)+k_i^{out}(C)$, where $k_i^{in}(C)=\sum_{j\in C}A_{ij}$ is the number of edges connecting node i to the nodes belonging to subnetwork C and $k_i^{out}(C)=\sum_{j\notin C}A_{ij}$ is clearly the number of edges connecting node i towards the rest of the network. The strong community is a subnetwork that satisfies the constraint:

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

In a strong community, each node has more connections within the community than with the rest of the network. Compared with strong community, weak community is under a relaxed constraint:

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

Weak community requires that the sum of node degrees within the community is larger than the sum of node degrees toward the rest of the network. According to the definitions, a strong community is also a weak community, whereas the converse is not true.

Another definition of community structure was proposed by Raghavan et al. [35]. This definition has been shown to be highly accordant with the community structure in real-world networks and it is also adopted in our proposed algorithm. Let Ω be the set of existing communities in the network and $|\Omega|$ denote the number of communities. The total degree of any node i is split into $|\Omega|$ contributions: $k_i=\sum_{C\in\Omega}k_i(C)$, where $k_i(C)=\sum_{j\in C}A_{ij}$ is the number of edges between node i and the nodes belonging to community C. Given a community C, the Raghavan definition can be written as

$$k_i(C)\geq k_i(C'),\quad \forall i\in C,\quad \forall C'\in\Omega \tag{3}$$

It is required that each node has no fewer connections within the community it belongs to than it has with each of the other communities. When there are only two communities in the network, the Raghavan definition is approximately equivalent to the definition of strong community. When the network contains more than two communities, the constraint of Raghavan definition is weaker than that of strong community.

2.2. Community detection

The goal of community detection is to find a partition that could divide the network into the most significant communities. Hence, community detection in complex network can be naturally formulated as an optimization problem. For this purpose, Newman and Girvan [7] proposed the concept of modularity, which has become a widely used criterion for identifying community structure. The foundation of modularity is inspired by the idea that no community structure is expected to be found in a random network. By comparing the fraction of edges within the given communities and the expected value of such fraction in a random network, the modularity measures the significance of the community structure in the network. Formally, modularity can be written as

$$Q=\frac{1}{2m}\sum_{i,j\in V}\left(A_{ij}-\frac{k_ik_j}{2m}\right)\delta(i,j) \tag{4}$$

where m is the number of edges in the network, A_{ij} is the element of the adjacency matrix for the network, k_i is the degree of node i, and $\delta(\cdot)$ is the extended Kronecker delta function, i.e., $\delta(i,j)=1$ if nodes i and j are in the same community; otherwise, $\delta(i,j)=0$. The term $k_ik_j/2m$ indicates the expected number of edges connecting node i and node j in a random network of the same size and node degree distribution. If the number of edges within communities is greater than the expected number in a random network, the modularity value Q would be greater than 0. Larger value of modularity indicates more significant community structure in the network.

On the basis of modularity, the community detection can be equivalent to a modularity optimization problem. The search for the optimal modularity has been proven to be an NP-hard problem due to the fact that the number of possible partitions grows faster than any power of the network size. Large numbers of optimization methods have been proposed to solve the problem of community detection in the network. Newman [8] proposed a greedy algorithm FN that starts from a partition in which each

node is a unique group and then repeatedly merges a pair of groups with the largest gain of modularity. Clauset et al. [9] utilized a sophisticated data structure to reduce the calculation complexity of modularity and made the FN algorithm applicable to large-scale networks. Duch and Arenas [10] presented a divisive algorithm that optimizes the modularity using a heuristic search based on the extremal optimization. Kumar and Jayaraman [11] used the technique of group search optimization to reveal the optimal community structure in real-world networks. Shang et al. [12] proposed an improved genetic algorithm called MIGA to make an approach to the largest modularity.

However, Fortunato and Barthélemy [13] have found that the modularity optimization based algorithms may fail to find the communities smaller than a certain size, which is determined by the total size of the network and the degree of interconnectedness between the communities. This is famously known as the resolution limit of modularity optimization. To overcome the resolution limit, many other quality metrics have been proposed to evaluate the community structure at different scales. Arenas et al. [14] introduced a scale parameter into the modularity to tune the resolution level. Li et al. [15] proposed a variant of modularity called the *Modularity Density* to measure the significance of community structure. Pizzuti [16] put forward the *Community Score* (*CS*) criterion to guarantee highly intra-connected and sparsely inter-connected communities. Lancichinetti et al. [17] proposed the *Community Fitness* (*CF*) to determine the community scales.

Another approach to solve the resolution limit is to formulate the community detection as a multi-objective optimization problem. The multi-objective optimization based algorithms achieve the Pareto optimal solutions by simultaneously optimizing multiple objective functions that evaluate the community structure from different perspectives. Shi et al. [18] decomposed *Modularity* into two conflicting objective functions and optimize them using the evolutionary algorithm PESA-II. Pizzuti [19] adopted *Community Score* and *Community Fitness* as two objectives and used the genetic algorithm NSGA-II to achieve optimal partitions. Gong et al. [20] presented the algorithm MOEA/D-Net which optimizes two conflicting objective functions decomposed from *Modularity Density*. Gong et al. [21] also proposed several improvements of the objective functions derived from *Modularity Density*.

3. Theory of cellular learning automata

In this section, cellular automata (CA) and learning automata (LA) are briefly described first. Then, we introduce the theory of cellular learning automata (CLA), which is the combination of cellular automata and learning automata. Finally, a description of irregular cellular learning automata (ICLA) is presented as an extension of basic cellular learning automata.

3.1. Cellular automata

Cellular automata (CA) is a mathematical idealization of complex systems constructed from large numbers of simple identical components with local interactions [36]. Cellular automata is a non-linear dynamic model characterized by discrete space and time. A cellular automata consists of a regular lattice of identical cells, each of which takes on a finite set of states. The local environment of a cell is usually composed of the cell itself and its neighboring cells. In each discrete time step, the cells of the cellular automata update their states synchronously by interacting with the local environments according to a local rule. The evolution of cellular automata is completely specified by the initial states of the cells and the updating rule. Cellular automata can produce complex patterns with simple structure, displaying the potential to simulate different sophisticated natural systems.

3.2. Learning automata

Learning automaton is an adaptive decision-making model, which attempts to learn the optimal action from a finite set of allowable actions through a series of interactions with unknown random environments [37].

The functionality of a learning automaton can be described in terms of a sequence of repetitive feedback cycles in which the learning automaton interacts with the environment. During a cycle, the learning automaton chooses an action from its available actions according to the probability distribution kept over the action set. Then, the environment gives a response to the chosen action, which can be either a reward or a penalty. At last, the learning automaton updates the action probability distribution depending on this response and the knowledge acquired in the past cycles. The repetitions of the feedback cycles constitute the learning process of learning automaton. This learning process finally converges to the optimal action that maximizes the probability of being rewarded.

