Computer & Robotics

A New Multi-Agent Bat Approach for Detecting Community Structure in Social Networks

Saeed Alidoost, Behrooz Masoumi *

Faculty of Computer and Information Technology Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran

Received 14 July 2019; Revised 11 August 2019; Accepted 06 October 2019; Available online 05 November 2019

Abstract

The complex networks are widely used to demonstrate effective systems in the fields of biology and sociology. One of the most significant kinds of complex networks is social networks. With the growing use of such networks in our daily habits, the discovery of the hidden social structures in these networks is extremely valuable because of the perception and exploitation of their secret knowledge. The community structure is a great topological property of social networks, and the process to detect this structure is a challenging problem. In this paper, a new approach is proposed to detect non-overlapping community structure. The approach is based on multi-agents and the bat algorithm. The objective is to optimize the amount of modularity, which is one of the primary criteria for determining the quality of the detected communities. The results of the experiments show the proposed approach performs better than existing methods in terms of modularity.

Keywords: *Social networks, Multi-agent systems, Swarm intelligence, Bat algorithm, Community detection, Modularity.*

1. Introduction

In recent years, the use of social networks that is a subset of complex systems has become part of both our everyday lives and the researchers' fascinating topic. A social network is generally modelled as a graph. Both nodes and edges are two significant components of this graph. Each node implies a member of this network, and each edge indicates a relationship between each pair of nodes [1, 2, 3]. One of the important issues of research on social networks has been the community structure detection problem, which has been of great interest to researchers for more than a decade. The key goal of the community detection problem is to figure out the hidden communities of the network. An association of members of the network called the community if that has more internal connections than external links with the rest of the network [4, 5]. The reason for this research field is to achieve a proper community structure, which causes essential information about the relationships between the function and the network topology, which will be discovered [6].

The detecting of community structure involves some complexities. One of which is structural complexity, which in fact refers to the structural characteristics of the network such as whether or not the edges have any weight, whether the edges are directed or not, and the assumption of being or not both constant or variable number of nodes and edges. Another serious matter in this regard is the assumption of the dynamic of nodes, which itself will cause many challenges in the community detection process due to the need to consider new features to identify similar nodes. Furthermore, designing a reusable algorithm for different

* Corresponding author. Email: masoumi@qiau.ac.ir

applications, especially large-scale networks, and improving the accuracy at the detecting process of community structure are other challengers. Apart from all of these challenges, community detection is an NP-hard problem, for which traditional methods of optimization are not capable of effectively solving them [7, 8].

To solve these challenges and achieve a suitable method that is able to determine the community structure with high convergence, this paper has tried to use a combination of both the multi-agent approach and swarm intelligence algorithm since these two approaches are able to complete each other and gain better results in such issues. In this proposed method, some agents with three operators are designed to have behaviour and performance based on the bat algorithm. These operators are called the determining degree of node belonging operator, the adaptive mutation operator, and the self-learning operator. The motivation to use these two popular approaches is to achieve the benefits of the bat algorithm and mitigate its disadvantages [9, 10]. One of the main disadvantages of the bat algorithm is the extremely fast convergence in the initial steps, which leads to a reduction of the global convergence rate of the algorithm [11]. Therefore, due to the above reason and the particular importance in achieving the ideal convergence value, this paper is implemented the bat algorithm using a multi-agent approach to reduce trapping in the local optima and extend the broader range of candidate solutions with defining some operators to get this target [10, 12].

To evaluate the effectiveness of the proposed method, the papers in [13, 14] used the popular real-world networks such as the Karate Network, the Dolphin Network, the Political Books Network, and the Football Network. Moreover, the paper compared the proposed method with several other famous methods in this field of studies, such as the Genetic Method, the Memetic Method, and the Multi-agent Genetic Method. The results of these experiments demonstrate that the proposed method has improved the quality of the community structure. To determine the quality, the paper utilized the two really reputable and reliable criteria, the modularity and the normalized mutual information.

The rest of this paper is organized as follows: Section 2 surveys the previous related works; Section 3 introduces the bat algorithm and briefly describes it; Section 4 describes the proposed method completely; Section 5 represents the results of the execution of our algorithm and comparisons with other algorithms. Finally, section 6 expresses the conclusions and future work.

2. Related Works

The main goal of the community detection problem is to discover the existing hidden phenomenon in the networks. Many evolutionary algorithms have been proposed to solve problems in this research topic. Recently, multi-agent systems have been integrated with various evolutionary algorithms to solve constraint satisfaction and combinatorial optimization problems and have achieved good and effective results [15, 16, 9, 17]. Some of the most important issues will be reviewed in the following.

In [15], the research presented a genetic algorithm and the quality of a detected association has been assessed by a community score and then optimize it. In this algorithm is used the locus-based adjacency representation. The main advantage of this representation is that the number of communities in the decoding process is determined automatically. In addition, they have modelled a specific mutation operator. These operators reduce the search space by generating reliable solutions and this condition causes that the convergence of the algorithm is improved.

