Journal of Electrical and Computer Engineering Innovations
(JECEI)
Journal homepage: http://www.jecei.sru.ac.ir

Research paper

BSOGA: Community Detection in Social Networks Based on Bee Swarm Optimization Using Genetic Algorithm

Fatemeh Akbari , ***Eynollah Khanjari*** *

Department of Computer Engineering, Tehran Branch, Iran University of Science and Technology (IUST), Tehran, Iran.

Article Info

Article History:
Received 17 July 2025
Reviewed 05 September 2025
Revised 02 November 2025
Accepted 09 November 2025

Keywords:
Community detection
Social networks
Genetic Algorithm
Bee colony algorithm
Locus-based adjacency
Modularity

*Corresponding Author's Email Address: *Khanjari@iust.ac.ir*

Abstract

Background and Objectives: So far, several methods have been proposed to detect communities, which indicate the high importance of discovering communities for understanding social networks and detecting useful and hidden patterns in the network. The goal of such analyses is to find a group of users with common characteristics. Basically, social networks are considered as graphs, so the analysis is also done using graph methods, in which nodes represent individuals and edges represent relationships between them. Since community detection is an NP-complete problem, several meta-heuristic approaches have been used to tackle this problem, mainly considering "*modularity*" as the objective function. In most approaches, modularity has been used, which suffers from the limitation of resolution and cannot detect communities that are small in size, and considers them in combination with large communities.

Methods: In this paper, a new hybrid algorithm of bee colony and genetics is proposed for community detection, which performs optimization using the "*balanced modularity*" fitness function. In this algorithm, parallel processing is used to speed up optimization, the genetic algorithm is used to create the initial population, and genetic operators are used in the search by bees.

Results: Experiments on well-known real-world networks, including karate, American football, dolphins, and political books, have shown that our method provides more accurate results than the state-of-the-art community detection methods.

Conclusion: The combined optimization of the bee colony and genetics not only provides a globally optimal solution but also does not need prior information about the number as well as the structure of communities.

How to cite this paper:
F. Akbari, E. Khanjari, "BSOGA: Community detection in social networks based on bee swarm optimization using genetic algorithm," J. Electr. Comput. Eng. Innovations, 14(2): 377-390, 2026.

DOI: 10.22061/jecei.2025.11871.838

URL: https://jecei.sru.ac.ir/article_2455.html

Introduction

The research field of network community detection has grown significantly, leading to a great scale of research. Today, most complex real-world systems in the fields of biology, sociology, and engineering are studied as networks of connected communities. Some examples of networks are: collaboration networks, transportation networks, the Internet, neural networks, social networks, etc. [1].

Some approaches have been suggested to discover communities, that indicate a wide importance of communities for perceiving social networks and discovering useful and hidden patterns in the network [2]. The purpose of such analyses is to find a group of users with common characteristics. Basically, social networks are considered as graphs, so the analysis is also accomplished using graph methods where nodes represent people and edges represent the relationships between them [3]. Most of the community detection algorithms exclusively use network topology information. Hierarchical clustering [4], [5], modularity optimization [6]-[8], spectral clustering [9], [10] and statistical inference [11], [12] have been investigated in the literature.

It is infeasible or very hard to find optimal solutions for all complex networks, and the value of a solution relies on the algorithm and characteristics of the intended network [1]. Due to the growing complexity of the network and, as a result, the exponential growth of the solution area, the classic methods are no longer suitable. The approach to solve the community detection problem can be based on heuristics or based on optimization. Discovery-based community detection algorithms are often based on quantitative assumptions, such as Girvan-Newman's algorithm, where the assumption is that edges between communities have higher between-edge values compared to edges within communities. Optimization-based community detection algorithms deal with the maximization of an objective function that gathers structural information in the network [13]. Most evolutionary algorithms are population-based algorithms that begin with a population of accidental solutions. Afterward, the population is progressed using a set of evolutionary operators, such as mutation and recombination operators to attain better results [14].

In this paper, a new algorithm based on a combination of the bee colony algorithm and genetics is introduced. In this method, the parallel technique has been used to accelerate the optimization and it has also been tried to cover the weaknesses of other methods. One of the weaknesses is the fitness function, in most approaches, modularity has been used, which suffers from the limitation of clarity and cannot recognize communities that are small in size, and considers it in combination with large communities. Therefore, in this method, the balanced modularity fitness function [15] has been used. The proposed method using the advantages of the genetic algorithm in creating an efficient initial population and using operators in the search of bees has led to great success in the detection of communities.

Problem statement

G(V, E) denotes the input graph, where *V* is the set of vertices and *E* is the set of edges. In the community detection problem, the target is to detect a clustering $C = \{C_1,..., C_N\}$ of the vertices, where the ratio of links inside cluster C_i is high compared to the ratio of links between C_i and other clusters. Suppose $R = \{R_1,..., R_k\}$ demonstrates *k* feasible partitions for the input network *G*. This problem is formulated as follows:

$$f(R^*) = \max f(R), \tag{1}$$

in which $R^* \in R$ is the desired partition, and *f* denotes the fitness function [1].

Contribution

The community detection topic is a basic matter in computer science because many network problems, like discovering, etc., are similar to the subgraph isomorphism matter that lately enjoyed a notable interest [1]. Most approaches consider modularity as a fitness function. Though modularity procedures suffer from the resolution limitation. These procedures cannot discover communities smaller than a threshold, and their aim is to merge these into larger communities, which leads to wrong conclusions [24]. To dominate the resolution limitation of the methods that rely on modularity fitness function, we exploit balanced modularity [15].