Learning automata can be classified into two categories: fixed structure stochastic automata (FSSA) and variable structure stochastic automata (VSSA). In the definition of VSSA, a learning automaton can be completely defined by a quadruple $(\alpha, \beta, \mathbf{p}, T)$, where

- $\alpha = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ is the action set of learning automaton.
- $\beta = \{\beta_1, \beta_2, \ldots, \beta_m\}$ is the value set of the response from the environment.
- $\mathbf{p} = (p_1, p_2, \ldots, p_r)$ is the action probability vector, where p_i is the probability of choosing action α_i and satisfies $\sum_{i=1}^{r} p_i = 1$.
- T is the learning algorithm that modifies the action probability vector according to the response from the environment. The update obeys $\mathbf{p}(t+1) = T(\mathbf{p}(t), \alpha(t), \beta(t))$, where $\alpha(t)$, $\beta(t)$ and $\mathbf{p}(t)$ are respectively the chosen action, the environment's response and the action probability vector at cycle t.

The core factor of learning automata is the choice of the learning algorithm T. The basic learning algorithm is the Linear Reward Penalty Algorithm L_{RP}. Let α_i be the action chosen and β be the response from the environment at cycle t. The action probability vector is updated according to Eq. (5) if the chosen action is rewarded by the environment ($\beta = 0$), and it is updated according to Eq. (6) if the chosen action is penalized ($\beta = 1$):

$$p_j(t+1) = \begin{cases} p_j(t) + a(1 - p_j(t)) & \text{if } j = i \\ (1-a)p_j(t) & \text{if } j \neq i \end{cases} \tag{5}$$

$$p_j(t+1) = \begin{cases} (1-b)p_j(t) & \text{if } j = i \\ \dfrac{b}{r-1} + (1-b)p_j(t) & \text{if } j \neq i \end{cases} \tag{6}$$

where r is the cardinality of the action set, a and b are respectively the reward and penalty parameters.

In order to increase the speed of convergence, Thathachar and Sastry [33] introduced the concept of estimator by presenting a Pursuit Algorithm denoted by CP_{RP}. The CP_{RP} pursues the action that is currently estimated to be the optimal action. The current optimal action is determined according to the estimate vector $\hat{\mathbf{D}}$, which can be calculated as

$$\hat{D}_i(t) = \frac{W_i(t)}{Z_i(t)} \quad \text{for } i = 1, 2, \ldots, r \tag{7}$$

where $Z_i(t)$ is the number of times the action α_i has been chosen

up to cycle t, and $W_i(t)$ is the number of times the action α_i has been rewarded up to cycle t. The action with the largest value of $\hat{D}_i(t)$ is estimated to be the current optimal action for cycle t. During each cycle, the CP_{RP} firstly chooses an action from its available actions according to the action probability vector. Then, if the chosen action is either rewarded or penalized, the CP_{RP} only increases the probability of the current optimal action α_m according to the following equations:

$$p_j(t+1)=\begin{cases} p_j(t)+a(1-p_j(t)) & \text{if } j=m \\ (1-a)p_j(t) & \text{if } j\neq m \end{cases} \tag{8}$$

where a is the reward parameter.

The CP_{RP} algorithm is similar in design to the L_{RP} algorithm, in the sense that both algorithms modify the action probability vector through interactions with the environment. The main difference lies on the way they approach to the solution. The L_{RP} moves the action probability vector $\mathbf{p}(t)$ in the direction of the most recently rewarded action, while the CP_{RP} moves $\mathbf{p}(t)$ in the direction of the action which has the highest reward estimate [38]. More improvements in the field of learning automata can be seen in [39–41].

3.3. Cellular learning automata

Cellular learning automata (CLA) is a powerful mathematical model for many decentralized problems and dynamic phenomena by combining cellular automata (CA) and learning automata (LA). The basic idea of cellular learning automata is to adjust the state transition probability of stochastic cellular automata using learning automata. Cellular learning automata can be simply defined as a class of cellular automata where each cell is assigned with a learning automaton. The learning automaton residing in a particular cell determines its state on the basis of its action probability vector. The local environment of a learning automaton is composed of the learning automata in the neighboring cells. Each learning automaton attempts to learn the optimal action by interacting with its local environment. With a lattice of such learning automata, cellular learning automata is capable of producing complicated behavior patterns.

The mechanism of cellular learning automata can be described as follows. At first, the internal states of every cell are specified. The action probability vector of each learning automaton residing in the cells is initialized on the basis of past experiences or at random. Then, the learning automaton in each cell determines its state according to the action probability vector and receives an response from the local environment. Finally, the action probability vector of every learning automaton is updated depending on the environment's response. This process is repeated until the optimal state of any cell is achieved.

Formally, d-dimensional cellular learning automata can be represented by a structure $A=(Z^d,\Phi,L,N,f)$ [23] where

- Z^d is a lattice of cells. Each cell in the lattice is represented by a d-tuple of integer numbers.
- Φ is the state set for the cellular learning automata.
- L is the set of learning automata each of which is assigned to a cell of the cellular learning automata.
- $N=\{\overline{x_1},\overline{x_1},\ldots,\overline{x_m}\}$ is a finite set of integer numbers called neighborhood vector, which determines the relative position of the cells. The neighboring cells of a particular cell u are the set of cells $\{u+x_i|i=1,2,\ldots,m\}$.
- $f:\Phi^m\rightarrow\beta$ is the local rule of the cellular learning automata, where Φ^m is the states of the learning automata in the neighboring cells and β is the value set of the response. It computes the response for each learning automaton according to the current states of the learning automata in the neighboring cells.

Cellular learning automata can be classified into two categories: synchronous cellular learning automata and asynchronous cellular learning automata. In synchronous cellular learning automata, the learning automata residing in the entire lattice of cells are activated at the same time in parallel. For asynchronous cellular learning automata, only some of the learning automata are activated independently from each other at a given time.

In some practical applications, the interactions between cellular learning automata and external environments are also taken into account. Such cellular learning automata is referred to as open synchronous cellular learning automata (OSCLA) [42], in which the evolution of cellular learning automata depends on not only local environments (neighboring cells) but also external environments. In [42], Beigy and Meybodi have found many important behavioral properties of open synchronous cellular learning automata. Since open synchronous cellular learning automata interacts with different kinds of environments, it can produce more complicated patterns and behaviors.

3.4. Irregular cellular learning automata

Irregular cellular learning automata (ICLA) [43] is an extension of basic cellular learning automata, in which the restriction of rectangular lattice structure in basic cellular learning automata is removed. This extension is due to the fact that many applications, such as graph processing and network analysis, cannot be adequately modeled in the form of rectangular lattice.

Irregular cellular learning automata can be described as an undirected graph, where each node represents a cell which is equipped with a learning automaton. The operations of irregular cellular learning automata are approximately the same as those of basic cellular learning automata. The main difference is that for irregular cellular learning automata, the environment of a learning automaton is constituted by the learning automata residing in the neighbor nodes in the graph. Generally, irregular cellular learning automata can be represented by a structure $A=(G,\Phi,L,f)$, where

- $G=(V,E)$ is an undirected graph, where V is the set of nodes and E is the set of edges. The graph determines the relative position of the cells of irregular cellular learning automata.
- Φ is the state set for the irregular cellular learning automata.
- L is the set of learning automata each of which is assigned to a cell of the irregular cellular learning automata.
- $f:\Phi^{N(i)}\rightarrow\beta$ is the local rule of the irregular cellular learning automata for each node i, where $N(i)$ is the set of the neighbor nodes of node i in graph G, $\Phi^{N(i)}$ is the states of learning automata in the neighbor nodes and β is the value set of the response. It computes the response for each learning automaton based on the current states of the learning automata residing in the neighbor nodes.

