A michigan memetic algorithm for solving the community detection problem in complex network

Mehdi Rezapoor Mirsaleh¹ and Mohammad Reza Meybodi²

¹²Soft Computing Laboratory, Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran
e-mail: (mrezapoorm, mmeybodi)@aut.ac.ir

Abstract

Community structure is an important feature in complex networks which has great significant for organization of networks. The community detection is the process of partitioning the network into some communities in such a way that there exist many connections in the communities and few connections between them. In this paper a michigan memetic algorithm; called MLAMA-Net; is proposed for solving the community detection problem. The proposed algorithm is an evolutionary algorithm in which each chromosome represents a part of the solution and the whole population represents the solution. In the proposed algorithm, the population of chromosomes is a network of chromosomes which is isomorphic to the input network. Each node has a chromosome and a learning automaton (LA). The chromosome represents the community of corresponding node and saves the histories of exploration. The learning automaton represents a meme and saves the histories of the exploitation. The proposed algorithm is a distributed algorithm in which each chromosome locally evolves by evolutionary operators and improves by a local search. By interacting with both the evolutionary operators and local search, our algorithm effectively detects the community structure in complex networks and solves the resolution limit problem of modularity optimization. To show the superiority of our proposed algorithm over the some well-known algorithms, several computer experiments have been conducted. The obtained results show MLAMA-Net is effective and efficient at detecting the community structure in complex networks.

Keywords: Complex Networks, Community Detection Problem, Learning Automata (LA), Memetic Algorithm (MA).
1. Introduction

Community structure in an important feature of complex networks which is used for understanding the function and organization of complex network. A community or a partition refers to a sub graph such that the nodes in sub graph are more densely connected internally than to the rest of graph. Community detection problem is a well-known problem in complex networks which is used in sociology, biology and computer science where systems are modeled by graphs. Newman and Girvan was defined modularity as an important measure function for evaluating the quality of community structure in complex networks [1]. Modularity is based on the idea that no community structure is expected to be found in a random network. On the basis of modularity concept, community detection can be modeled as a modularity optimization problem. Due to the wide spread of applications of community detection, the numerous optimization algorithms have been proposed to solve this problem in recent years. Genetic algorithm (GA) is one of the famous methods for optimization problem. Genetic algorithms belong to the larger class of evolutionary algorithms (EAs) which generate solutions for optimization problem using techniques inspired by natural evolution. Genetic algorithms can be divided into two broad approaches, michigan and pittsburgh approaches. In pittsburg approach, one of the chromosomes in the population becomes the solution of the problem being solved whereas in michigan approach the whole population represents the solution. For example in the community detection problem, in michigan approach, each chromosome encodes a community for a node and the set of all chromosomes in the population represents the solution which is a partitioning for the network whereas in the pittsburgh approach each chromosome encodes a partitioning for the whole network. Combining a genetic algorithm with a local search produces a type of evolutionary algorithm which is known as a memetic algorithm (MA) [2]. Memetic algorithm uses the local search to either accelerate the discovery of good solutions, for which evolution alone would take too long to discover, or to reach solutions that would otherwise be unreachable by evolution or a local search alone. Learning automata are based on the general schemes of reinforcement learning algorithms. They select actions via a stochastic process and apply them on a random unknown environment. They can learn the best action by iteratively performing and receiving stochastic reinforcement signals from the unknown environment. These stochastic responses from the environment show the favorability of the selected actions, and the learning automata change their action selecting mechanism in favor of the most promising actions according to responses from the environment.
In the first part of this paper we present a new algorithm based on the michigan memetic algorithm called MLAMA-Net, for detecting community structure in network. In the proposed algorithm, the population of chromosomes is a network of chromosomes which is isomorphic to the input network. The chromosome in each node of the network represents a part of the solution and the whole population represents the solution. When the algorithm terminates the chromosome in each node represents a community for its corresponding node and the whole population represents a partitioning for the network. Each node of network is also equipped with a learning automaton which is utilized to save the histories of local search. The proposed algorithm is a distributed algorithm in which each chromosome locally explores a community for its corresponding node by evolutionary operators and improves its community by a local search. MLAMA-Net introduces the priority concept to solve the resolution limit of modularity optimization in community detection problem. In the second part of the paper, the proposed algorithm is compared with other well-known algorithms for the community detection problem on two groups of networks which are used in the literature: real world networks and synthetic networks. Computer simulations show that the proposed algorithm outperforms other algorithms in terms of quality of solution and accuracy of solution in both real world networks and synthetic networks.

The rest of the paper is organized as follows. The theory of learning automata is described in section 2. In section 3, an overview of related works on community detection problem is represented. The new proposed algorithm is described in Section 4. Sections 5 is including of implementation considerations, simulation results, and comparison with other algorithms to highlight the contributions of the new algorithm. Finally, conclusions and future works are discussed in section 6.