In [18], a framework for detecting community structure has been proposed based on the use of a multi-agent approach and swarm intelligence. The research used the locus-based adjacency representation and utilized the genetic algorithm to define the behaviour of the agent operators. This study considered four operators for their agents and used the modularity criteria to perceive the quality of their selected communities, and used the normalized mutual information criteria to measure the accuracy of their proposed method. The results of experiments reveal that the method improved the modularity compared with other conventional methods.

In [19], the researchers compare some significant metaheuristic algorithms. These algorithms include the original Bat Algorithm (BA), Gravitational Search Algorithm (GSA), modified Big Bang–Big Crunch algorithm (BB-BC), improved Bat Algorithm based on the Differential Evolutionary algorithm (BADE), effective Hyper heuristic Differential Search Algorithm (HDSA) and

Scatter Search algorithm based on the Genetic Algorithm (SSGA). They used the locus-based adjacency representation and the modularity criterion to assess the quality of their proposed method. They proved that the HDSA algorithm in their study is more efficient and competitive than the other algorithms that were tested.

In [16], a multi-objective genetic algorithm (MOGA-Net) is used to detect community structure. This algorithm optimizes two objective functions called community scores and community fitness simultaneously. The number of edges within the communities is maximized and the number of edges in the communities is minimized by maximizing the community score and community fitness. This algorithm is also used to describe the locus-based adjacency representation. The main advantage of multi-objective approaches is that they do not produce just one solution, but they produce a set of solutions. Researchers in [20] proposed a new method using a multi-agent approach. They defined two types of agents for their proposed community detection process; one has considered at the node level and other at the community level. In [21], the researchers presented a novel method using the swarm intelligence approach and the utilization of the bat algorithm. They used the locus-based adjacency representation and the modularity criterion to assess the quality of their proposed method.

3. Bat Algorithm

Yang developed the bat algorithm (BA) based on the echolocation features of micro bats in 2010 [22]. In this algorithm, Bats can hunt their prey by sound emission and reception. When the bat searches its prey, the loudness is very high and its wavelength is low, while after finding the prey, the loudness suddenly decreases, and the wavelength increases. Thus, they usually use the loudness to measure convergence in different purposes [23, 24].

The loudness can vary from a large (positive) value A0 to a constant value Amin. The pulse rate (δ) is in the range [0,1], where 0 means no pulses in all, and 1 means the maximum pulse rate [22, 23, 24, 12, 11]. The bat algorithm has several substantial advantages including automatic control and fast movement with the use of the loudness and pulse rates, discovering the best solution in quick time, simplicity, flexibility, and easy implementation [22, 23, 24]. The benefits of this algorithm have made this research a great deal of interest. The bat algorithm also has disadvantages that compel researchers to combine it with other algorithms and approaches for earning better results. This has led to developing the new innovative methods of bat algorithm such as the fuzzy bat algorithm, the multi-objective bat algorithm, the K-Mean bat algorithm, and the binary bat algorithm. The main fault of this algorithm is the reduction of the global convergence rate of the algorithm due to its very rapid convergence in the initial steps [25]. Algorithm 1 provides a better understanding of the bat algorithm process.

Algorithm 1: Bat algorithm

```
Begin
 Input the parameters of the algorithm and initial data
 Generate M initial possible situations
 While (the termination criteria are not satisfied)
    Evaluate fitness value for all solutions
    Rank all solutions according to their fitness values and find
    the current best solution
     For j = 1 to M
          Generate new solutions by adjusting frequency, and
                updating velocities and locations/ solutions
          Generate Rand randomly
         If Rand > δj
               Select a solution among the best solutions
               Generate a local solution around the best
               solution
         End if
         Generate a new solution by random fly
         Generate Rand randomly
         If (Rand < Aj) and (the new solution is better than
         the old one)
               Accept the new solutions
               Increase δj and reduce Aj
         End if
       Next j
 End while
 Report all solutions
End
```

4. The Proposed Approach: Multi-Agent Bat Approach

This section explains the framework of the proposed approach that reaches the goal of finding the best part of the graph.

4.1. Modularity Definition

Newman and Girvan in [14] defined modularity Q as a criterion to evaluate the quality of a division of a network. It was broadly used to signify the quality of the community structure. It is a quality evaluation that reveals the difference between both the detected communities and a random graph. The introduced method in [18] takes modularity optimization as its fitness function and assesses the detected community in the population to enhance the modularity values as much as possible. The modularity Q can be defined as follows:

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

Where s is the entire communities, L is the sum of all edges in the network, l_k is the number of edges inside community k and d_k is the summation of degrees of the whole of nodes inside community k.