4. Our proposed algorithm

In this section, we propose a cellular learning automata based algorithm called CLA-net for detecting the community structure in complex networks. In our proposed algorithm, the whole network is modeled as an irregular cellular learning automata and the optimal community structure is identified through the evolution of the cellular learning automata. The details of our algorithm are presented below. First, the solution representation for the problem of community detection is introduced. Next, we describe the construction of the cellular learning automata. Then, a detailed description of the framework of the proposed CLA-net algorithm is presented. At last, we make an analysis of the time complexity of the proposed algorithm.

4.1. Solution representation

Given a complex network $G=(V,E)$, the community structure can be generally represented by a membership vector $\mathbf{C}=(c_1,c_2,\ldots,c_n)$, where c_i indicates the index of the community that node i belongs to. However, the main drawback of membership representation is that it requires prior knowledge of the number of communities in the network.

In the CLA-net algorithm, we adopt the locus-based adjacency representation proposed in [44] and employed by Pizzuti [16] for community detection. In locus-based adjacency representation, the solution is represented by a solution vector $\mathbf{S}=(s_1,s_2,\ldots,s_n)$, where s_i indicates that node i and node s_i are in the same community. The solution vector only describes the edges rather than the communities. A decoding process is necessary to fully reveal the community structure in the network, which can be done through a breadth-first search or a depth-first search in linear time. After decoding, the solution vector is transferred into the membership vector representing the community structure. Fig. 1(a) shows a sample network with 14 nodes, which are divided into three communities. A solution vector in locus-based adjacency representation and the corresponding membership vector are respectively shown in Fig. 1(b) and (c).

The main advantage of locus-based adjacency representation is that it can represent community structure with dynamic number of communities. The number of communities is automatically determined in the decoding process.

4.2. Solution construction

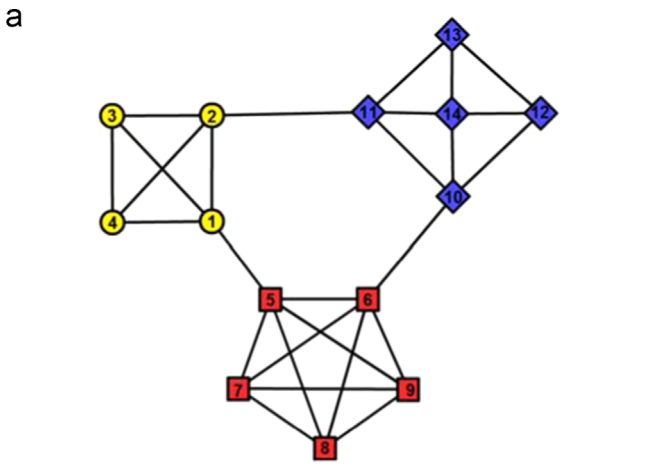
In the CLA-net, the whole network is modeled as an irregular cellular learning automata. To construct such an irregular cellular learning automata, each node in the network is equipped with a cell of cellular learning automata, and then a learning automaton is assigned to each cell. The state of each cell depends on the current action chosen by the learning automaton residing in it. The structure of a learning automata L_i residing in node i can be described by a 3-tuple $(\alpha,\beta,\mathbf{p})$, where

- $\alpha_i=\{\alpha_{i1},\alpha_{i2},\ldots,\alpha_{ir}\}=N(i)$ is the action set, where $N(i)$ is the set of the neighbor nodes of node i in the network.
- $\beta_i=\{0,1\}$ is the value set of the response from the environment, where 0 and 1 respectively correspond to reward and penalty.
- $\mathbf{p}_i=(p_{i1},p_{i2},\ldots,p_{ir})$ is the action probability vector, where p_{ij} is the probability of choosing action α_{ij} for learning automaton L_i.

The solution vector is constituted by the states of the cells in the entire network, namely the current actions chosen by all the learning automata. Hence, at each cycle t, the solution vector can be written as

$$\mathbf{S}(t)=(\alpha_1(t),\alpha_2(t),\ldots,\alpha_n(t)) \tag{9}$$

where $\alpha_i(t)$ is the action chosen by learning automata L_i at cycle t. The solution vector is updated along with the evolution of the cellular learning automata.



Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Solution Vector	2	1	1	3	6	7	5	9	6	13	12	14	12	11

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Membership Vector	1	1	1	1	2	2	2	2	2	3	3	3	3	3

Fig. 1. (a) A sample network with three communities. Different communities are rendered in different shapes; (b) a solution vector in locus-based adjacency representation for the community structure; (c) the membership vector corresponding to the community structure.

4.3. Framework of the proposed algorithm

The CLA-net algorithm employs open synchronous cellular learning automata (OSCLA) due to its powerful capability of producing complicated patterns. The learning automata in the entire network are activated simultaneously and each learning automaton is influenced by both its local environment and the global environment. The local environment of any learning automaton is constituted by the learning automata in the neighbor nodes and the global environment contains all the learning automata in the network. The action probability vector of the learning automaton is updated according to the response from the environments using the CP_{RP} algorithm. The interactions between the learning automaton and the environments are shown in Fig. 2.

The procedure of the CLA-net algorithm is described as follows. At each cycle t, each learning automaton chooses an action according to its action probability vector. The actions of the learning automata in the entire network constitute the solution vector $\mathbf{S}(t)=(\alpha_1(t),\alpha_2(t),\ldots,\alpha_n(t))$. Through the decoding process, the solution vector $\mathbf{S}(t)$ is transferred into the membership vector $\mathbf{C}(t)=(c_1(t),c_2(t),\ldots,c_n(t))$ to represent the obtained community structure. The global environment calculates the modularity $Q(t)$ for the current community structure in the network. The local environment of each learning automaton records the communities of the neighbor nodes in the network. For the learning automaton in each node, it would receive a reward response when it simultaneously satisfies the following conditions:

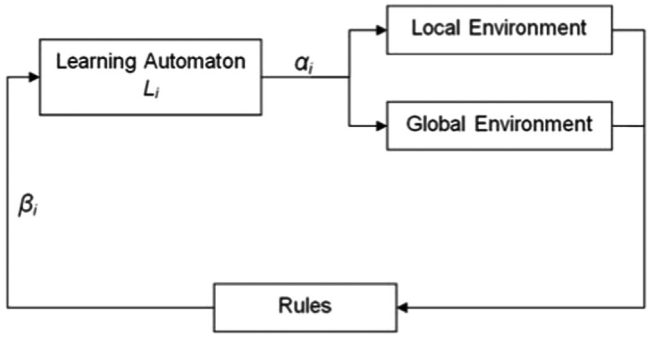


Fig. 2. The interactions between the learning automaton and the environments.