2. Theory of learning automata

A learning automaton [3] is an adaptive decision-making unit. It can be described as determination of an optimal action from a set of actions through repeated interactions with an unknown random environment. It selects an action based on a probability distribution at each instant and applies it on a random environment. The environment sends a reinforcement signal to the automaton after evaluating the input action. The learning automaton processes the response
of environment and updates its action probability vector. By repeating this process, the automaton learns to choose the optimal action so that the average penalty obtained from the environment is minimized. The environment is represented by a triple \(<a, \beta, \zeta>\) where \(a = \{a_1, ..., a_r\}\) is the finite set of the inputs, \(\beta = \{\beta_1, ..., \beta_m\}\) is the set of outputs that can be taken by the reinforcement signal, and \(\zeta = \{c_1, ..., c_r\}\) is the set of the penalty probabilities, where each element \(c_i\) of \(\zeta\) is associated with one input action \(a_i\). When the penalty probabilities are constant, the random environment is said a stationary random environment. It is called a non stationary environment, if they vary with time. Depending on the nature of the reinforcement signal, there are three types of environments: P-model, Q-model and S-model. The environments, in which the output can take only one of two values 0 or 1, are referred to as P-model environments. The reinforcement signal in Q-model environment selects a finite number of the values in the interval \([a, b]\). When the output of environments is a continuous random variable in the interval \([a, b]\), it is referred to as S-model. The relationship between the learning automaton and the random environment is shown in Fig. 1.

![Fig. 1: The relationship between the learning automaton and random environment](image)

There are two main families of Learning automata [4]: fixed structure learning automata and variable structure learning automata. Variable structure learning automata are represented by a triple \(<\beta, a, T>\), where \(\beta\) is the set of inputs, \(a\) is the set of output actions, and \(T\) is learning algorithm which is used to modify the action probability vector. Learning algorithm is the critical factor affecting the performance of variable structure learning automata. Suppose learning automaton selects action \(a_i(k) \in a\) according to action probability vector \(p(k)\) at instant \(k\). The
action probability vector \( p(k) \) is updated by the learning algorithm given in Eq.(1), if the selected action \( \alpha_i(k) \) is rewarded by the random environment, and it is updated as given in Eq. (2), if the taken action is penalized. \( a \) and \( b \) denote the reward and penalty parameters and \( r \) is the number of actions that can be taken by learning automaton.

\[
P_j(n + 1) = \begin{cases} 
  P_j(n) + a(1 - P_j(n)) & j = i \\
  (1 - a)P_j(n) & \forall j, j \neq i
\end{cases} \tag{1}
\]

\[
P_j(n + 1) = \begin{cases} 
  (1 - b)P_j(n) & j = i \\
  b/(1 - r) + (1 - b)P_j(n) & \forall j, j \neq i
\end{cases} \tag{2}
\]

If \( a = b \), the recurrence equations (1) and (2) are called linear reward-penalty (\( L_{R-p} \)) algorithm, if \( a \gg b \) the given equations are called linear reward-\( \epsilon \) penalty (\( L_{R \epsilon p} \)), and finally if \( b = 0 \) they are called linear reward-Inaction (\( L_{R-I} \)). In the latter case, the action probability vector remains unchanged when the taken action is penalized by the environment.

Learning automata have a vast variety of applications in combinatorial optimization problems [5-10], computer networks [8, 11-14], queuing theory [15, 16], signal processing [17, 18], information retrieval [19, 20], adaptive control [21-23], neural networks engineering [24, 25] and pattern recognition [26-28].

### 3. Related work

The community detection problem can be considered as a modularity optimization problem [1]. A network can be modeled by a graph \( G = (V, E) \) where \( V \) is the set of \( |V| = n \) nodes and \( E = \{(i, j)| i, j \in V \} \) is the set of \( |E| = m \) edges. The modularity can be written as

\[
Q = \frac{1}{2m} \sum_{i,j \in V} \left( A_{ij} - \frac{k_i k_j}{2m} \right) \delta(i,j) \tag{3}
\]

Where \( A_{ij} \) is the element of the adjacency matrix \( A \), i.e., \( A_{ij} = 1 \) if node \( v_i \) is connected to node \( v_j \); otherwise, \( A_{ij} = 0 \), \( m \) is the number of edges, \( k_i = \sum_{j \in V} A_{ij} \) is the degree of node \( v_i \) and \( \delta(.) \) is the delta function, i.e., \( \delta(i,j) = 1 \) if node \( v_i \) and node \( v_j \) are in the same community; otherwise, \( \delta(i,j) = 0 \). The term \( \frac{k_i k_j}{2m} \) represents the expected number of edges between node \( v_i \)
and node $v_j$ in a random network with the same size and nodes degree distribution. The modularity with larger value shows the more significant community structure in given graph. However, resolution limit problem is the main problem in modularity optimization based algorithm that they may fail to find communities which are smaller than a certain size. Many approaches [29, 30] have been proposed to solve the resolution limit problem by definition the other quality metrics. Most of these approaches need a tunable parameter to determine the resolution level of community structure. The formulation the community detection problem as a multi objective optimization problem is another approach to overcome the resolution limit problem [31, 32]. The multi objective optimization based algorithms search the optimal solutions by optimizing multiple objective functions that evaluate the community structure from different views. Due to the wide spread of applications of community detection, the wide variety algorithms have been reported for solving the community detection problem in the literature. A greedy algorithm, called FN, is proposed in [33] for community detection problem. The FN algorithm starts from a set of isolated nodes in which each node is a unique community and then iteratively connects a pair of communities with the maximum value of modularity at each step. In [34] an algorithm, called CNM, was proposed for detecting community structure in network which works based on the optimization of modularity and uses a modern data structure to reduce the calculation complexity of modularity. CNM algorithm is faster than FN algorithm and allows us to extend community structure analysis to large networks.

A memetic algorithm, called Meme-Net, which is based on genetic algorithm and a hill climbing strategy as the local search was proposed in [35]. Meme-Net optimizes another quality function, modularity density, which includes a tunable parameter that allows one to explore the network at different resolutions.