In this paper, a new meta-heuristic algorithm based on label propagation, locus-based adjacency representation, genetic algorithm, bee colony algorithm, and parallel processing is proposed for community detection. At first, a population of solutions is formed using locus-based adjacency representation and label propagation based on a genetic algorithm. In the next step, based on the combined meta-heuristic algorithm of genetics and colony bees, it performs the optimization process on the population. The main contributions are as follows:

- Combined optimization of genetic algorithm and bee colony.
- Reducing the problem search space by creating the initial population based on label propagation and locus-based adjacency representation.
- Using the balanced modularity as the fitness

function results in more natural communities compared to the real ones.
- Converting locations from continuous values to discrete values.
- Parallel processing capability of the proposed algorithm.
- A general comparative analysis with other community detection algorithms on different types of real-world datasets.

The rest of this paper is organized as follows: Section Related work reviews the background of community detection. Section Background explains the introduction. In the proposed method section, the BSOGA algorithm is characterized, and its pseudocode is provided. Section experiments demonstrate the effectiveness of BSOGA using experiments. eventually, the conclusion is presented in the section conclusion.

Related Work

Some algorithms have been suggested by researchers to discover communities, which shows the wide range of research in the field of identifying communities. Most of the existing methods exclusively use network topology information, including hierarchical clustering, spectral clustering, Graph partitioning algorithms, modularity optimization, and statistical inference. Hierarchical clustering methods are proper for graphs that have hierarchical structures [17].

The opinion of these methods is to create a binary tree that combines the same partitions based on the likeness between nodes. Blundell et al. [18] proposed an aggregate hierarchical algorithm known as the “Leuven method”. Their method first devotes each node to a community, and next these communities are joined together based on their likeness to form larger communities. Li et al. [19] proposed a recursive algorithm that divides the network into two subnetworks and uses a spectral clustering procedure in each iteration. This algorithm continues till a stopping criterion is met, finally yielding a binary tree hierarchy.

Spectral clustering is another algorithm for discovering communities. This method uses spectral features such as the eigenvalue spectrum of matrix representations (for example: adjacency matrix, Laplacian matrix, etc.) of networks to identify communities. Then, they apply a clustering algorithm such as k-means to identify communities. However, they are not reliable when the network is very sparse [2]. The first algorithm was suggested by Donath and Hoffman [20], who used the eigenvector matrix and eigenvalue matrix for graph clustering. Fielder [21] reached the bipartition of a graph using the second smallest eigenvalue of the Laplacian matrix—that is, the difference between the degree matrix and the adjacency matrix.

Partitioning algorithms are another method for identifying communities. These algorithms divide vertices into groups of predefined size to obtain the minimum number of links [22]. If we do not provide the number of partitions and do not create a partition with the minimum slice size, an unimportant solution will be output [16]. The Kernighan-Lin algorithm [23] is a heuristic method for partitioning graphs. It aims to minimize a scoring function that is the difference between the intra-community and inter-community links.

Modularity optimization is a method for identifying communities. It tries to find the maximum or minimum quality function that represents the quality of the community structure. Modularity is a concept that depends on the maximum difference between a real network and another form of the real network [2]. Algorithms based on modularity tend to maximize modularity [16], which is specified as follows:

$$Q = \sum_{c=1}^{n} \frac{L_c}{L} - \left(\frac{K_c}{2L}\right)^2 \quad (2)$$

where n is the total number of communities, L_c indicates the number of links within the community c, L indicates the total of links and K_c is the total degree of nodes. Modularity is in the range [-1,1].

In 2018, authors [24] proposed CC-GA algorithm for community discovery. This algorithm is based on the genetic algorithm and uses locus-based adjacency representation and clustering coefficient to create the initial population using modularity for optimization. In [25], a generation genetic algorithm (GGA+) is presented that contains effective initialization methods, and this approach allows for adaptive analysis of the specifications of a network. In [3], the authors presented an artificial bee colony algorithm. In this approach, several objective functions have been applied to discover communities. These functions are divided into two groups. The first group includes the functions to be minimized, which include: Expansion, Internal Density, Cut Ratio, etc. The second group includes the objectives to be maximized, which include: Modularity and Community Fitness. The algorithm is then started for each objective function and repeated several times (up to 100 times) on different datasets. With this work, the input graph was divided into a number of communities. In [26], the authors presented a bee colony-based community detection algorithm that uses local search. The proposed approach describes a multi-class algorithm based on bee swarm optimization that incorporates some characteristics to ameliorate other approaches. In each subgroup, some bees perform a local search to locate regions to generate good solutions. In this approach, it uses a master-slave topology to create a balance between exploration and mining. This method is

based on the Bee Swarm Optimization (BSO) algorithm presented by the authors [27]. This approach uses local search and modularity to gain the optimal solution. In this approach, the function-slave strategy is used. In this strategy, one of the groups that is responsible for producing the first is the slave swarm, and another group is the representatives of the slaves. In [28], The researchers presented an algorithm based on the bee swarm optimization. The algorithm uses modularity as an optimization. One solution is to partition the V nodes of the G network. Each cluster includes similar nodes and demonstrates a community. A feasible solution is specified by the number of communities. Xiao et al. [29] suggested a method based on classification, which was useful in modularity optimization.

Statistical inference is another method for identifying communities. One of these models is the random block model. This model depends on the maximum probability of entering communities in the given graph. However, the methods using the random block model need to know the number of communities, which is not known in advance in real-world networks [2]. Yang et al. [30] suggested a Dynamic Stochastic Block Model to model communities. Two versions have been suggested, one is online, which changes the model with time, while the other is offline, which learns the model with data gained at all time steps.

Background

Since the proposed algorithm combines locus-based adjacency representation, genetic algorithm, bee swarm optimization, and the label propagation, we first provide an overview of the background and the related concepts and methods in this section.

Label propagation

Suppose that a node a has neighbors , a_2,..., a_k and each neighbor has a label indicating the community. Next, it specifies its community through the labels of its neighbors. Then, initialize each node with unique labels and allow the labels to propagate. Next, dense groups of nodes quickly reach a consensus on a unique label. Finally, nodes with similar labels are merged as a community. At each step, every node updates its label through the labels of its neighbors [31].