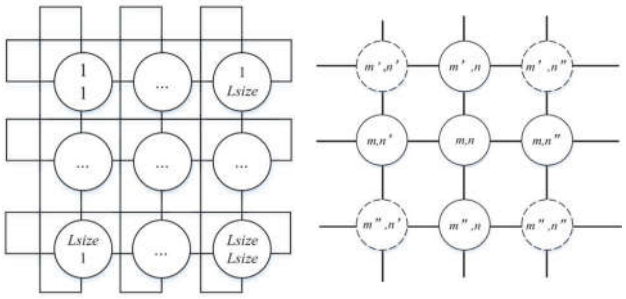
4.2. Community Detection by Agents

Agents can understand and react to the environment [18, 26]. In the proposed approach, an agent is defined as a division of the network. Every agent is a candidate solution for the community detection problem, and the energy value of each agent is equal with the modularity criterion that referred to in Eq. (1). The agents live in a lattice-like environment L called agent lattice. Each agent is fixed at one point in the lattice and can only exchange information with its neighbours. The number of factors is Lsize × Lsize. The network of agents is defined as Fig. 1 (A). Assume that an agent located on m, n, $m, n = 1,2, \dots, L_{size}$, then the neighbors of this agent can be defined as $neighbors_{m,n}$ in Eq. (2) and Eq. (3). We can also observe neighbours $L_{m,n}$ in section (B) of Fig. 1

$$neighbors_{m,n}\{L_{m',n}, L_{m,n'}, L_{m,n"}, L_{m",n}\} \quad (2)$$

$$m' = \begin{cases} m-1 & m \neq 1 \\ L_{size} & m = 1 \end{cases} \quad n' = \begin{cases} n-1 & n \neq 1 \\ L_{size} & n = 1 \end{cases} \quad (3)$$

$$m" = \begin{cases} m+1 & m \neq L_{size} \\ 1 & m = L_{size} \end{cases} \quad n" = \begin{cases} n+1 & n \neq L_{size} \\ 1 & n = L_{size} \end{cases}$$



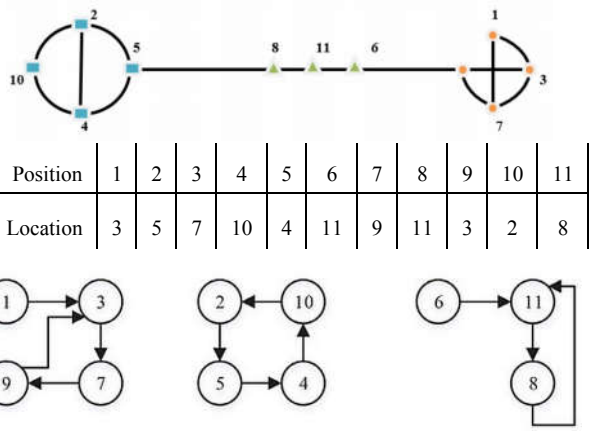
A) Model of agent lattice B) The neighbours of an agent

Fig. 1. The presentation of the agent lattice and its neighbours.

4.3. Representation and Initialization of Agents

In the proposed approach, the locus-based adjacency representation introduced in [27] to illustrate agents. The reason for using the locus-based adjacency representation is that the method restricts the possible solution space and reduces invalid search and also prevents the insertion of individual nodes in a partition.

After creating the initial list of nodes (positions), a position is randomly selected as the starting point for discovering the neighboring nodes. Next, one of the neighboring nodes of the desired position is randomly assigned as the location if exists an edge between them. Continuing this process, the value selected as the location is assumed to be a new position and one of its neighbors will be selected again as the location value. This process will continue until the entire list is reviewed and all the nodes are met. After this step, the initial components will be obtained and will include at least one clique.



Position	1	2	3	4	5	6	7	8	9	10	11
Location	3	5	7	10	4	11	9	11	3	2	8

Fig. 2. The method of encoding the network based on the locus-based adjacency representation [19].

The initialization procedure is quite simple. This action is fast and fully utilizes the communication link of all nodes. As a result, this kind of representation can get fairly good feedback at the beginning of the work, but it should be noted that this achievement is still far from the optimal status. Fig. 2 provides the process for better understanding.

4.4. Bat Operators of Agents

In this section, operators of the proposed algorithm will be introduced. In the proposed approach, three operators are designed based on the bat algorithm; these operators are described in the following order: determining the degree of node belonging operator, adaptive mutation operator, and self-learning operator.

4.4.1. Determining the Degree of Node Belonging Operator

Initially, network components are constructed using the locus-based adjacency representation. Then, one of the components is randomly considered as a community, and on the rest of the network, the bats look for the nodes with the best degree of belonging to the community. A random number with a normal distribution is assigned to each node, and the degree of belonging is also evaluated by each bat based on Eq. (4), and the average of this random number is considered in each step; in other words, the loudness value of the bat is equal to the degree of belonging of each node.

$$Belonging(n, C) = \frac{nbNeighb(n, C)}{nbBonds(n)} \quad (4)$$

In the above equation, the numerator represents the number of edges from node n toward community C, and the denominator is all edges of node n throughout the network.

Therefore, using these two values loudness value and normal random distribution, the final nodes that make up the new candidate solutions are determined. The process of doing so is that if the random number is larger than the average loudness value of bat, one of the neighbours of the node is chosen randomly; otherwise, the node in the new list of nodes belonging to the community is preserved. Fig. 3 illustrates this trend. The described procedure continues until the whole of the network nodes are evaluated or the loop counter reaches a determined iteration. Then, the desired nodes are considered as part of the initial community, respectively, and will create a new temporary community C, in such a way that the community C has at least one clique; It will also be examined in each step that if there are common nodes of communities with the probability of s_{sp}, they are separated from each other, one of which has a loop to have a better structure, and another without a loop to achieve opportunity again of gain a better division of the network. In the next step, the computing of the modularity criterion on this community is calculated by the agent; this process continues until the modularity criterion is increased by adding each node to the temporary community; otherwise, if the modularity value is decreased then the added node to the temporary community C is removed. Then, the process will continue on other nodes in the belonging node list.