A genetic algorithm [36], called GA-Net, was proposed to discover communities in social networks. GA-Net optimizes an efficient fitness function to identify densely connected communities of nodes with sparse connections between communities. GA-Net is efficient because the variation operators are modified to take into consideration only the actual correlations among the nodes.

In [31] a multi-objective evolutionary algorithm, called MOCD, was proposed for community detection problem. After analyzing and comparing a variety of objective functions that have been
used for community detection, MOCD exploits the concept of correlation between objectives which characterizes the relationship between any two objective functions.

A multi-objective genetic algorithm to uncover community structure in complex network, called MOGA-Net, was proposed in [32]. MOGA-Net optimizes two objective functions able to identify densely connected communities of nodes having sparse inter connections. MOGA-Net generates a set of network divisions at different hierarchical levels in which solutions at deeper levels are contained in solutions having a lower number of communities. The number of modules is automatically determined by the objective functions.

In [37] a multi-objective evolutionary algorithm, called MOEAD-Net, was proposed for solving the community detection in networks. MOEAD-Net optimizes two conflicting objective functions decomposed from modularity density. The algorithm maximizes the density of internal degrees, and minimizes the density of external degrees simultaneously.

In [38] a hybrid community detection algorithm based on the modularity optimization and an improved genetic algorithm, called MIGA, was proposed. MIGA uses modularity as objective function and takes the simulated annealing method as the local search method.

An algorithm based on open cellular learning automata, called CLA-Net, was proposed for solving the community detection problem in [39]. In CLA-Net the whole network is modeled as irregular cellular learning automata (ICLA) in which the solution is constituted by current actions chosen by the learning automata in the network. CLA-Net effectively solves the resolution limit of modularity optimization by interacting with both the global and local environments.

4. The proposed algorithm for community detection problem

In this section, we propose a new evolutionary algorithm called MLAMA-Net for solving the community detection problem. The proposed algorithm is a hybrid algorithm based on the memetic algorithm and learning automata. In this algorithm the chromosomes which are represented on the basis of michigan approach are associated to the nodes of network. For this purpose an initial population isomorphic to input network is created. To construct initial population, each network node is equipped with a chromosome, and then a learning automaton is assigned to it. The chromosome represents the community of corresponding node and saves the
histories of exploration and the learning automaton represents a meme and saves the histories of the exploitation. Each node $v_i$ of network can be modeled by a duple $< CR^i, M^i(t) >$ where $CR^i$ is a chromosome which represents the community of node $v_i$ by an integer number and $M^i(t)$ is a meme which save the effect (history) of the local search on the chromosome $CR^i$ at generation $t$. Initial chromosome $CR^i$ is created randomly by selecting a random integer number from set $\zeta = \{c_1, c_2, ..., c_n\}$ as the set of all possible communities. At the beginning of each generation the evolutionary operators are applied on chromosome $CR^i$. First, mutation operator is applied on chromosome $CR^i$ with rate $r_m$ in which the value of chromosome $CR^i$ is replaced by other value of set $\zeta = \{c_1, c_2, ..., c_n\}$. Then, crossover operator is performed on chromosome $CR^i$ and one of its neighbors, which is selected randomly, with rate $r_c$ in which the value of chromosome $CR^i$ is exchanged with the value of selected chromosome. Let $GF$ be the fitness function which is used to evaluate the fitness of a chromosome based on its genotype and genotypes of its adjacent chromosomes. The fitness of chromosome $CR^i$ at generation $t$, which is referred to as genetic fitness, is denoted by $GF^i(t)$. The genetic fitness of chromosome $CR^i$ associated to node $v_i$ of graph $G = (V, E)$ is calculated as follows:

$$GF^i(t) = \frac{1}{2m_t} \sum_{i,j \in N_i} \left( A_{ij} - \frac{k_i k_j}{2m_t} \right) \delta(i, j)$$

(4)

Where $N_i = \{u | [u, v_i] \in E\}$ is the set of neighbors of node $v_i$, $k_i$ is the degree of node $v_i$, $A$ is a binary matrix where indicates the adjacency of chromosomes in which $A_{ij} = 1$, if chromosome $CR^i$ is adjacent to chromosome $CR^j$; otherwise, $A_{ij} = 0$ and $m_t = \sum_{i,j \in N_i} A_{ij}$. The term $\delta(.)$ is the delta function, i.e., $\delta(i, j) = 1$ if node $v_i$ and node $v_j$ are in the same community; otherwise, $\delta(i, j) = 0$.

The effect (history) of the local search on chromosome $CR^i$ at generation $t$ is represented by meme $M^i(t)$ where is equipped with a learning automaton $LA_i$ in which $\underline{a}_i = \{c_1, c_2, ..., c_n\}$ is the set of actions (communities) that can be taken by learning automaton $LA_i$. The effect of the local search on chromosome $CR^i$ at generation $t$ is represented by the action probability vector of learning automaton in the meme $M^i(t)$ as given by Eq. (5).

$$M^i(t) = [M_1^i(t), M_2^i(t), ..., M_n^i(t)]$$

(5)

where
\[ 1 \leq i \leq n \text{ and } \forall \, i, \sum_{k=1}^{n} M_k^i(t) = 1. \]