1. The modularity $Q(t)$ of the current community structure is not smaller than the best modularity Q_{best} in the past cycle.
2. The node satisfies the constraint of Raghavan definition of community structure in Eq. (3), which requires the node to have no fewer connections within its community than it has with each of the other communities.

Otherwise, the learning automaton would receive a penalty response. Then, for each learning automaton, the current optimal action is determined according to Eq. (7) and the action probability vector is updated according to Eq. (8). This process is repeated until the obtained community structure remains fixed in some consecutive cycles.

The interactions with the global environment guarantee that the CLA-net algorithm makes an approach to the optimal modularity of the community structure. The interactions with the local environment constrain the obtained community structure by Raghavan definition. By interacting with these two different kinds of environments, the CLA-net algorithm can effectively solve the resolution limit problem of modularity optimization.

The main framework of the proposed CLA-net algorithm for community detection in complex network is given below.

Algorithm. CLA-net.

Input

$A_{n\times n}$: the adjacency matrix of the network $G=(V,E)$, where V is the set of the nodes, E is the set of the edges connecting the nodes and $n=|V|$ is the number of nodes in the network. $A_{ij}=1$ if node i and node j are directly connected; otherwise, $A_{ij}=0$.

Parameters

a : the reward parameter for the update of the action probability vector, where $0<a<1$.

Variables

r_i: the number of the actions for learning automaton L_i, which is equal to the degree of node i in the network.

$W_{ij}(t)$: the number of times the jth action of learning automaton L_i has been rewarded up to cycle t , with $1\le i\le n$ and $1\le j\le r_i$.

$Z_{ij}(t)$: the number of times the jth action of learning automaton L_i has been chosen up to cycle t , with $1\le i\le n$ and $1\le j\le r_i$.

Q_{best}: The largest modularity obtained in the past cycles.

Method

Initialization

Step 1: $p_{ij}=1/r_i$, for $1\le i\le n$ and $1\le j\le r_i$.

Step 2: Initialize $W_{ij}(t)$, $Z_{ij}(t)$ and Q_{best} by randomly choosing the action for each learning automaton L_i a small number of times, with $1\le i\le n$ and $1\le j\le r_i$.

repeat

Step 3: Each learning automaton L_i chooses an action $\alpha_i(t)$ according to its action probability vector $p_i(t)$, for $1\le i\le n$.

Step 4: The solution vector $\mathbf{S}(t)=(\alpha_1(t),\alpha_2(t),\ldots,\alpha_n(t))$ is transferred into the membership vector $\mathbf{C}(t)=(c_1(t),c_2(t),\ldots,c_n(t))$ to represent the obtained community structure through the decoding process.

Step 5: The global environment calculates the modularity $Q(t)$ of the community structure represented by the membership vector $\mathbf{C}(t)$.

Step 6:

for each learning automaton $L_i(1\le i\le n)$ **do**

if $Q(t)\ge Q_{best}$ and $k_i(c_i(t))\ge k_i(c')$, $\forall c'\ne c_i(t)$ **then**

The response from the environments $\beta_i(t)=0$

else

The response from the environments $\beta_i(t)=1$

end if

end for

Step 7: update $Q_{best}=max(Q(t),Q_{best})$.

Step 8: For each learning automaton $L_i(1\le i\le n)$, given the chosen action $\alpha_i(t)=\alpha_{iq}$, update $W_{iq}(t)$ and $Z_{iq}(t)$ according to the following equations:

$$\begin{cases} W_{iq}(t)=W_{iq}(t-1)+(1-\beta_i(t)) \\ Z_{iq}(t)=Z_{iq}(t-1)+1 \end{cases}$$

Step 9: The current optimal action of each learning automaton L_i is estimated according to Eq. (7), for $1\le i\le n$.

Step 10: Update the action probability vector $\mathbf{p}_i$ of each learning automaton L_i according to Eq. (8), for $1\le i\le n$.

until The obtained community structure remains fixed in some consecutive cycles.

Output

The solution vector $\mathbf{S}(t)$, the membership vector $\mathbf{C}(t)$ and the corresponding community structure in the network.

4.4. Complexity analysis

We also make an analysis of the time complexity of the proposed CLA-net algorithm. Let n and m respectively denote the number of nodes and edges in the network. The average node degree is therefore $O(\bar{k})=O(m/n)$. The main time complexity of the CLA-net algorithm lies in Step 3, Step 5 and Step 6, since the other steps can be accomplished in linear time $O(n)$. Since the number of the actions for each learning automaton is equal to the degree of the node in the network, Step 3 needs $O(\bar{k})\cdot O(n)=O(m)$ time to choose the actions for all the learning automata. The calculation of modularity in Step 5 costs $O(m)$ time. It takes $O(\bar{k})$ time to verify the constraint of the Raghavan definition for each node, so that Step 6 needs $O(\bar{k})\cdot O(n)=O(m)$ time. According to the above analysis, the total time complexity of the proposed CLA-net algorithm is $O(Tm)$, where T is the number of the cycles.

5. Experimental results and discussion

We have tested the proposed CLA-net algorithm on both synthetic benchmark networks and real-world networks. The performances of the CLA-net algorithm are compared with several other community detection algorithms including CNM [9], MIGA [12], Meme-net [45], GA-net [16], MOCD [18], MOGA-net [19] and MOEA/D-net [20]. The experiments are implemented by MATLAB 2009b running on a PC with a 2.4 GHz processor and 3GB memory.

To evaluate the performances of different community detection algorithms, we adopt *Modularity* [7] and *Normalized Mutual Information* (*NMI*) [17] as the basic measures for the experiments reported in this paper. *Modularity* Q evaluates the significance of community structure in the network. The larger value of Q indicates more internal connections within the communities than would be expected by pure chance. *Normalized Mutual Information* (*NMI*) is specially for the networks with known community structure. It evaluates the similarity between real communities and the communities obtained by the algorithms. The value of *NMI* is between [0, 1] and the larger value indicates that the obtained communities are more accordant with the real situation.

5.1. Synthetic networks

5.1.1. Evaluation on GN benchmark networks

We first do the experiments on GN benchmark networks introduced by Girvan and Newman [46] to evaluate the accuracy of the proposed algorithm. The GN benchmark network contains 128 nodes, which are divided into four communities with 32

nodes each. The average degree of the nodes in the network is 16. A critical mixing parameter λ is used to control the community structure in the network. Each node shares a fraction $1-\lambda$ of connections with the nodes in its community, and λ of connections with the other nodes of the network. Smaller value of λ indicates that the community structure in the network is more significant. When $\lambda < 0.5$, the network has relatively strong communities; on the contrary, the communities are rather indistinct. We vary the mixing parameter λ from 0 to 0.5 with a span of 0.05 and test the CLA-net algorithm on GN benchmark networks in comparison with other algorithms. Since the built-in communities in benchmark networks are already known, we use the Normalized Mutual Information (*NMI*) to evaluate the performances of different community detection algorithms. Fig. 3 shows the performances of different algorithms on GN benchmark networks, where each data point is an average over 100 runs.