Locus-Based Adjacency Representation

It is a representation that each chromosome is a separate graph. The size of each chromosome is equal to the number of nodes in the graph, and each node is connected to only one of its neighboring nodes [24]. LAR is a graph representation that uses a vector of elements with n locations, both location and vector elements represent the nodes of the graph. If j is assigned to location i. LAR shows the edges between elements, so a decoding stage is needed to obtain the community structure. The number of communities is determined at this stage. In fact, we do not require to determine the number of communities. Figure 1 shows a graph with nine nodes, in 1b the vector of a randomly generated chromosome. The community structure of this chromosome is shown in part c of Fig. 1.

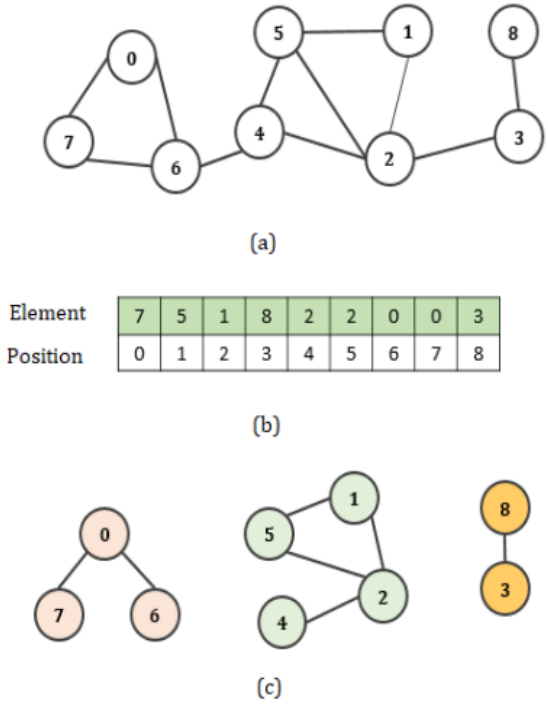


Fig. 1: Locus-based adjacency representation and related community structure [32].

Genetic Algorithm

Genetic algorithm optimization inspired by natural selection. It is a search algorithm based on population. New populations are produced by repeated application of operators to the population. Chromosome display, selection, crossover, mutation, and calculation of fitness function are the important components of the genetic algorithm [33]. The method of the genetic algorithm is that the population (Y) which includes n chromosomes is randomly initialized. The fitness of each chromosome is calculated in Y. Two chromosomes C_1 and C_2 are selected from Y with fitness value. The single-point crossover operator with probability (C_p) is applied to C_1 and C_2 to produce children called O. After that, the uniform mutation is applied to the generated offspring (O) with the probability of mutation (M_p) to produce new O' offspring. New children are included in the new population [33].

Bee Swarm Optimization

This algorithm simulates the feeding behavior of bees. In the bee colony algorithm, there is a population of food situations, and artificial bees change these food

situations over time. The bee colony algorithm consists of three types of bees that are performed according to the division of labor: employee bees, onlookers, and scout bees that fly in a subsequent search space to find the optimal solution [3]. Employee bees build the nest, clean the colony, feed the queen and male bees, protect the colony, and collect food [33]. Scouts' role in this process: Scouts are responsible for finding new food sources. Scouts fly around the hive to find food sources; moreover, the scout bee must tell other bees how far the flower source is. Onlooker bees collect nectar from the source and return to the colony, and empty the nectar into the food store [34]. In general, what bees do in a colony is this: employee bees explore food situations, while onlooker bees wait for information from employee bees about the nectar level of the food situation. Onlooker bees select the feeding position using information from the employee bees and use the selected feeding positions. Finally, scout bees find new random locations. Each solution in the search space contains a set of optimization parameters that demonstrate the locations of the food source. The number of employee bees is equal to the number of food sources. The quality of a food source is called "fitness value" and is related to its location [33]. The pseudocode of the bee colony algorithm is characterized in Fig. 2.

Input: A Network G = (V, E)
Output: Community membership assignments for network's nodes
Initialize the parameters: number of food sources, colony size cs, maximum number of iterations MaxNumber.
Randomly initialize the positions of the food sources $\overrightarrow{x_r}$
Repeat
  Phase: Employee bees
    Product a new solution $\overrightarrow{v_r}$ in the neighborhood of $\overrightarrow{x_r}$ using equation (4) and evaluate it. Then apply a greedy Selection between $\overrightarrow{v_r}$ and $\overrightarrow{x_r}$
  end
  Calculate the probability values P_r foreach $\overrightarrow{x_r}$
  Phase: Onlooker bees
    Product a new solution $\overrightarrow{v_r}$ in the neighborhood of $\overrightarrow{x_r}$ using equation (4), selected depending on $\overrightarrow{P_r}$ and evaluate it. Apply a greedy Selection between $\overrightarrow{v_r}$ and $\overrightarrow{x_r}$
  end
  Memorize the best solution achieved yet.
  Phase: Scout bees
  Determine the abandoned solution, if exists, replace it with a new produced solution using equation (3).
End
$t \leftarrow t + 1$

Until $t > MaxNumber$

Fig. 2: Bee colony optimization algorithm [3].

In the first stage, all vectors of food sources x are initialized by scout bees using (3) [3].

$$\overrightarrow{x_r}[i] = RC(i) \quad (3)$$

According to the above equation, $RC(i)$ represents the neighboring food source that is randomly selected.