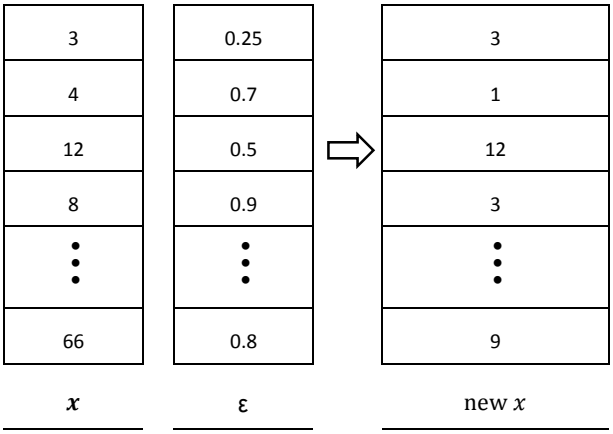


Fig. 3. How to generate a new solution by the bat using the old solution x and the random number ε when the average of loudness value is $A^t = 0.6$.

Finally, agents that do not have any cliques will die, and then the steps will be performed from the beginning on the nodes belonging to the dead agent. This causes the integration of communities to take place at a lower cost. Moreover, it should be noted that agents with the best value of the modularity criterion at each step are stored at a smaller network for conducting the learning operator.

4.4.2. Adaptive Mutation Operator

In order to prevent invalid searches in the search space and to preserve the diversity of the population with the aim of achieving the best energy for the operators (best modularity), this operator has been implemented using the proposed method introduced in [28]. Gaussian distribution has been used to realize the production of a new solution.

Therefore, the new position of the bats is calculated in Eq. (5). But, this method can easily determine the amount of standard deviation to generate large or small numbers to the required value, which is one of the parameters of this type of distribution of numbers.

$$X_{new} = X_{old} + \varepsilon \overline{A^t}\, \mathrm{N}(0, \sigma) \quad (5)$$

In the above equation, $\mathrm{N}(0, \sigma)$ represents a normal/Gaussian distribution, X_{old} represents the current population of the algorithm, ε represents a random step, $\overline{A^t}$ and represents the average of loudness value of bat.

In [28] is described as the rules for the adaptive mutation in random walk size and pulse rates to control the exploration of new situations and extraction in local optimizations.

4.4.3. Self-Learning Operator

This operator is able to use the knowledge of the self-agent to achieve the best community that leads to more modularity value. In this section, a small network of best agents in the size $sL_{size} \times sL_{size}$ is formed according to Eq. (6).

$$sL = \begin{cases} L_{m,n} & m' = 1, \quad n' = 1 \\ sL_{m',n'} & otherwise \end{cases} \quad (6)$$

At the above equation, $L_{m,n}$ is the agent that conducts the self-learning operator, $sL_{m',n'}$ is generated by executing the adaptive mutation operator in neighbours of $L_{m,n}$ with the probability of s_{pm}.

The self-learning operator is done on the agents that have had the best result so far (the best agent has the most modularity value). Using the two previous operators, the operator can create a diverse set of nodes. This process increases the perception of the agent from its environment and approaches convergence.

Finally, the best agent with the maximum energy will be exchanged with the agent that executes the self-learning operator. Each agent for conducting the self-learning operator has a binary logic variable, which, if this value is true, can execute this function. After each run, the value of the learning label of the agent is equal to False; this condition causes to prevent reusing the function by the agent. The pseudo-code presented in Algorithm 2 shows the algorithm of the self-learning operator to better understand the process.

Algorithm 2: Self-learning Operator

```
  Input:
      sLt : communities at the tth iteration of sL;
      sNs: the maximum number of iteration without improvement;
      Lm,n: an community in L to conduct the self-learning operator;
      sBest(t) : the best community in sL0, sL1, ....., sLt ;
      sCBest(t) : the best community in sLt ;
  Output:
      Lm,n ← sBest(t);
      Learning(Lm,n)←False;
t ← 0;
n ← 0;
sL0 ←initialize sL using the adaptive mutation operator and update
sBest0;
while (n < sNs) do
    t ← t + 1;
    sLt ←conduct the determining degree of node belonging operator
    on sLt ;
    Update sCBest(t);
    if (Q(sCBest(t)) > Q(sBest(t)−1)) then
        n ← 0;
        sBestt ← sCBest(t);
    else
        n ← n + 1;
        sBest(t) ← sBest(t)−1;
        sCBest(t) ← sBest(t);
    end
end
```

4.5. Implementation of the Multi-Agent Bat Algorithm

After that explaining the desired operators, the proposed framework is visible in Algorithm 2 and Fig. 4 to better understand this process.