\( M_k^i(t) \) denotes the probability that action \( k \) of leaning automaton in the meme \( M^i(t) \) is selected in the exploitation process. In other words, \( M_k^i(t) \) is the probability that community \( c_k \) is selected by local search for node \( v_i \). \( M_k^i(0) \) where \( 1 \leq i, k \leq n \) is initially set to \( 1/n \). Updating the action probability vector of learning automaton associated to meme \( M^i(t) \) is performed on the basis of the result of applying the local search on the chromosome \( CR^i \) as described in next paragraph. Let \( MF \) be a function which is used to evaluate the effect (history) of local search on a chromosome. The effect of the local search on chromosome \( CR^i \) at generation \( t \); which is referred to as memetic fitness; is denoted by \( MF^i(t) \). The memetic fitness of chromosome \( CR^i \) is calculated as follows:

\[ MF^i(t) = M_k^i(t), \tag{6} \]

where \( k \) is the action of learning automaton in the meme \( M^i(t) \) which corresponds to the value of chromosome \( CR^i \). Memetic fitness of a chromosome changes when the action probability vector of learning automaton in its corresponding meme is updated. Updating is performed on the basis of the result of applying the local search on a chromosome. It is worth noting that, local search changes only the action probability vector of the meme not value of the chromosome. That is, local search only changes the memetic fitness not the genetic fitness. Local search is applied on chromosome \( CR^i \) based on the genetic information (genotype and genetic fitness) and memetic information (action probability vector and memetic fitness) of chromosome \( CR^i \) and the genetic and memetic information of its adjacent chromosomes.

Let \( \alpha_i \) be the community of node \( v_i \) which is represented by chromosome \( CR^i \), \( N_i = \{ u \mid [u, v_i] \in E \} \) be the set of neighbors of node \( v_i \), \( N_i(c) = \{ j \in N_i : \alpha_j = c \} \) be the set of neighbors of node \( v_i \) with community \( c \) and \( |N_i(c)| \) be the number of neighbors of node \( v_i \) with community \( c \). At generation \( t \), the action \( \alpha_i \) of learning automaton associated to meme \( M^i(t) \) is rewarded, if community of node \( v_i \) has the highest priority amongst its neighbors' communities. Otherwise, it will be penalized.

The priority of node \( v_i \) can be described as follows:

\[
\text{priority}(v_i) = \begin{cases} 
1 & \text{if } (|N_i(\alpha_i)| > |N_j(\alpha_j)|) \forall \, j \in N_i \\
1 & \text{if } (|N_i(\alpha_i)| = |N_j(\alpha_j)|) \text{ and } (MF^i(t) > MF^j(t)) \forall \, j \in N_i \\
0 & \text{otherwise}
\end{cases} \tag{7}
\]
Where $MF^i(t)$ and $MF^j(t)$ are the memetic fitness of chromosomes $CR^i$ and $CR^j$ at generation $t$, respectively. The priority concept can effectively overcome the resolution limit problem in the proposed algorithm. In the last step of proposed algorithm, learning automaton associated to meme $M^i(t)$ randomly chooses one of its actions and as a result a new chromosome is generated. If the genetic fitness of new chromosome be higher than genetic fitness of chromosome $CR^i$, the newly generated chromosome replaces the chromosome $CR^i$.

The community detection process continues (in parallel) for each node $v_i$, if the probability of an action of learning automaton associated to meme $M^i(t)$ exceeds a pre-specified threshold, e.g., $\pi_i$. The relationship between node $v_i$ and its neighbors is shown in Fig. 2

![Diagram](image)

Fig. 2: The relationship between node $v_i$ and its neighbors in MLAMA-Net algorithm

The proposed algorithm is a fully distributed algorithm in which each chromosome locally evolves based on its adjacent chromosomes and independent of the other chromosomes. The operation of the proposed algorithm can be described as follow. Initial chromosomes are created randomly and the probability of selecting an action for all learning automata is set to $1/n$. The proposed algorithm is progressed in a number of generations as long as the termination criteria
are not satisfied. Each generation is divided to three phases: exploration phase, exploitation phase and memetic effect phase. In exploration phase the mutation and the crossover operators are applied on chromosome $CR^i$ with rates $r_m$ and $r_c$ respectively. In exploitation phase, local search is applied to chromosome $CR^i$, and then the action probability vector of the meme $M^i(t)$ (the history) is updated according to a learning algorithm. In memetic effect phase, chromosome $CR^i$ is replaced with a new chromosome which is generated based on the action probability vector of learning automaton associated to meme $M^i(t)$, if genetic fitness of new chromosome is higher than genetic fitness of chromosome $CR^i$. Pseudo code for proposed algorithm demonstrated in Fig. 3:

---

The proposed michigan memetic algorithm for community detection problem (MLAMA-Net)

```plaintext
1:  Input: Node $v_i$, threshold $\pi_i$
2:  Output: $k$
3:  Begin Algorithm
4:    $t=0$;
5:    Node $v_i$ is equipped with chromosome $CR^i$ and meme $M^i(t)$;
6:    Chromosome $CR^i$ is created randomly by selecting a random integer number from set \{$c_1, c_2, ..., c_n$\};
7:    Meme $M^i(t)$ is equipped with automatons $LA_i$;
8:    Automaton $LA_i$ forms its action set by \{$c_1, c_2, ..., c_n$\};
9:    While ($\max_k M^i_k(t) \leq \pi_i$)  
10:      //----------------------------------Exploration Phase----------------------------------
11:      If (random() < $r_m$)  
12:          Apply Mutation operator on chromosome $CR^i$;
13:      End If
14:      If (random() < $r_c$)  
15:          Select randomly chromosome $CR^j$ from adjacent of chromosome $CR^i$;
16:          Apply Crossover operator on chromosome $CR^i$ and chromosome $CR^j$;
17:      End If
18:      //----------------------------------Exploitation Phase----------------------------------
19:      If (Node $v_i$ has the highest priority amongst the its neighbors) Then
20:          Automaton $LA_i$ rewards the action $\alpha_i$;
21:      Else
22:          Automaton $LA_i$ penalizes the action $\alpha_i$;
23:      End If
24:      //----------------------------------Memetic Effect Phase----------------------------------
25:      Generate a new chromosome based on the action probability vector of meme ($M^i(t)$)
26:      If (Genetic fitness of new chromosome is higher than genetic fitness of chromosome $CR^i$)
27:          Replace chromosome $CR^i$ with new generated chromosome;
28:      End If
29:      $t=t+1$;
30:    End While
31:  End Algorithm
```