Employee bees are looking for new food sources, such as v_r, which have more nectar in the neighborhood x_r. Then find the neighboring food source using (4) and evaluate profitability. x_k is a random food source and i is a randomly selected parameter. Then produce a new food source v_r, its profitability is computed, and a greedy choice between v_r and x_r is applied. The fitness function value obtained from (5). Where f_i is the objective function value of the solution i. The objective function is the "steering wheel" in the process (In this algorithm, this is f in the equation) [3].

$$\overrightarrow{v_r}[i] = \overrightarrow{x_k}[i] \quad (4)$$

$$fit_i = \{\frac{1}{1+f_i}, \qquad f_i \geq 0\ 1 + abs(f_i),\ f_i < 0 \quad (5)$$

The onlooker bees use the social knowledge provided by the employee bees. In every iteration of the algorithm, nectar information is shared by employee bees in the dance area. An onlooker bee then assesses the nectar information provided, applies a probabilistic method to choose one of the food sources, and follows the employee bee that finds the chosen food source. For this aim, the selection method based on profitability can be used. The value of the probability that the onlooker bee chooses the x_r is obtained through the following equation [3].

$$P_r = \frac{0.9 * fit_r(\overrightarrow{x_r})}{fit_{best}} + 0.1 \quad (6)$$

Scout bees use an accidental flight pattern to find a new food source and replace a new abandoned. Employee whose solutions do not improve through a predetermined number of trials, here called constraint, are discarded. Next, the scouts begin searching for new solutions randomly using (3) [3].

The Proposed Method

The proposed method, named Bee Swarm Optimization based on Genetic Algorithm (BSOGA), which is based on the BSO and GA algorithms and we describe it in this section. The BSOGA flowchart is painted in Fig. 3. the general method of BSOGA is as follows; first, the label propagation algorithm and locus-based adjacency representation are used to provide the initial population. Second, we obtain the fitness of each of the food sources created in the previous step using the balanced modularity. Then, according to the number of repetitions of the algorithm, the search is performed by employee bees, onlooker and scout bees using

genetic operators. Since the bee colony-based algorithms work based on integers and the created food sources are continuous, so the decoder is used in different stages. The rest of this section explains the details of BSOGA.

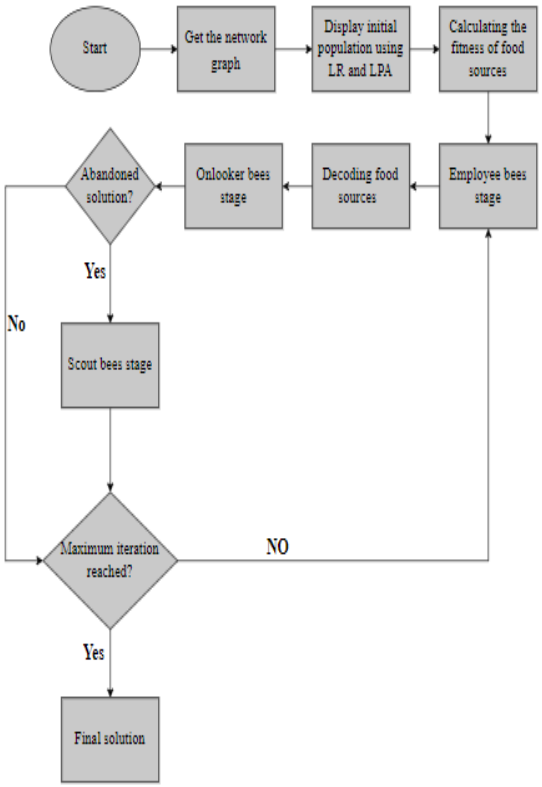


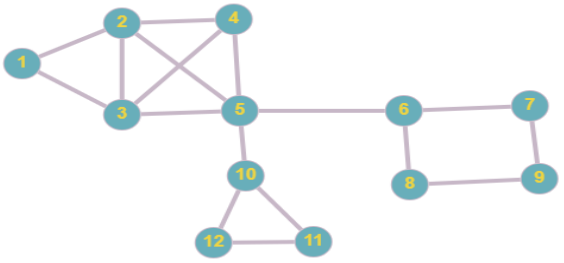
Fig. 3: Flow chart of the BSOGA algorithm.

Representation and Initialization

Each member of the population, i.e, a chromosome or food source, is the same as the set of nodes in the input network graph. Therefore, like the genetic algorithm, there is a population parameter called $npop$, which determines the number of food sources. So $X = \{x_1, x_2, \dots, x_m\}$ represents chromosomes or food resources, whose number is equal to the input population parameter. To create each chromosome using the locus-based adjacency representation, it starts randomly from a node and then randomly connects to a node among the neighboring nodes. Then it chooses another node randomly. and continues the previous procedure.

This procedure continues until there are no nodes left that have not been checked for connecting to the adjacent node. After the end of the method, a graph is obtained, which is divided into a number of communities. In the next step, in order to label each node in the communities created in the graph, the label propagation method is used. First, it starts with the first node and gives it a unique label (for example, community label 1), and then that node and its neighbors are given community label 1, and then it goes to the second node. If it has a label, then nothing happens.

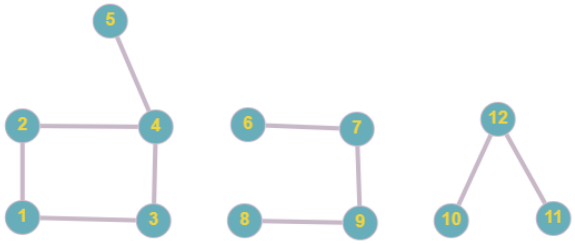
Otherwise, that node and its neighbors are labeled community 2, and so on, until the last node. Figure 4 (a) demonstrates a sample network graph with 12 nodes. Figure 4 (b) shows the locus-based adjacency method applied to the input graph. Figure 4 (c) shows the graphical representation of the chromosome in (b). Figure 4 (d) shows the corresponding label of each node in each community.



(a)

Node	1	2	3	4	5	6	7	8	9	10	11	12
Gene	3	1	4	2	4	7	9	9	8	12	12	11

(b)



(c)

Node	1	2	3	4	5	6	7	8	9	10	11	12
Label	1	1	1	1	1	2	2	2	2	3	3	3

(d)

Fig. 4: (a) An Example of input network graph; (b) locus-based adjacency representation of a chromosome; (c) The graph representation of the chromosome in (b); (d) Labeling nodes using label propagation.