As shown in Fig. 3, when the mixing parameter $\lambda < 0.3$, almost all the algorithms, except GA-net and MOGA-net, can find the community structure corresponding to the correct partitions ($NMI \approx 1$). When $\lambda > 0.3$, the community structure in the network becomes indistinct, resulting in that the *NMI* of all the algorithms begins to decrease. Our algorithm slightly outperforms the MIGA and MOEA/D-net, and shows obvious advantages over the other algorithms. Moreover, it can be seen that our algorithm CLA-net always gives excellent results until the mixing parameter λ reaches 0.4, beyond which its performance deteriorates sharply. This phenomenon is mainly due to the fact that some nodes in the network no longer satisfy the constraint of the Raghavan definition for community structure when $\lambda > 0.4$, which would mislead the evolution of the cellular learning automata in our algorithm. From the results, we can see that the proposed CLA-net algorithm has great performances on GN benchmark networks in most situations.

5.1.2. Evaluation on LFR benchmark networks

The GN benchmark networks do not reflect some important properties of real-world networks, like the power-law distribution of node degrees and community sizes. Therefore, Lancichinetti et al. [47] proposed the LFR benchmark networks, which are more consistent with the properties of real-world networks. In LFR benchmark networks, both node degrees and community sizes follow the power-law distribution with exponents τ_1 and τ_2 respectively. The significance of the community structure depends on a mixing parameter μ, which indicates the average fraction of the connections to other communities per node. The benchmark network with smaller mixing parameter μ has more significant community structure.

We also apply the CLA-net algorithm on LFR benchmark networks along with other community detection algorithms. In our experiments, the network size is set to 1000 and the power-law exponents τ_1 and τ_2 are set to 2 and 1 respectively. The node degree is between [0, 50] and has an average value of 20. The community size is in the range from 10 to 50. Since some of the algorithms can perform well even when the community structure in LFR benchmark networks is indistinct, the mixing parameter μ varies from 0 to 0.8 with a span of 0.05. The *NMI* is still used as the measure to evaluate the performances of different algorithms. We run every algorithm 100 times and the statistical results are reported in Fig. 4.

Seen from Fig. 4, when the mixing parameter $\mu < 0.1$, there is no obvious difference among the results of all the algorithms. As the mixing parameter μ increases, the performances of most algorithms, except CLA-net, MIGA and Meme-net, decline sharply. When $\mu < 0.5$, our CLA-net algorithm can always find the correct partitions of the benchmark networks ($NMI \approx 1$). The performance of the CLA-net algorithm is in close proximity to that of the MIGA and Meme-net, and is significantly superior to the performances of other algorithms. As the mixing parameter μ exceeds 0.5, some of the build-in communities no longer satisfy the Raghavan definition, resulting in the sharp decline of the performance of the CLA-net algorithm. When $\mu > 0.6$, our CLA-net algorithm is outperformed by the MOEA/D-net, but still performs better than the other algorithms. The MOEA/D-net decomposes the problem of community detection into a number of scalar optimization subproblems in terms of different measures of community structure. The decomposition strategy has been proven to be highly effective at finding evenly distributed Pareto optimal solutions [48], so that it makes the MOEA/D-net outperform the other algorithms when the community structure in the network is indistinct. From the curves, we can draw the conclusion that our algorithm can perform well on LFR benchmark networks in most situations.

5.1.3. Evaluation of resolution limit

In the proposed CLA-net algorithm, the cellular learning automata makes an approach to the largest modularity of community structure by interacting with the global environment. We also make a brief experiment to verify whether our algorithm has resolution limit.

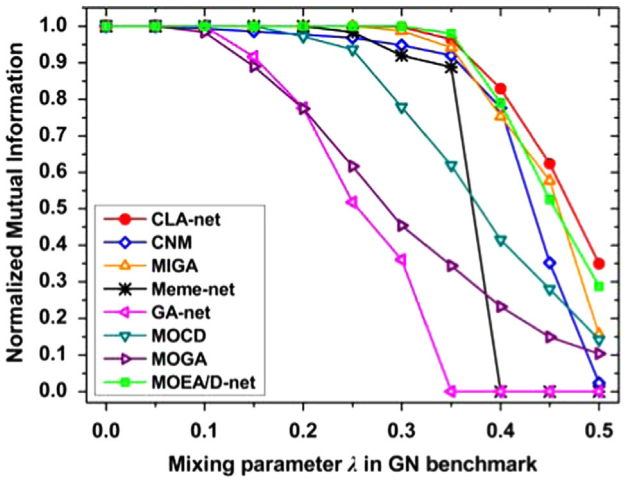


Fig. 3. The average *NMI* obtained by CLA-net, CNM, MIGA, Meme-net, GA-net, MOCD, MOGA-net and MOEA/D-net on GN benchmark networks.

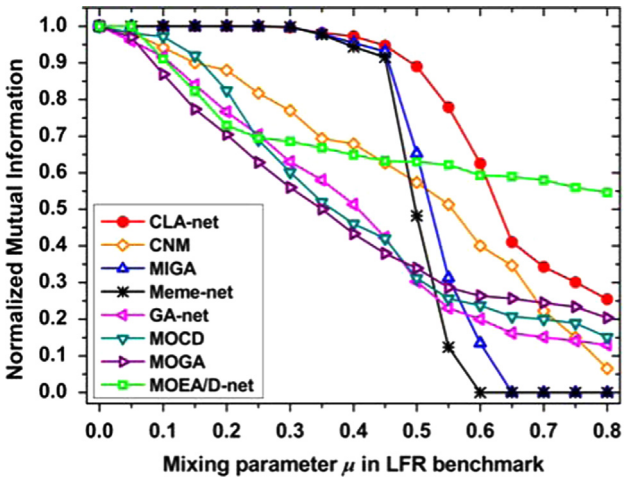


Fig. 4. The average *NMI* obtained by CLA-net, CNM, MIGA, Meme-net, GA-net, MOCD, MOGA-net and MOEA/D-net on LFR benchmark networks.

The test network contains 54 nodes and is composed of two communities C_1 and C_2 with 50 and 4 nodes respectively. In each community, a node is connected with all the other nodes in the same community. The single edge connecting the two communities is between node u in community C_1 and node v in community C_2. A schematic of the test network is shown in Fig. 5. The modularity of the community structure in the network is 0.00845. However, according to basic modularity optimization, node u would be assigned to community C_2, because this partition of the network achieves a larger modularity value of 0.01203. This phenomenon is precisely due to the resolution limit of modularity optimization.

We run the proposed CLA-net algorithm in comparison with the CNM and MIGA on the test network 1000 times. Since the CNM and MIGA solely depend on the modularity optimization, they always incorrectly identify node u as a member of community C_2. On the contrary, our CLA-net algorithm can always find the correct communities in the network. The experimental results demonstrate that our algorithm effectively solves the resolution limit of modularity optimization.