---

Fig. 3. Pseudo code for MLAMA-Net
5. Experimental Results

In this section several experiments are described that study the efficiency of proposed algorithm. The results of proposed algorithm is compared with the results of the best-known community detection algorithms such as: CNM [34], Meme-Net [35], GA-Net [36], MOCD [31], MOGA-Net [32], MOEAD-Net [37], MIGA [38] and CLA-Net [39]. To show the performances of the different algorithms, we use two standard measures consist of Normalized Mutual Information (NMI) [40] and modularity [1]. Normalized Mutual Information (NMI) which is used for networks with known community structure evaluates the similarity between real community structure and community structure found by the algorithm. The NMI can be written as

$$NMI(A, B) = \frac{-2 \sum_{a \in A} \sum_{b \in B} |a \cap b| \log \left( \frac{|a \cap b| \cdot n}{|a| \cdot |b|} \right)}{\sum_{a \in A} |a| \log \left( \frac{|a|}{n} \right) + \sum_{b \in B} |b| \log \left( \frac{|b|}{n} \right)}$$  \hspace{1cm} (8)

Where $a$ is a community in real community structure $A$ and $b$ is a community in obtained community structure $B$. The value of NMI is between $[0, 1]$ and the larger value means that the found community structure is more similar to real community structure. Modularity (Eq. (3)) which is used for networks with unknown community structure evaluates the quality of community structure in complex networks. The modularity with larger value shows the more significant community structure in given network. We compare the MLAMA-Net with other algorithms on two groups of networks which are used in the literature: real world networks and synthetic networks. The characteristics of the real world networks are given in Table 1. The modularity measure is used to evaluate the quality of the community structure obtained by different algorithms because the community structures of most these networks are unknown.

<table>
<thead>
<tr>
<th>Network</th>
<th>Number of nodes</th>
<th>Number of edges</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Karate</td>
<td>34</td>
<td>78</td>
<td>Zachary’s karate club</td>
</tr>
<tr>
<td>Dolphins</td>
<td>62</td>
<td>159</td>
<td>Lusseau’s dolphins</td>
</tr>
<tr>
<td>Book</td>
<td>105</td>
<td>441</td>
<td>A network of Books about US politics</td>
</tr>
<tr>
<td>Football</td>
<td>115</td>
<td>616</td>
<td>American College football union</td>
</tr>
<tr>
<td>Net science</td>
<td>1589</td>
<td>2742</td>
<td>Coauthor ship network of scientists working on network theory</td>
</tr>
<tr>
<td>Power grid</td>
<td>4941</td>
<td>6594</td>
<td>The topology of the Power Grid of the United States</td>
</tr>
</tbody>
</table>

GN [41] and LFR [42] are two synthetic benchmark networks which are used to compare proposed algorithm with other algorithms. In GN benchmark, each network was generated with
128 nodes which are partitioned to 4 communities of 32 nodes each. The average degree of a node is equal to 16. Each node shares a fraction $1 - \lambda$ of its edges with the other nodes in its community and a fraction $\lambda$ with the other nodes of network; $\lambda$ is the mixing parameter. The network with smaller value of $\lambda$ has more significant community structure. Fig. 4 demonstrates four samples of GN benchmark with different values for mixing parameter.

Fig. 4- Four samples of GN benchmark with 128 nodes and different mixing parameter values (a) $\lambda=0.05$, (b) $\lambda=0.10$, (c) $\lambda=0.15$ and (d) $\lambda=0.30$
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 [39]. Therefore, the LFR benchmark networks, which are more consistent with the properties of real-world networks were proposed in [42]. In LFR benchmark, the node degrees and the size of communities are taken from the power law distribution with exponent $\tau_1$ and $\tau_2$ respectively. Also, each node shares a fraction $1 - \mu$ of its edges with the other nodes in its community and a fraction $\mu$ with the other nodes of network; $\mu$ is the mixing parameter. The network with smaller value of $\mu$ has more significant community structure. Fig. 5 demonstrates four samples of LFR benchmark with different values for mixing parameter.
The NMI measure is used to evaluate the similarity between real community structure and community structure obtained by different algorithms for networks in GN and LFR benchmark. For all experiments the mutation rate is 0.05, crossover rate is 0.5, the action probability vectors of learning automata are updated according to $L_{R-I}$ learning algorithm with $a=0.5$ and algorithm is terminated (for all nodes) when the probability of an action of learning automaton is 0.95 or greater ($\pi_t = 0.95$).
Experiment 1