Fitness Function: Balanced Modularity

To calculate the fitness of each chromosome or food source, we use the balanced modularity fitness function [15] which overcomes the problems that could arose by using modularity. Depending on the size of communities, the proposed algorithm considers a combination of the following two parameters as a fitness function.

Modularity

Modularity is a famous parameter of the quality of community structure in a network. Modularity is calculated as follows:

$$Q = \sum_{c=1}^{n} \frac{L_c}{L} - \left(\frac{K_c}{2L}\right)^2 \quad (7)$$

where n is the total of communities, L_c is the number of links within the community c, L is the total number of links in the network and K_c is the total degree of nodes. The modularity is in the range [-1,1]. A positive scale indicates the best community structure, while a negative scale demonstrates the absence of community structure. [24].

Community Balancing

To balance the size of the communities and prevent the formation of too big or too small clusters, the community balancing (CB) parameter is described as follows [15]:

$$CB = \frac{1}{\sqrt{\sum_{i=1}^{N} \left(\frac{T}{N} - t_i\right)^2} + 1} \quad (8)$$

where N is the total of communities, T is the total of nodes in the network, and t_i is the total of nodes in the i-th community.

Then, this parameter is used to define balanced modularity as follows:

$$f = c_1 . Q + c_2 . CB \quad (9)$$

where the two coefficients c_1 and c_2 are constant values and are determined so that their sum is one. In fact, these two parameters have a value between zero and one.

What has been done in this approach is that the parameters c_1 and c_2 have been assigned automatically. That is, instead of manually setting the values and evaluating them for a fixed number of results, parameterization is performed until it reaches the best state, then it stops.

Search among Food Sources

Each solution in the search space is a set of optimization parameters that demonstrates the location of the food source [33].

Stage of Employee Bees

Employee bees are in charge of searching for food sources. And their number is equal to the food sources in the population. At first, the fitness of each of the primary food sources is calculated using the fitness function, and the best source in terms of fitness is remembered.

For each food source, employee bees choose a neighboring food source randomly (according to the input network graph).

$$\overrightarrow{x_r}[i] = RNB(i) \quad (10)$$

According to the above equation, $RNB(i)$ represents the neighboring food source that is randomly selected. Then the location of the new food source is calculated using the following equation:

$$position\big(RNB(i)\big) = position(\overrightarrow{x_r}[i]) + \varphi * (position(\overrightarrow{x_r}[i]) - position(RNB(i)) \quad (11)$$

where ϕ is an array of uniform random numbers between -1 and +1. Positions are initially decimal numbers, and since the bee colony algorithm works with natural or discrete numbers, they should be converted from continuous to discrete state based on a heuristic. It means that the decoding process should be done on decimal places using a decoder. The heuristic that performs the transformation is described in Fig. 5.

```
// Decoding algorithm: continues to discrete
inpout: new position, clustering List, best solution, community
(X[i]). //clustering list is every neighbor of every node.
outpout: decoding
Nvar= size(new position)
expV = exp(-new position)
signV = abs((1- expV)./(1 + expV))
for v = 1:nVar
    K = clustering list{v}
    nj = size(K)
    if 0.25<signV(v) and nj>1    //signV is a
number between 0 and 1.

        k = randsample(nj,1)
        Position(v) = K(k)

    else
        if rand<0.2
            Position(v) = best solution(v)
        end
    end
end
return outpout
```

Fig. 5: Pseudocode of the continuous to discrete decoding algorithm.

As shown in the pseudocode above, for each node, it finds its neighbors, and the number of neighbors is determined. Then it is determined that if the signV of the desired node is in the range of **0.25** and **1**, a random number in the range of K that is equal to the number of neighbors of the desired node and **1** is assigned to it, and otherwise the position of the desired node is equal to the position of the best solution so far. Then the

employee bees use a uniform crossover between the two food sources. The reason for using the uniform intersection operator is that it leads to the avoidance of local optima and helps to improve the structure of the network community [24]. Since this procedure involves randomization, it may produce offspring with low fitness values. We ignore those children. Then, among the children, a child who is fitter is chosen, and among the selected children, a source of food is chosen that is fitter. Then, in order to maintain population diversity in the selected source, we use mutation. The procedure of mutation is that a node is randomly selected in the food source, and then a neighboring node of the selected node is selected, and then a neighbor of the neighboring node is selected and replaced with each other, if the initially selected node is separately. No leap is made. This procedure is performed by employee bees for all food sources. The pseudocode of genetic operators used by employee bees is shown in Figs. 6 and 7.

```
//Uniform crossover
inpout: gamma=0.05, //parameter of crossover
 new position, //position of (x_r[i]) neighbor
best solution,
 position(x_r[i]),
 VarMin=0, //lower Bound of Variables
 VarMax=1 // upper Bound of Variables
outpout: crossover operator between two food sources.
alpha=randarray(-gamma,1+gamma,  size(position(x_r[i])))  //create  random
array between  first and second value
y1=alpha.* position(x_r[i])+(1-alpha).*(new position))
y2=alpha.*new position+(1-alpha).* position(x_r[i])
y1 = min (max (y1, VarMin, VarMax)
y2 = min (max (y2, VarMin, VarMax)

//decoding y1 and y2 with decoding pseudo code
C1 = decoding pseudo code (y1, ClusteringList, community (best solution),
community (position(x_r[i])))
C2 = decoding pseudo code (y2, ClusteringList, community (best solution),
community (new position))
//calculate fitness function of C1 and C2 using formula 9
[cost1, out1] = CostFunction(C1)
[cost2, out2] = CostFunction(C2)

if cost1>cost2
   x = y1
   C = C1
   out = out1
   cost = cost1
else
   x = y2
   C = C2
   out = out2
   cost = cost2
end
return outpout
```

Fig. 6: Pseudocode of the uniform intersection operator algorithm.

```
// mutation
inpout: mu=0.1, //parameter of mutation
 new position, //position of (x_r[i]) neighbor
 best solution,
 VarMin=0, // lower Bound of Variables
 VarMax=1 //upper Bound of Variables
outpout: crossover operator between two food sources.
nVar = size(x)
nmu = ceil(mu*nVar)
j=randsample(nVar,nmu)
sigma=0.2*(VarMax-VarMin)
y=x;
y(j)=x(j)+sigma*randn(size(j))

new Position = min(max(y,VarMin),VarMax)
//decoding new  Position with decoding pseudo code
//C is in crossover pseudo code
new Community = decoding pseudo code (new Position, ClusteringList,
Community (best solution), C)
// calculate fitness function of new community using formula 9
[new Cost, new Out] = CostFunction (new Community);

if new Cost<cost
  new Cost = cost
  new Position = x
  new Community = C
  new Out = out
end
return outpout
```

Fig. 7: Pseudocode of the mutation operator algorithm.