5.2. Real-world networks

We also compare the proposed CLA-net algorithm with other algorithms on several real-world networks which are widely used in the community detection literature. General information of these real-world networks is shown in Table 1. Since the community structures of most real-world networks are unknown, *modularity* is used to measure the quality of the communities obtained by the algorithms. The average performances of the CLA-net algorithm alongside other algorithms on real-world networks are shown in Table 2.

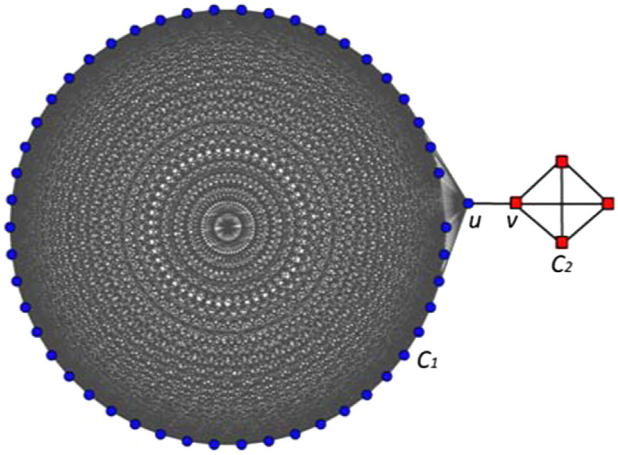


Fig. 5. A schematic of the test network for the evaluation of resolution limit. In the network, different communities are rendered in different shapes.

As shown in Table 2, the CLA-net algorithm can find the community structure with the largest Q value on dolphin network, football network and SFI network. It also has the best average performances among all the algorithms on dolphin network and football network.

For karate network, the MOGA-net and the MOEA/D-net achieve the largest Q value. The modularity of the community structure obtained by the CLA-net algorithm is only 0.001 smaller than the largest value. This slight difference is due to the membership of a fuzzy node. Since this node has equal connections to the core nodes of two different communities, it can be either classified to the first community or to the second one.

On netscience network and powergrid network, the performances of MIGA and Meme-net are absent, since they are very time-consuming and need too much time to converge. The CLA-net algorithm is only inferior to the CNM, but outperforms the other algorithms. The CNM shows obvious superiority over all the other algorithms from the perspective of modularity. However, many small communities in the network cannot be identified by the CNM due to the resolution limit. Our algorithm finds about 350 communities in netscience network and 600 communities in powergrid network, while the CNM only identifies 215 communities in netscience network and 40 communities in powergrid network. Therefore, our algorithm can discover some small communities ignored by the CNM.

From the above experiments on real-world networks, we can see that the CLA-net algorithm is promising and effective for community detection in complex networks in real applications.

5.3. Evaluation of time complexity

Finally, we experimentally measure the time complexity of the CLA-net algorithm. The execution time of the algorithm CLA-net is compared with that of several other community detection algorithms including GA-net, MOCD, MOGA-net, MOEA/D-net and CNM. The MIGA and Meme-net are not involved in this experiment because they are very time-consuming. For the GA-net, MOCD, MOGA-net and MOEA/D-net, the population size is set to 100 and the maximum number of generation is set to 500. The LFR benchmark networks are still used in our experiments. For the test benchmark networks, the mixing parameter μ is set to 0.2, average node degree is set to 20 and the community size is in the range from 10 to 50. The number of edges in the network varies from 1000 to 10,000. The execution time and the iteration number of the CLA-net algorithm on benchmark networks are shown in Fig. 6, where each data point is an average over 10 networks.

As is shown from Fig. 6, the CNM exhibits the lowest time complexity among all the algorithms in our experiments. Our CLA-net algorithm is only inferior to the CNM and faster than the other algorithms. The time of the CLA-net algorithm increases slightly worse than linearly because the iteration number slowly increases along with the number of edges. Therefore, the time complexity of the CLA-net algorithm is in correspondence with the previous analysis in Section 4.4.

Table 1
General information of the real-world networks.

Network	Description	Node	Edge
Karate	Zachary's karate club [49]	34	78
Dolphins	Lusseau's dolphins [50]	62	159
Football	American College football union [46]	115	616
SFI	Collaboration network of scientists at the Santa Fe Institute [46]	118	200
Netscience	Coauthorship network of scientists working on network theory [51]	1589	2742
Powergrid	The topology of the Power Grid of the United States [52]	4941	6594

Table 2
Maximum and average modularity of CLA-net, CNM, MIGA, Meme-net, GA-net, MOCD, MOGA-net and MOEA/D-net on real-world networks.

Network		CLA-net	CNM	MIGA	Meme-net	GA-net	MOCD	MOGA-net	MOEA/D-net
Karate	Q_{max}	0.4188	0.3807	0.4188	0.4020	0.4059	0.4188	**0.4198**	**0.4198**
	Q_{av}	0.4175	0.3807	0.3952	0.3857	0.4059	0.4188	0.4158	**0.4198**
Dolphin	Q_{max}	**0.5277**	0.4955	0.5210	0.5155	0.5014	0.5259	0.5258	0.5210
	Q_{av}	**0.5268**	0.4938	0.4629	0.4838	0.4946	0.5210	0.5215	0.5189
Football	Q_{max}	**0.6046**	0.5733	0.5911	0.5888	0.5940	0.5958	0.5280	0.6044
	Q_{av}	**0.6042**	0.5706	0.5478	0.5512	0.583	0.5785	0.5173	0.6032
SFI	Q_{max}	**0.7486**	0.7355	0.7363	0.7097	0.5867	0.7483	0.7430	0.7312
	Q_{av}	0.7404	0.7334	0.6725	0.6818	0.5748	**0.7474**	0.7323	0.7211
Netscience	Q_{max}	0.9346	**0.9555**	–	–	0.8581	0.8923	0.8916	0.9143
	Q_{av}	0.9177	**0.9554**	–	–	0.8473	0.8886	0.8810	0.9060
Powergrid	Q_{max}	0.7505	**0.9345**	–	–	0.6660	0.7065	0.7035	0.6880
	Q_{av}	0.7350	**0.9338**	–	–	0.6571	0.7003	0.6949	0.6815