In this experiment we compared the results obtained from MLAMA-Net with the results of eight other algorithms, CNM algorithm [34], Meme-Net algorithm [35], GA-Net algorithm [36], MOCD algorithm [31], MOGA algorithm [32], MOEAD-Net algorithm [37], MIGA [38] and CLA-Net algorithm [39] for different real world networks described in Table 1, in terms of modularity measure. We use the modularity measure because the community structures of most real world networks are unknown. Table 2 presents the results of the different algorithms for 6 different real world networks with respect to the average and maximum values of modularity measure, standard deviation and the p-values of the two-tailed t test. Each reported result was averaged over 100 runs. We performed a parametric test (t test) at the 95% significance level to provide statistical confidence. The t tests were performed after ensuring that the data followed a normal distribution (by using the Kolmogorov–Smirnov test). From the results described in Table 2 we report the following:

- The MLAMA-Net algorithm can find the community structure with largest modularity value on karate network, dolphins network, book network, football network and power grid network. Although, in net-science network, the modularity value obtained by CNM algorithm is largest value, the modularity value obtained by proposed algorithm is 0.0005 smaller than the largest value.
- The difference between the performance of the MLAMA-Net and the performance of the other algorithms is statistically significant (p-value<0.05) in most cases.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Karate</td>
<td>Max. 0.4198</td>
<td>0.4188</td>
<td>0.3807</td>
<td>0.4188</td>
<td>0.4020</td>
<td>0.4059</td>
<td>0.4188</td>
<td>0.4198</td>
<td>0.4198</td>
</tr>
<tr>
<td></td>
<td>Avg. 0.4136</td>
<td>0.4175</td>
<td>0.3807</td>
<td>0.3952</td>
<td>0.3857</td>
<td>0.4059</td>
<td>0.4188</td>
<td>0.4158</td>
<td>0.4198</td>
</tr>
<tr>
<td></td>
<td>Std. 0.0102</td>
<td>0.0123</td>
<td>0.0234</td>
<td>0.0224</td>
<td>0.0156</td>
<td>0.0123</td>
<td>0.0012</td>
<td>0.0036</td>
<td>0.0012</td>
</tr>
<tr>
<td></td>
<td>P value</td>
<td>-1.64E-02</td>
<td>6.70E-23</td>
<td>1.37E-18</td>
<td>1.37E-19</td>
<td>7.55E-05</td>
<td>1.56E-03</td>
<td>4.46E-02</td>
<td>2.75E-08</td>
</tr>
<tr>
<td>Dolphins</td>
<td>Max. 0.5277</td>
<td>0.5277</td>
<td>0.4955</td>
<td>0.5210</td>
<td>0.5155</td>
<td>0.5014</td>
<td>0.5259</td>
<td>0.5258</td>
<td>0.5210</td>
</tr>
<tr>
<td></td>
<td>Avg. 0.5222</td>
<td>0.5268</td>
<td>0.4938</td>
<td>0.4629</td>
<td>0.4838</td>
<td>0.4946</td>
<td>0.5210</td>
<td>0.5215</td>
<td>0.5189</td>
</tr>
<tr>
<td></td>
<td>Std. 0.0014</td>
<td>0.0124</td>
<td>0.0356</td>
<td>0.0452</td>
<td>0.0141</td>
<td>0.0252</td>
<td>0.0153</td>
<td>7.83E-01</td>
<td>3.73E-02</td>
</tr>
<tr>
<td></td>
<td>P value</td>
<td>1.72E-23</td>
<td>3.46E-40</td>
<td>2.40E-30</td>
<td>2.38E-08</td>
<td>2.15E-08</td>
<td>4.09E-01</td>
<td>7.83E-01</td>
<td>3.73E-02</td>
</tr>
<tr>
<td>Book</td>
<td>Max. 0.5272</td>
<td>0.5268</td>
<td>0.5181</td>
<td>0.5272</td>
<td>0.5181</td>
<td>0.5230</td>
<td>0.5230</td>
<td>0.5181</td>
<td>0.5268</td>
</tr>
<tr>
<td></td>
<td>Avg. 0.5255</td>
<td>0.5254</td>
<td>0.5178</td>
<td>0.5269</td>
<td>0.5178</td>
<td>0.5230</td>
<td>0.5208</td>
<td>0.5027</td>
<td>0.5236</td>
</tr>
</tbody>
</table>
## Experiment 2

This experiment’s goal was to evaluate the accuracy of the community structures produced by different algorithms. For this purpose we use the GN benchmark network [41] with 128 nodes, which are partitioned to four communities with 32 nodes each. A critical mixing parameter $\lambda$ is used to control the community structure in the network. The mixing parameter $\lambda$ was varied from 0 to 0.5 by increments of 0.05. Since the communities in benchmark networks are already known, we use the Normalized Mutual Information (NMI) to evaluate the performances of different algorithms. Fig. 6 shows the average NMI obtained by MLAMA-Net and other algorithms on GN benchmark networks with different mixing parameter. From the results shown in this figure we report the following:

- The MLAMA-Net algorithm outperforms other algorithms for all values of mixing parameter. The MLAMA-Net decomposes the problem of community detection into several sub problems (in MLAMA-Net algorithm each node of input network and its neighbors form a sub problem). The decomposition strategy has been proven to be highly effective at finding evenly distributed Pareto optimal solutions [43], so that it makes the MLAMA-Net outperforms the other algorithms when the community structure in the network is indistinct.

- For mixing parameter $\lambda<0.3$, all algorithms, except GA-Net and MOGA-Net, can find the community structure corresponding to correct partitioning (NMI $\approx 1$).