Then the employee bees calculate the fitness of all food sources using (5) and (9) and the probability of their sources using (12) and provide this information to the onlooker bees.

$$p = \frac{1}{size(npop)} \tag{12}$$

where, $npop$ is the initial population size.

Stage of Onlooker Bees

The onlooker bees obtain the location of a food source based on the roulette wheel selection method and the probability calculated by the employee bees from among the food sources obtained by the employee bees. The pseudocode for calculating the location of the food source is shown in Fig. 8.

The onlooker bees then choose a food source at random.

And between the two previous and current food sources, the crossing and mutation procedure, similar to Figs. 7 and 8, is performed. Then, as a result of genetic manipulation, a new food source is created by the onlooker bees.

```
// RouletteWheelSelection algorithm pseudocode
inpout: p, // Probability calculated by employee bees
pick = 0,
offset = 0.0,
npop //population size
outpout: Select Source Site
rndNumber = random number
for (i = 0; i < npop; i++)
   offset += chromosome[i].probability
   if (rndNumber < offset)
      pick = i
      break
   end
end
return Source Site
```

Fig. 8: Roulette wheel selection algorithm pseudocode.

Then, the fitness function is applied to the primary food source and the new food source, and whichever one has more fitness is chosen between the two sources. The important point is that there is a parameter C which is of matrix type and its size is $npop$ *1. What this matrix does is that during the optimization steps, if the condition of greater than or equal to the current fitness is not met, a unit is added to the corresponding location of that resource in the matrix. Actually, this matrix is a counter.

In this approach, there is a parameter called L, whose value is assumed to be **10**. It is actually a threshold value. Therefore, if the value of the counter related to the food source obtained in the matrix C is less than or equal to the threshold value, the onlooker bees introduce this food source as the final solution to the community detection the problem, otherwise the stage of scout bees begins.

Stage of Scout Bees

Scout bees are responsible for randomly searching among primary food sources. In fact, if the food source chosen by the scout bees does not meet the condition of the threshold, they choose another food source randomly. Therefore, they search among all the food sources in the initial population based on the information stored for them from the beginning of the algorithm. One of the parameters used by scout bees is the threshold value (L). What the scout bees do is to repeat an operation between all food sources if the element value in matrix C is bigger than or equivalent to the threshold value (here **10**), such as: finding a new location for the food source, clustering and finding fitness of the food source. Then, it searches again among the food sources and if the fitness value of each one was higher than the fitness of the best solution, it returns it as the final solution of the community detection problem. The pseudocode of scout bees is shown in Fig. 9.

```
// scout bees pseudo code
inpout: L=10, VarMin=0, VarMax=1, Varsize=npop
outpout: select new food source

  for i = 1:nPop
    if C(i) >= L
      pop(i).Position = create array between VarMin and VarMax size of VarSize
      pop(i).Community = clustering i food source
      calculate fitness value of pop(i).community using formula 9
      C(i) = 0
    end
  end

  //Update Best Solution Ever Found
  for i = 1:nPop
    if pop(i).Cost >= BestSol.Cost
      BestSol = pop(i);
    end
  end
return outpout
```

Fig. 9: Pseudocode of the scout bees algorithm.

Pseudocode of the Proposed Approach

In the previous parts, the general procedure of the proposed approach has been described. Figure 10 shows the pseudocode of the proposed approach.

Parallelism in the Proposed Method

In this section, we describe the parallel mode of the proposed approach.

As it is clear from the general procedure of this approach, some parts repeatedly perform a series of operations, like what employee bees do for all food sources.

In fact, they perform a series of identical operations for all food sources that have no dependence on each other. Also, parallelization is used in the stage of onlooker bees. So, instead of performing a series of tasks in a loop, tasks can be performed simultaneously for all food sources. In this case, the time complexity is greatly reduced. Figure 11 shows the pseudocode of the parallelism in the proposed approach.

```
// BSOGA pseudo code
inpout: npop, // size of population
 MaxIt, //number of iteration
L=10, //Abandonment Limit Parameter
C  //matrix for Abandonment Counter
outpout: optimal BSOGA result
Create and initialize population of Chromosome in npop size using section
Representation and initialization
Calculate fitness of each Chromosome using equation 9
while I <= MaxIt
     employee bees step
    for each food source in population (npop)
        Random selection of a neighboring food source
        Calculate position of neighboring food source using equation 10
        Decoding of positions from continuous to discrete mode using figure 5
       Uniform crossover between previous and new food source figure 6
       Mutation on food source figure 7
   for each food sources calculate probability using equation 12
    onlooker bees step
    select one position of food source with roulette wheel and probabilities in
employee bees step
     select one food source randomly that not equal to previous
    calculate position of new food source using equation 10
   Decoding of position from continuous to discrete using figure 5
   Uniform crossover between previous and new food source figure 6
   Mutation on food source figure 7
   if fitness function value of food source ≤ L
    this food is optimal result solution
  else
    go scout bees step

     scout bees step
     for each food source in population (npop)
       if C(i) ≥ L
        select position for food source i using equation 11
        clustering food source I using section Representation and initialization
        calculate fitness function using equation 9
        C(i)=0
   Update Best Solution Ever Found
 i= i+1
end while
return outpout
```