As it is clear in this algorithm, the initial population is originally formed using the locus-based adjacency representation, and the learning label of the agent is set to True. The process can continue within the loop until the end of its maximum iteration. At the beginning of the loop, the determining degree of node belonging operator uses to generate the communities with the best of the modularity value. Then, the adaptive mutation operator is implemented to simultaneously meet two important effects of exploration in the search space and to preserve the diversity of the population. Subsequently, the self-learning operator on sl is executed on the best of agents in the L network, which plays a significant and irreplaceable role in improving the performance of the proposed algorithm.

Algorithm 3: Multi-agent bat algorithm

```
  Input:
      Lt : communities at the t-th generation of L;
      sl: the number of communities carried out self-learning
          operator;
      Best(t) : the best community in L0, L1, ....., Lt ;
      CBest(t) [sl]: the best sl communities in Lt ;
      CBest(t) : the best community in Lt ;
      Ns: the maximum number of generations without improvement;
  Output:
      Transform the optimal agent in Lt into a partition solution and
      output;
t ← 0;
n ← 0;
L0 ← initialize the population by locus based adjacency representation,
assign the Learning labels of L0 as True and Update Best0;
while (n < Ns) do
    t ← t + 1;
    Lt ←conduct the determining degree of node belonging operator
    on Lt and update Learning labels of Lt;
    Lt ←conduct the adaptive mutation operator on Lt and
    update Learning labels of Lt ;
   CBest(t) [sl] ←Finding the best sl agents in Lt ;
   for i ← 1 to sl do
       if Learning(CBest(t) [i]==True) then
           Conduct self-learning-operator on CBest(t) [i]
       end
   end
   Update CBest(t);
   if (Q(CBest(t)) > Q(Bestt−1)) then
       n ← 0;
       Best(t) ← CBest(t);
   else
       n ← n + 1;
       Best(t) ← Best(t)−1;
       CBest(t) ← Best(t);
   end
end
```

5. Experimental Results

In this section, the results of performed experiments will be indicated for proving the superiority of the proposed approach compared with other available methods, such as the Genetic method, the Memetic method, and the Multi-agent genetic method. To do this, we use real-world datasets, for instance, Karate Club, Dolphin Network, Political Books, and Football. These results were achieved with the two criteria of modularity and normalized mutual information. All experiments were conducted on a machine with a 3.2 GHz CPU and 4GB of memory. The Microsoft Visual Studio 2017 was also used to perform the implementation, and the outputs of this execution are reported as test results in this section. Moreover, the processing time is neglected in the proposed approach and only the performance improvement is investigated.

In this paper, four datasets with real information are used to examine the proposed approach. Executing the proposed approach is repeated in accordance with Table 1 for each dataset and the results are presented as the output of this method. The parameters used in the proposed approach are shown in Table 2. several parameters are chosen from [18] because of creating equality between the proposed approach and the multi-agent genetic algorithm.

In Table 2, Lsize is the number of agents in the network of agents, sLsize is the size of the smaller network of the agents for conducting the self-learning operator, sl is the smaller network of the best agents for conducting the self-learning operator, Ns represents the maximum number of iteration without improvement, sNs implies the maximum number of iteration without improvement for the self-learning operator, s_sp indicates the separation probability of detected communities, A is the loudness value of the bats in the initial step, r is the pulse value of the bats in the initial step, α and γ are respectively for calculating the loudness value and pulse rate in the next steps of the bat algorithm.

Table 1. The number of evaluations used by the proposed approach, multi-agent genetic algorithm, HDSA, Memetic algorithm, and genetic algorithm

Graph	Karate club	Dolphins	Political books	Football
Evaluation	3200	5000	5500	7500

Table 2. Parameter determination

L_{size}	sL_{size}	sl	N_s	sN_s	s_{sp}	A	r	α	γ
5	3	10	50	50	0.4	0.9	0.01	0.99	0.02

As mentioned earlier, both the modularity and normalized mutual information have been used to evaluate the accuracy of the proposed approach and compare it with other common methods. In the following, at first, the results of the modularity criterion are demonstrated in the form of a table and a chart, and then we will define the normalized mutual information criterion and present its results in the form of a table and a chart.