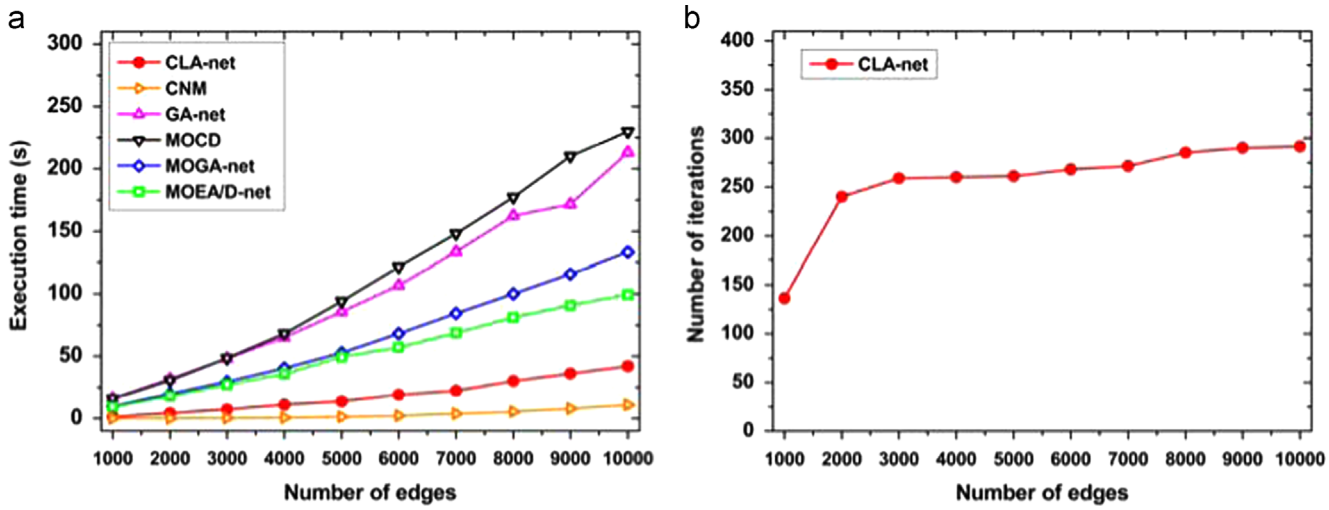


Fig. 6. (a) Average execution time of the CLA-net, CNM, GA-net, MOCD, MOGA-net and MOEA/D-net on benchmark networks of different sizes. (b) Average number of iterations of the CLA-net on benchmark networks of different sizes.

6. Conclusion

In this paper, we propose the CLA-net algorithm to solve the problem of community detection in complex network. In our algorithm, the whole network is modeled as an irregular cellular learning automata (ICLA) and the solution is constituted by current actions chosen by the learning automata in the network. At each cycle, every learning automaton chooses an action depending on its action probability vector and then updates the action probability vector according to the response from the environments using the CP_{RP} algorithm. Through a series of interactions with both the global environment and the local environments, the cellular learning automata makes an approach to the optimal community structure without resolution limit.

The CLA-net algorithm introduces the cellular learning automata to solve the problem of community detection in complex network. It requires no prior information about the network and effectively solves the resolution limit of modularity optimization. We test the CLA-net algorithm along with several other community detection algorithms on both synthetic and real-world networks for comparison. The experimental results demonstrate the effectiveness and efficiency of our proposed algorithm.

We only deal with disjoint communities in this paper. However, many real-world networks, especially social networks, have overlapping community structure. In this case, a node may belong to multiple communities. Therefore, extending the CLA-net algorithm to overlapping community structure will be the focus of our work in the future.

Acknowledgements

This work is funded by National Science Foundation of China (61271316 and 61071152), 973 Program (2010CB731403, 2010CB731406 and 2013CB329605), Chinese National "Twelfth Five-Year" Plan for Science & Technology Support (2012BAH38 B04) and Outstanding Innovative Talent Program Foundation of Henan Province (134200510025). This work is also supported by the Project of the Distinguished Professorship in Henan Province China for Professor De-Shuang Huang, Key Laboratory for Shanghai Integrated Information Security Management Technology Research and Chinese National Engineering Laboratory for Information Content Analysis Technology.

References

[1] S.H. Strogatz, Exploring complex networks, Nature 410 (2001) 268–276.
[2] R. Albert, A.L. Barabási, Statistical mechanics of complex networks, Rev. Mod. Phys. 74 (2002) 47–97.
[3] M.E.J. Newman, Detecting community structure in networks, Eur. Phys. J. B 38 (2004) 321–330.
[4] S. Fortunato, Community detection in graphs, Phys. Rep. 486 (2010) 75–174.