Fig. 10: Pseudocode of BSOGA algorithm.

```
// PBSOGA pseudo code
inpout: npop, // size of population
 MaxIt, //number of iteration
L=10, //Abandonment Limit Parameter
Co,  //matrix for Abandonment Counter
Pop //npop*1 matrix that each element is an array

outpout: optimal PBCOGA result
Create and initialize population of Chromosome in npop size
using section Representation and initialization
Calculate fitness of each Chromosome using equation 9
while i <= MaxIt
      employee and onlooker bees step
      parfor i=1: npop
              CD_BCO_employee_GA recall //employee bees
stage
              CD_BCO_Onlooker_GA recall //onlooker bees
stage
      end
      scout bees step
          for each food source in population (npop)
            if Co(i) ≥ L
              select position for food source i using equation 11
              clustering food source I using section
              Representation and initialization
              calculate fitness function using equation 9
              Co(i)=0
         Update best solution ever found
 i= i+1
end while
return outpout
```

Fig. 11: psuedocode of PBSOGA algorithm.

Complexity analysis of the proposed method

Let n be the total nodes, m the total edges, $npop$ the total initial population, and $MaxIt$ the total number of iterations of the original BSOGA loop. Locus-based adjacency representation method and label propagation are used to create the initial population. The time complexity of the label propagation is almost linear [31], that is, it is of the order of $O(npop * n)$. The locus-based adjacency representation method consists of two parts. The first is to calculate the adjacency list for each node. This part of the time order is $O(n * m)$. The second is to randomly select a neighboring node for each node. This part is of the order of $O(1)$.

Therefore, its total time complexity is of the order of $O(npop * n * m)$. Therefore, the total time complexity of creating the initial population is $O(npop * n * m)$. The fitness function is of the order of $O(npop * n^2)$. We examine the time complexity related to the search steps by employee, onlooker and scout bees, for each step separately. The time complexity of the employee bees stage is $O(npop * n^2)$. The time complexity of the onlooker bees stage is $O(n^2)$. The time complexity of the scout bees stage is $O(npop * (n * m + n^2))$. Therefore, the time complexity of the entire search stage by all three groups is equal to $O\big(MaxIt * npop * (n * m + n^2)\big)$.

Experiments

The BSOGA has experimented on the real-world datasets. The results represent the efficacy of the BSOGA. In all experiments, BSOGA is run fifty times, and the mean results are reported. In this section, the results of experiments are reported and BSOGA is compared with other mentioned algorithms. The adjustable parameters of BSOGA algorithm in the implementation part include: $npop$ (number of initial population), $MaxIt$ (number of repetitions of the algorithm), $gamma$ (intersection operator parameter) and mu (intersection

operator parameter). In the implementation, gamma (Crossover rate) and mu parameters are assumed to be **0.05** and **0.1**, respectively.

The important point is that the parameters c_1 and c_2 are automatically adjusted in the balanced modularity fitness function.

Experimental Setup

This section, containing explanations of real-world datasets, comparison methods, and evaluation criteria. All the experiments are operated on the Windows **10**, Corei3 **2** GHz CPU, **4** GB memory. All codes are written in MATLAB R2021b.

Datasets

The experiments have been executed on well-known real-world network datasets, including Zachary's Karate Club [35], Dolphins Social Network [36], Krebs' Political books [24], American Football Clubs [24] (see Table 1).

Table 1: Real-world datasets used for the experiments

Dataset	#NC	#Edges	#Nodes	Description
karate club	2	78	34	Karate social network from a sports club [35]
Dolphins	2	159	62	Dolphins social network [36]
Political books	3	441	105	Political books [24]
American football	12	613	115	American College Games Network [24]

Evaluation Criteria

To compare BSOGA and another algorithms on the real-world networks, the modularity and the normalized mutual information (NMI) [37] are used. The NMI is used to meter the similarity between clustering reached by a method and that reached by another method. The NMI of two community A and B of a network is described as follows:

$$NMI(A,B)=\frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B}C_{ij}log(\frac{C_{ij}N}{C_iC_j})}{\sum_{i=1}^{C_A}C_ilog(\frac{C_i}{N})+\sum_{j=1}^{C_B}C_jlog(\frac{C_j}{N})} \quad (14)$$

C is the confusion matrix that element *cij* is the number of nodes in community *i* of the cluster *A* are in the cluster *j* of the partition *B* as well, *CA* (*CB*) demonstrate the number of clusters in the partition *A* (*B*), and *ci* (*cj*) demonstrate the totality of the elements of *C* in row *i*. If *A* and *B* are matched, *NMI*(*A*, *B*) = 1. If *A* and *B* are dissimilar, *NMI*(*A*, *B*) = 0. The NMI be in the scale of 0 and 1.

Comparison Algorithms

The results of the BSOGA algorithm have been compared with BSOCD-LS [26] BSOCD [28], LPA [38], BCALS [39], CCGA [24], GGA+ [25], EA [40], CNM [41] and Memetic [42].

Experiment Results

In this section, BSOGA is compared with other approaches. In Fig. 12, the BSOGA algorithm is compared with several algorithms based on bee colony and based on genetics and non-evolutionary based on modularity on four datasets of karate club, dolphins, political books and American football. According to the figure, the BSOGA algorithm has performed better than the rest of the algorithms and the best modularity in it has been much better than the rest of the algorithms. The results of the evaluations for modularity in mean, worst, and best modes and NMI for the Karate Club, Dolphins, Political Books, and American Football datasets are displayed in Table 2.