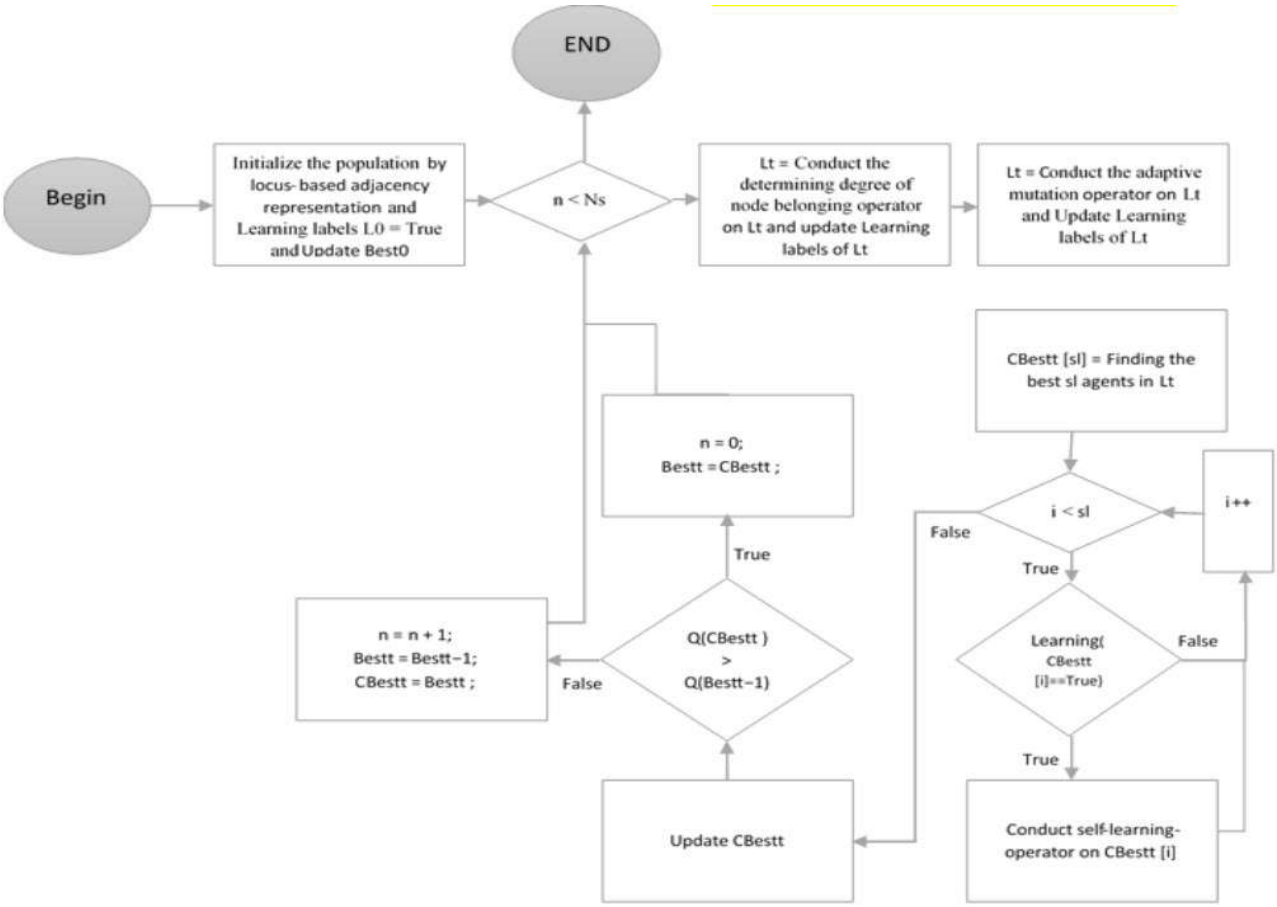


Fig. 4. The flowchart of the process in the Multi-agent bat algorithm.

Table 3 and Fig. 5 illustrate the obtained values for the modularity criterion in the form of both tables and charts. In these results, QMin represents the lowest result, and QMax represents the best result among the various repetitions; the QAvg represents the average of all performances based on the proposed approach with three previously introduced methods. As presented in the diagrams, the proposed approach makes improvements compared with the previous methods and obtained better results on all databases.

Table 3. The comparisons in terms of modularity Q (QMax, QAvg , QMin) on the four real-world networks based on the proposed approach, Multi-agent Genetic algorithm, Memetic algorithm, and Genetic algorithm [18], HDSA [19]

Dataset	Genetic algorithm			Memetic algorithm			HDSA			Multi-agent algorithm		genetic	Proposed approach		
	Q_{Max}	Q_{Avg}	Q_{min}	Q_{Max}	Q_{Avg}	Q_{min}	Q_{Max}	Q_{Avg}	Q_{min}	Q_{Max}	Q_{Avg}	Q_{min}	Q_{Max}	Q_{Avg}	Q_{min}
Karate	0.4198	0.3741	0.0767	0.4198	0.4083	0.0134	0.4198	0.4198	0.4198	0.4198	0.4194	0.0019	0.4259	0.4203	0.0920
Dolphins	0.5227	0.4928	0.0119	0.5028	0.4273	0.3050	0.5285	0.5282	0.5276	0.5286	0.5271	0.0007	0.5337	0.5318	0.1022
Polbooks	0.5212	0.4871	0.0369	0.5139	0.4436	0.0216	0.5272	0.5272	0.5272	0.5273	0.5270	0.0001	0.5376	0.5297	0.0371
Football	0.5661	0.5020	0.0237	0.5492	0.4904	0.0233	0.6046	0.6033	0.6019	0.6046	0.6020	0.0026	0.6218	0.5989	0.0246