[5] J. Xie, S. Kelley, B.K. Szymanski, Overlapping community detection in networks: the state-of-the-art and comparative study, ACM Comput. Surv. 45 (2013) 43.
[6] M.E.J. Newman, The structure and function of complex networks, SIAM Rev. 45 (2003) 167–256.
[7] M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2004) 026113.
[8] M.E.J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (2004) 066133.
[9] A. Clauset, M.E.J. Newman, C. Moore, Finding community structure in very large networks, Phys. Rev. E 70 (2004) 066111.
[10] J. Duch, A. Arenas, Community detection in complex networks using extremal optimization, Phys. Rev. E 72 (2005) 027104.
[11] G.K. Kumar, V. K. Jayaraman, Clustering of Complex Networks and Community Detection Using Group Search Optimization, arXiv:1307.1372 [cs.NE].
[12] R. Shang, J. Bai, L. Jiao, C. Jin, Community detection based on modularity and an improved genetic algorithm, Physica A 392 (2013) 1215–1231.
[13] S. Fortunato, M. Barthélemy, Resolution limit in community detection, Proc. Natl. Acad. Sci. USA 104 (2007) 36–41.
[14] A. Arenas, A. Fernández, S. Gomez, Analysis of the structure of complex networks at different resolution levels, New J. Phys. 10 (2008) 053039.
[15] Z. Li, S. Zhang, R.S. Wang, X.S. Zhang, L. Chen, Quantitative function for community detection, Phys. Rev. E 77 (2008) 036109.
[16] C. Pizzuti, GA-Net: a genetic algorithm for community detection in social networks, In: Parallel Problem Solving from Nature (PPSN), Springer, Berlin, 2008, pp. 1081–1090.
[17] A. Lancichinetti, S. Fortunato, J. Kertesz, Detecting the overlapping and hierarchical community structure in complex networks, New J. Phys. 11 (2009) 033015.
[18] C. Shi, Z.Y. Yan, Y.N. Cai, B. Wu, Multi-objective community detection in complex networks, Appl. Soft Comput. 12 (2012) 850–859.
[19] C. Pizzuti, A multiobjective genetic algorithm to find communities in complex networks, IEEE Trans. Evolut. Comput. 16 (2012) 418–430.
[20] M. Gong, L. Ma, Q. Zhang, L. Jiao, Community detection in networks by using multiobjective evolutionary algorithm with decomposition, Physica A 391 (2012) 4050–4060.
[21] M. Gong, X. Chen, L. Ma, Q. Zhang, L. Jiao, Identification of multi-resolution network structures with multi-objective immune algorithm, Appl. Soft Comput. 13 (2013) 1705–1717.
[22] M.R. Meybodi, H. Beigy, M. Taherkhani, Cellular learning automata and its applications, Sharif J. Sci. Technol. 19 (2003) 54–77.
[23] H. Beigy, M.R. Meybodi, A Mathematical Framework for Cellular Learning Automata, Adv. Complex Syst. 7 (2004) 295–319.
[24] M.R. Meybodi, M.R. Kharazmi, Application of cellular learning automata to image processing, J. Amirkabir 14 (2004) 1101–1126.
[25] D.K. Patel, S.A. More, Edge detection technique by fuzzy logic and cellular learning automata using fuzzy image processing, in: Proceedings of 2013 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2013, pp. 1–6.
[26] M. Esnaashari, M.R. Meybodi, Dynamic point coverage problem in wireless sensor networks: a cellular learning automata approach, Ad Hoc Sens. Wirel. Netw. 10 (2010) 193–234.
[27] M. Esnaashari, M.R. Meybodi, Deployment of a mobile wireless sensor network with k-coverage constraint: a cellular learning automata approach, Wirel. Netw. 19 (2013) 945–968.
[28] H. Mohamadi, A.S.B.H. Ismail, S. Salleh, A learning automata-based algorithm for solving coverage problem in directional sensor networks, Computing 95 (2013) 1–24.
[29] R. Vafashoar, M.R. Meybodi, CLA-DE: a hybrid model based on cellular learning automata for numerical optimization, Appl. Intell. 36 (2012) 735–748.
[30] J.A. Torkestani, M.R. Meybodi, A cellular learning automata-based algorithm for solving vertex coloring problem, Exp. Syst. Appl. 38 (2011) 9237–9247.
[31] A. Piwonska, F. Seredynski, M. Szaban, Learning cellular automata rules for binary classification problem, J. Supercomput. 63 (2013) 800–815.
[32] P.V. Krishna, S. Misra, D. Joshi, M.S. Obaidat, Learning automata based sentiment analysis for recommender system on cloud, in: Proceedings of 2013 International Conference on Computer, Information and Telecommunication Systems (CITS), Piraeus-Athens, Greece, 2013, pp. 1–5.
[33] M. Mozafari, R. Alizadeh, A cellular learning automata model of investment behavior in the stock market, Neurocomputing 122 (2013) 470–479.
[34] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, D. Parisi, Defining and identifying communities in networks, Proc. Natl. Acad. Sci. USA 101 (2004) 2658–2663.
[35] U.N. Raghavan, R. Albert, S. Kumara, Near linear time algorithm to detect community structures in large-scale networks, Phys. Rev. E 76 (2007) 036106.
[36] S. Wolfram, Statistical mechanics of cellular automata, Rev. Mod. Phys. 55 (1983) 601–644.
[37] M.A.L. Thathachar, P.S. Sastry, Varieties of learning automata: an overview, IEEE Trans. Syst. Man Cybern. B 32 (2002) 711–722.
[38] B.J. Oommen, Recent advances in Learning Automata systems, in: Proceedings of the 2nd International Conference on Computer Engineering and Technology (ICCET), Chengdu, China, 2010, pp. 724–735.
[39] A.V. Vasilakos, G.I. Papadimitriou, A new approach to the design of reinforcement schemes for learning automata: Stochastic estimator learning algorithm, Neurocomputing 7 (1995) 275–297.
[40] X. Zhang, O.-C. Granmo, B.J. Oommen, The Bayesian pursuit algorithm: a new family of estimator learning automata, in: Proceedings of the 25th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA-AIE 2011), NY, USA, pp 608–620.
[41] X. Zhang, O.-C. Granmo, B.J. Oommen, Discretized Bayesian pursuit—a new scheme for reinforcement learning, in: Proceedings of the 25th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA-AIE 2012), Dalian, China, 2012, pp. 784–793.
[42] H. Beigy, M.R. Meybodi, Open synchronous cellular learning automata, Adv. Complex Syst. 10 (2007) 527–556.
[43] M. Esnaashari, M.R. Meybodi, Irregular cellular learning automata and its application to clustering in sensor networks, in: Proceedings of the 15th Conference on Electrical Engineering (ICEE), Volume on Communication, Telecommunication Research Center, Tehran, Iran, 2007, pp. 14–28.
[44] Y.J. Park, M.S. Song, A genetic algorithm for clustering problems, in: Proceedings of the 3rd Annual Conference on Genetic Algorithms, 1989, pp. 2–9.
[45] M. Gong, B. Fu, L. Jiao, Memetic algorithm for community detection in networks, Phys. Rev. E 84 (2011) 056101.
[46] M. Girvan, M.E.J. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. USA 99 (2002) 7821–7826.
[47] A. Lancichinetti, S. Fortunato, F. Radicchi, Benchmark graphs for testing community detection algorithms, Phys. Rev. E 78 (2008) 046110.
[48] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (2007) 712–731.
[49] W.W. Zachary, An information flow model for conflict and fission in small groups, J. Anthropol. Res. 33 (1997) 452–473.
[50] D. Lusseau, The emergent properties of a dolphin social network, Proc. R. Soc. Lond. B 270 (2003) S1860–1888.
[51] M.E.J. Newman, Finding community structure in networks using the eigenvectors of matrices, Phys. Rev. E 74 (2006) 036104.
[52] D.J. Watts, S.H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (1998) 440–442.

Yuxin Zhao received the B.S. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2010. He won the First Class Scholarship of Shanghai Jiao Tong University, in 2008. He is currently pursuing his Ph.D. degree in electronic engineering at Shanghai Jiao Tong University. His research interests include complex network, data mining and information security.

Wen Jiang received the B.Sc. and M.S. degree in automation from Henan University, Kaifeng, China, in 2009 and in pattern recognition and intelligent system at the University of Science and Technology of China, Hefei, China, in 2012, respectively. He was selected as a straight "A" student of Henan province and won the National Inspiration Scholarship, the National Scholarship, and three times the First Class Scholarship of Henan University.

He is currently pursuing his Ph.D. degree with Department of Electronic Engineering, Shanghai Jiao Tong University. His research interests include learning automata and their applications, time series analysis, pattern recognition, data mining (big data), text and Web mining, machine learning, and hybrid intelligent systems.

Prof. Shenghong Li received the B.S. and the M.S. degrees in electrical engineering from Jilin University of Technology, China, in 1993 and 1996 respectively, and received the Ph.D. degree in radio engineering from Beijing University of Posts and Telecommunications, China, in 1999. Since September 1999, he has been working in Shanghai Jiao Tong University, China, as a research fellow, an associate professor and a professor, successively. In 2010, he worked as a visiting scholar in Nanyang Technological University, Singapore. His research interests include information security, signal and information processing, artificial intelligence. He published more than 80 papers, coauthored four books, and holds ten granted patents. In 2003, he received the 1st Prize of Shanghai Science and Technology Progress in China. In 2006 and 2007, he was elected for New century talent of Chinese Education Ministry and Shanghai dawn scholar.

Yinghua Ma received the B.S. and the M.S. degrees in computer science from Shandong University, China, in 1993 and 1996 respectively, and received the Ph.D. degree in computer science from Shanghai Jiao Tong University, China, in 2004. She has been working in School of Information Security Engineering, Shanghai Jiaotong University, China, as a research fellow since 2005. Her research interests include semantic analysis and algorithm design.

Guiyang Su received the Ph.D. degree in computer science from Shanghai Jiao Tong University, China, in 2004. He has been working in School of Information Security Engineering, Shanghai Jiaotong University, China, as a research fellow since 2005. His research interests include semantic analysis and algorithm design.

Xiang Lin received the B.S. in automation from Shandong University of Electric Power, China, in 2002, and received the M.S. degree in communication and information system from Shanghai Jiao Tong University, China, in 2005. He has been working in School of Information Security Engineering, Shanghai Jiaotong University, China, as a research fellow since 2005. His research interests include information security and public opinion analysis.