### Table 1: Comparison of Community Detection Algorithms on Different Networks

<table>
<thead>
<tr>
<th>Network</th>
<th>Max.</th>
<th>Avg.</th>
<th>Std.</th>
<th>P value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Football</td>
<td>0.6058</td>
<td>0.6042</td>
<td>0.0025</td>
<td>-</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.6050</td>
<td>0.6042</td>
<td>0.0025</td>
<td>-</td>
</tr>
<tr>
<td>Std.</td>
<td>0.0023</td>
<td>0.0025</td>
<td>0.0021</td>
<td>-</td>
</tr>
<tr>
<td>P value</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Power grid</td>
<td>0.9357</td>
<td>0.7350</td>
<td>0.0025</td>
<td>-</td>
</tr>
<tr>
<td>Avg.</td>
<td>0.9336</td>
<td>0.7350</td>
<td>0.0025</td>
<td>-</td>
</tr>
<tr>
<td>Std.</td>
<td>0.0010</td>
<td>0.0025</td>
<td>0.0014</td>
<td>-</td>
</tr>
<tr>
<td>P value</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

* Best results are highlighted in bold
- The NMI of almost algorithms, except MLAMA-Net, begins to decrease for mixing parameter between $[0.3, 0.4]$.
- The NMI of almost algorithms begins to decrease for mixing parameter $\lambda > 0.4$, because the community structure becomes indistinct.

![Fig. 6 - The average NMI obtained by different algorithms on GN benchmark networks](image)

**Experiment 3**

The GN benchmark networks [41] do not reflect some important properties of real world networks, like the power law distribution of node degrees and community sizes. Therefore, we used the LFR benchmark networks [42], which are more consistent with the properties of real world networks, to evaluate the accuracy of the community structures produced by different algorithms. In this experiment the size of networks is set to 1000 and the node degrees and the size of communities are taken from the power law distribution with exponent $\tau_1 = 2$ and $\tau_2 = 1$ respectively. The degree of nodes is in the range from 0 to 50 with average value 20 and the community size is between $[10, 50]$. The mixing parameter $\mu$ was varied from 0 to 0.8 by increments of 0.05. Since the communities in benchmark networks are already known, we use the Normalized Mutual Information (NMI) to evaluate the performances of different algorithms. Fig. 7 shows the average NMI obtained by MLAMA-Net and other algorithms on LFR
benchmark networks with different mixing parameter values. From the results shown in this figure we report the following:

- For mixing parameter $\mu<0.1$, all algorithms, except GA-Net and MOGA-Net, can find the community structure corresponding to correct partitioning ($\text{NMI} \approx 1$).
- The NMI of almost algorithms, except MLAMA-Net, CLA-Net, Meme-Net and MIGA being to decrease for mixing parameter between [0.1, 0.35].
- For mixing parameter between [0.35, 0.5], the NMI values of all algorithms, except MLAMA-Net, decrease.
- The MLAMA-Net algorithm outperforms the other algorithms for mixing parameter in range [0, 0.7]. This is due to the decomposition of the input network into a number of sub networks and using the priority concept in community detection process in the proposed algorithm.
- For mixing parameter $\mu>0.7$, the MOEAD-Net algorithm perform better than proposed algorithm, but still MLAMA-Net outperforms the other algorithms.

![Fig. 7 - The average NMI obtained by different algorithms on LFR benchmark networks](image)

**Experiment 4**

The goal of this experiment was to study the impact of the resolution limit problem on the community structure obtained by the MLAMA-Net algorithm. For this purpose, we use a sample network with 54 nodes. The sample network is partitioned to two communities $C_1$ with 50 nodes
and $C_2$ with 4 nodes. The topologies of communities $C_1$ and $C_2$ are the same as the complete graph, in which every pair of distinct nodes in a community is connected by an edge. Also, node $u$ in community $C_1$ is connected to node $v$ in community $C_2$. Fig. 8 shows the described test network for evaluation of resolution test problem. The modularity of the community structure in the test network is 0.0097. However, according to basic modularity optimization, node $u$ would be assigned to community $C_2$, because this community structure achieves a larger modularity value of 0.0101. This is due to the resolution limit of modularity optimization. We run the proposed MLAMA-Net algorithm on the test network 100 times. Since the other algorithms such as CNM and MIGA solely depend on the modularity optimization, they always incorrectly identify node $u$ as a member of community $C_2$ [39]. The proposed algorithm can always find the correct communities in the test network, because it uses both the modularity and priority concept in community detection process.

![Community Detection](image)

Fig. 8 – The test network with two communities with different size for evaluation of resolution limit problem

**Experiment 5**

This experiment aimed to find the impact of learning automata and priority function on MLAMA-Net performance. For this purpose we compared the proposed algorithm with the
The proposed algorithm in which the learning automaton residing in each meme is replaced by a pure chance automaton (MLAMA-Net-PC) and the proposed algorithm in which the priority function is replaced by same priority for all nodes (MLAMA-Net-SP) for different networks, in terms of modularity and Normalized Mutual Information (NMI) measures. In pure chance automaton the actions are always chosen with equal probabilities. Table 3 presents the results of the MLAMA-Net algorithm, the MALAM-Net-PC algorithm and the MALAM-Net-SP algorithm for different real world networks with respect to the average and maximum values of modularity measure, standard deviation and the p-values of the two-tailed t test. Each reported result was averaged over 50 runs. We performed a parametric test (t test) at the 95% significance level to provide statistical confidence. The t tests were performed after ensuring that the data followed a normal distribution (by using the Kolmogorov–Smirnov test). From the results described in Table 3 we report the following:

- The MLAMA-Net algorithm can find the community structure with largest modularity value on all real word networks.
- The difference between the performance of the MLAMA-Net algorithm and the performance of the MLAMA-Net-PC is statistically significant (p-value<0.05) in all cases.
- The difference between the performance of the MLAMA-Net algorithm and the performance of the MLAMA-Net-SP is statistically significant (p-value<0.05) in all cases.