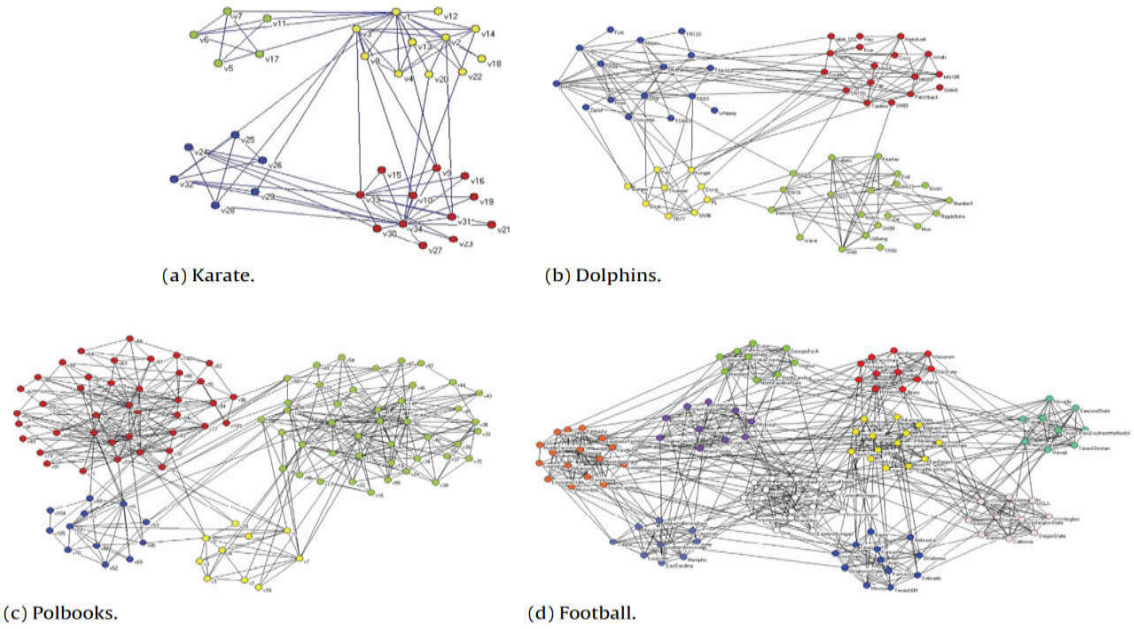


Fig. 5. The detected community structure is based on the maximum of the modularity criterion.

Normalized mutual information (NMI) is used to assess the outcomes [29, 30, 18]. NMI is a similarity measure estimating the similarity between the detected cuts and the true ones. Assume A and B are cuts of a network, and c_A expresses the number of communities in A while c_B denotes that of B. D is a confusion matrix, and D_{ij} stands for the number of nodes in the community i of A that also appear in community j of B. N is the number of elements. D_i is the sum over row i of D while D_j is the sum of elements in column j. The definition of $\mathrm{NMI}(\mathrm{A}, \mathrm{B})$ is shown at Eq. (7).

$$NMI(A,B) = \frac{-2\sum_{i=1}^{c_A}\sum_{j=1}^{c_B} D_{ij}\log\left(\frac{D_{ij}N}{D_i N_j}\right)}{\sum_{i=1}^{c_A} D_i \log\left(\frac{D_i}{N}\right) + \sum_{j=1}^{c_B} D_j \log\left(\frac{D_j}{N}\right)} \quad (7)$$

As shown in Table 3, the multi-agent genetic algorithm has a better outcome based on Q_{Max} in terms of the modularity criterion. Therefore, the results of calculating the normalized mutual information value for comparison of the multi-agent genetic algorithm and the proposed approach are presented in Table 4.

Table 4. The comparisons based on the NMI criterion on the four real-world networks.

Dataset	Multi-agent genetic algorithm		Proposed Approach	
	Max	Avg	Max	Avg
Karate	0.8432	0.8395	0.8751	0.8512
Dolphins	0.7912	0.7815	0.8233	0.8007
Polbooks	0.8250	0.8104	0.8496	0.8311
Football	0.9480	0.9027	0.9499	0.9168

6. Conclusion

In this paper, a new approach was proposed based on both the multi-agent and swarm intelligence with the design of three operators, namely, the determining degree of node belonging operator, the adaptive mutation operator, and the self-learning operator. Each of the operators had the behaviours of the bat algorithm for optimizing the amount of modularity. The proposed approach was executed on Karate Clubs, Dolphin Networks, Political Books, and Football. The achievement results based on both the modularity and the normalized mutual information criteria indicate that this method can achieve better results than other conventional

methods. As an idea for future work, developing an algorithm with the same approach can be considered for node mobility in online social networks because of in real-world networks of today, social networks are mostly mobility, and nodes are constantly changing their position; this event will cause the network topology to change frequently. Changing nodes has been a major challenge in the community detection process, and it has been necessary to define some features to identify the similarity between the nodes and belong them to the community, which would accept and converge this structure.

References

[1] Latora, V.; Nicosia, V.; Russo, G., Complex networks: principles, methods and applications (2017).

[2] Chen, G.; Wang, X.; Li, X., Fundamentals of complex networks: models, structures and dynamics, John Wiley & Sons (2014).

[3] Reihanian, A.; Feizi-Derakhshi, M.; Aghdasi, H., "Community detection in social networks with node attributes based on multi-objective biogeography based optimization", Engineering Applications of Artificial Intelligence, vol. 62, pp. 51-67 (2017).

[4] Silva, T. C.; Zhao, L., "Machine learning in complex networks", in Springer (2016).

[5] Ghaderi, S.; Abdollahpouri, A.; Moradi, P., "On the modularity improvement for community detection in overlapping social networks", in IEEE - 2016 8th International Symposium on Telecommunications (IST) (2016).

[6] Estrada, E., "Introduction to complex networks: structure and dynamics", in Evolutionary Equations with Applications in Natural Sciences, no. Springer, pp. 93-131 (2015).

[7] Huang, J.; Yang, B.; Jin, D.; Yang, Y., "Decentralized mining social network communities with agents", Mathematical and Computer Modelling - Elsevier, vol. 57, no. 11-12, pp. 2998-3008 (2013).

[8] Girvan, M.; Newman, M. E., "Community structure in social and biological networks", Proceedings of the national academy of sciences, vol. 99, no. 12, pp. 7821-7826 (2002).

[9] Cazabet, R.; Amblard, F., "Simulate to Detect: A Multi-agent System for Community Detection", in IEEE, Lyon, France (2011).

[10] Christian Blum, D. M., Swarm Intelligence: Introduction and Applications (Natural Computing Series), Springer (2008).

[11] Bozorg-Haddad, O.; Karimirad, I.; Seifollahi-Aghmiuni, S.; Loáiciga, H. A., "Development and application of the bat algorithm for optimizing the operation of reservoir systems", Journal of Water Resources Planning and Management (2014).

[12] Colin, T., "A comparison of BA, GA, PSO, BP, and LM for training feed forward neural networks in e-learning context", International Journal of Intelligent Systems and Applications, pp. 23-29 (2012).

[13] Lancichinetti, A.; Fortunato, S.; Radicchi, F., "Benchmark graphs for testing community detection algorithms", Physical review E, vol. 78, no. 4, 046110 (2008).

[14] Newman, M. E.; Girvan, M., "Finding and evaluating community structure in networks", Physical review E, vol. 69, no. 2, 026113 (2004).

[15] Pizzuti, C., "GA-Net: A genetic algorithm for community detection in social networks", in International Conference on Parallel Problem Solving from Nature (2008).

[16] Pizzuti, C., "A multi-objective genetic algorithm for community detection in networks", in Tools with Artificial Intelligence 21st International Conference (2009).

[17] Li, Y.; Liu, J.; Liu, C., "A comparative analysis of evolutionary and memetic algorithms for community detection from signed social networks", vol. 18, no. 2, pp. 329-348 (2014).

[18] Li, Z.; Liu, J., "A multi-agent genetic algorithm for community detection in complex networks", Physica A: Statistical Mechanics and its Applications, vol. 449, pp. 336-347 (2016).

[19] Atay, Y.; Koc, I.; Babaoglu, I.; Kodaz, H., "Community Detection from Biological and Social Networks: A Comparative Analysis of Metaheuristic Algorithms", Applied Soft Computing (2016).

[20] Cazabet, R.; Amblard, F., "Simulate to detect: a multi-agent system forc ommunity detection", in IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, Lyon, France (2011).

[21] Hassan, E. A.; Hafez, A. I.; Hassanien, A. E.; Fahmy, A. A., "A Discrete Bat Algorithm for the Community Detection Problem", in International Conference on Hybrid Artificial Intelligence Systems. HAIS 2015. Lecture Notes in Computer Science, vol. 9121. Springer, Cham (2015).

[22] Yang, X. S. , "A new meta-heuristic bat-inspired algorithm," Nature inspired cooperative strategies for optimization (NISCO 2010), vol. 284, pp. 65-74, 2010.

[23] Yang, X. S., "Meta-heuristic optimization with applications: Demonstration via bat algorithm", in Proceedings of 5th Bioinspired Optimization Methods and Their Applications (BIOMA2012), Bohinj, Slovenia (2012).

[24] Yang, X. S.; Gandomi, A. H., "Bat algorithm: A novel approach for global engineering optimization", in Engineering Computations (2012).

[25] Koffka, K.; Ashok, S., "A comparison of BA, GA, PSO, BP, and LM for training feed forward neural networks in e-learning context", International Journal of Intelligent Systems and Applications, vol. 7, p. 23–29 (2012).

[26] Liu, J.; Jing, H.; Tang, Y.Y., "Multi-agent oriented constraint satisfaction", Artificial Intelligence, vol. 136, no. 1, pp. 101-144 (2002).

[27] Park, Y.; Song, M., "A genetic algorithm for clustering problems", in Proceedings of the Third Annual Conference on Genetic Programming, pp. 568–575 (1998).

[28] Wasi Ul Kabir, Md.; Sakib, N.; Mustafizur, S.; Shafiul, M., "A Novel Adaptive Bat Algorithm to Control Explorations and Exploitations for Continuous Optimization Problems", International Journal of Computer Applications (0975 – 8887), vol. 94, no. 13, pp. 15-20 (2014).

[29] Labatut, V., "Generalised measures for the evaluation of community detection methods", International Journal of Social Network Mining, vol. 2, no. 1, pp. 44-63 (2015).

[30] Danon, L.; Diaz-Guilera, A.; Duch, J.; Arenas, A., "Comparing community structure identification", Journal of Statistical Mechanics: Theory and Experiment, no. 9, pp. 219-228 (2005).