Table 3 – Maximum modularity (Max.), average modularity (Avg.) and the result of statistical tests for the MLAMA-Net, MLAMA-Net-PC and MLAMA-Net-SP algorithms on real world networks

<table>
<thead>
<tr>
<th>Network</th>
<th>MLAMA-Net</th>
<th>MLAMA-Net-PC</th>
<th>MLAMA-Net-SP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Karate</td>
<td>0.4198</td>
<td>0.4174</td>
<td>0.4030</td>
</tr>
<tr>
<td></td>
<td>Max. 0.0102</td>
<td>0.0948</td>
<td>0.1052</td>
</tr>
<tr>
<td></td>
<td>P value    -</td>
<td>2.99E-28</td>
<td>1.46E-30</td>
</tr>
<tr>
<td>Dolphins</td>
<td>0.5277</td>
<td>0.5149</td>
<td>0.5201</td>
</tr>
<tr>
<td></td>
<td>Avg. 0.0032</td>
<td>0.1263</td>
<td>0.1376</td>
</tr>
<tr>
<td></td>
<td>P value    -</td>
<td>2.99E-27</td>
<td>2.49E-28</td>
</tr>
<tr>
<td>Book</td>
<td>0.5272</td>
<td>0.4815</td>
<td>0.4372</td>
</tr>
<tr>
<td></td>
<td>Max. 0.0046</td>
<td>0.0730</td>
<td>0.0746</td>
</tr>
<tr>
<td></td>
<td>P value    -</td>
<td>2.28E-57</td>
<td>8.01E-58</td>
</tr>
<tr>
<td>Football</td>
<td>0.6058</td>
<td>0.6003</td>
<td>0.5963</td>
</tr>
<tr>
<td></td>
<td>Avg.</td>
<td>Std.</td>
<td>P value</td>
</tr>
<tr>
<td>-------------------</td>
<td>--------</td>
<td>--------</td>
<td>----------</td>
</tr>
<tr>
<td><strong>Net science</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max.</td>
<td>0.9550</td>
<td>0.2039</td>
<td></td>
</tr>
<tr>
<td>Avg.</td>
<td>0.9544</td>
<td>0.1787</td>
<td></td>
</tr>
<tr>
<td>Std.</td>
<td>0.0004</td>
<td>0.0105</td>
<td>9.99E-21</td>
</tr>
<tr>
<td>P value</td>
<td>-</td>
<td>5.16E-187</td>
<td></td>
</tr>
<tr>
<td><strong>Power grid</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max.</td>
<td>0.9357</td>
<td>0.6660</td>
<td></td>
</tr>
<tr>
<td>Avg.</td>
<td>0.9336</td>
<td>0.3235</td>
<td></td>
</tr>
<tr>
<td>Std.</td>
<td>0.0010</td>
<td>0.1235</td>
<td>1.45E-71</td>
</tr>
<tr>
<td>P value</td>
<td>-</td>
<td>4.43E-44</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 9 shows the average NMI obtained by MLAMA-Net, MLAMA-Net-PC and MLAMA-Net-SP algorithms on GN benchmark networks with different mixing parameter. From the results shown in this figure we report the following:

- The MLAMA-Net algorithm outperforms the MLAMA-Net-PC and MLAMA-Net-SP algorithms for all values of mixing parameter.

![Fig. 9 - The average NMI obtained by MLAMA-Net, MLAMA-Net-PC and MLAMA-Net-SP algorithms on GN benchmark networks](image)

Fig. 10 shows the average NMI obtained by MLAMA-Net, MLAMA-Net-PC and MLAMA-Net-SP algorithms on LFR benchmark networks with different mixing parameter. From the results shown in this figure we report the following:

- The MLAMA-Net algorithm outperforms the MLAMA-Net-PC and MLAMA-Net-SP algorithms for all values of mixing parameter.
Conclusion

A new michigan memetic algorithm called MLAMA-Net is proposed in this paper for solving the community detection in complex networks. The proposed algorithm is an evolutionary algorithm in which each chromosome represents a part of the solution and the whole population represents the solution. In the proposed algorithm each node of input network is equipped with a chromosome and a learning automaton. The chromosome represents the community of corresponding node and saves the histories of exploration. The learning automaton represents a meme and saves the histories of the exploitation. The proposed algorithm is a distributed algorithm in which each chromosome, without any prior information, locally evolves by evolutionary operators and improves by a local search. MLAMA-Net algorithm introduces the priority concept to solve the resolution limit of modularity optimization in community detection problem. To show the superiority of our proposed algorithm over the some well-known algorithms, several computer experiments have been conducted. The obtained results show MLAMA-Net is effective and efficient at detecting community structure in complex networks. MLAMA-Net is compared with other well-known algorithms for the community detection problem on both synthetic and real world networks. Our experimental results showed the superiority of the proposed algorithm in terms of quality of solution and accuracy of solution in both real world networks and synthetic networks.
References