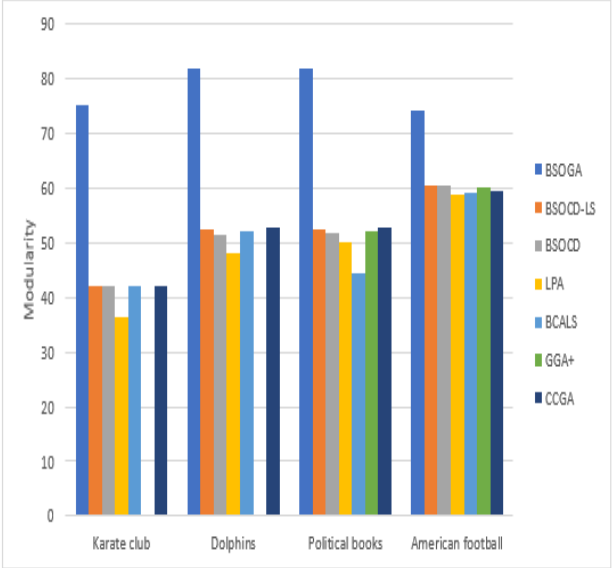


Fig. 12: Comparison of modularity-based BSOGA with six algorithms on karate, dolphins, political books, and american football datasets.

Table 2: The results of the evaluations in different situations in the BSOGA algorithm

Dataset	NMI	Q_{worst}	Q_{mean}	Q_{best}	#NC
karate club	0.84	0.75	0.75	0.75	2
Dolphins	0.89	0.82	0.82	0.82	2
Political books	0.75	0.80	0.81	0.82	3
American football	0.62	0.65	0.69	0.74	6

In Table 3, BSOGA algorithm is compared with EA [40], CNM [41] and Memetic [42] algorithms based on modularity and the number of detected communities.

Table 3: The results of the evaluations in different situations in the BSOGA algorithm

Algorithm	karate club		Dolphins		American football		Political books	
	#NC	M	#NC	M	#NC	M	#NC	M
BSOGA	**2**	**0.75**	**2**	**0.82**	6	**0.74**	**3**	**0.82**
EA	3	0.38	3	0.46	**7**	0.56	4	0.52
CNM	3	0.38	4	0.495	6	0.55	4	0.501
Memetic	12	0.402	4	0.518	**7**	0.604	4	0.523

According to the above table, it can be seen that the BSOGA algorithm has performed better compared to other algorithms.

By looking at the modularity and the number of detected communities, it is clear that it is very close to the real detected state.

The following figures show examples of results on karate, dolphins, political books, and American football datasets.

The results have been evaluated based on 50 iterations and an initial population size of 20. In each of the figures, the right part of the communities detected in the proposed algorithm and the left part are the communities detected in the real state. In the figures, the colors indicate the communities.

As it is clear from the figures, the proposed algorithm has detected the communities very close to the real state.

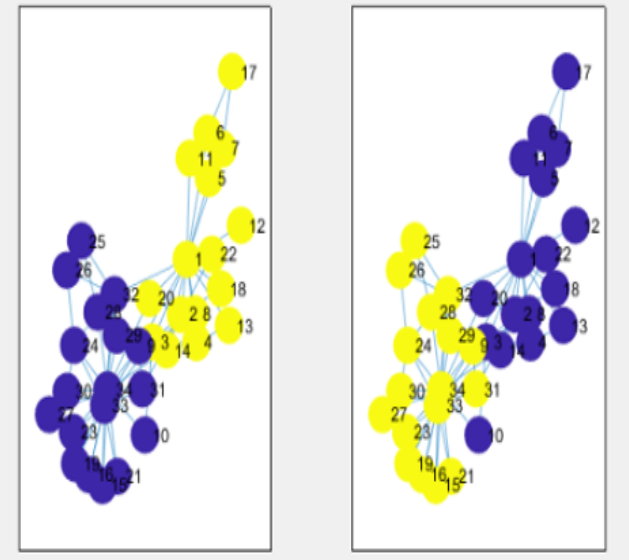


Fig. 13: Comparison of community detection with BSOGA algorithm in karate club dataset.

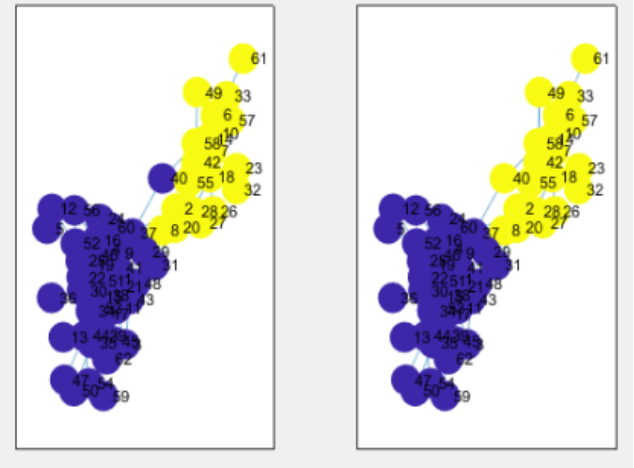


Fig. 15: Comparison of community detection with BSOGA algorithm in dolphins dataset.

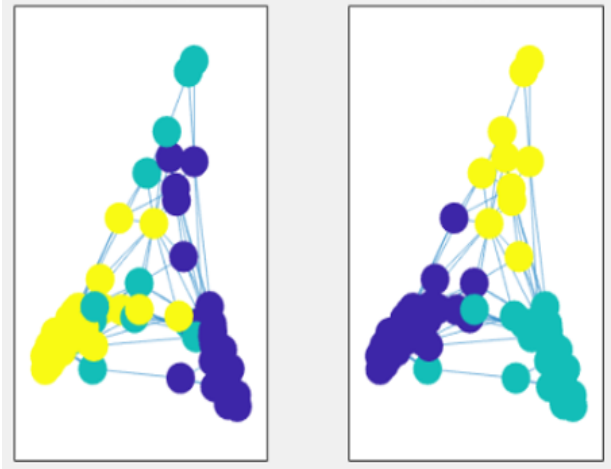


Fig. 14: Comparison of community detection with BSOGA algorithm in political books dataset.

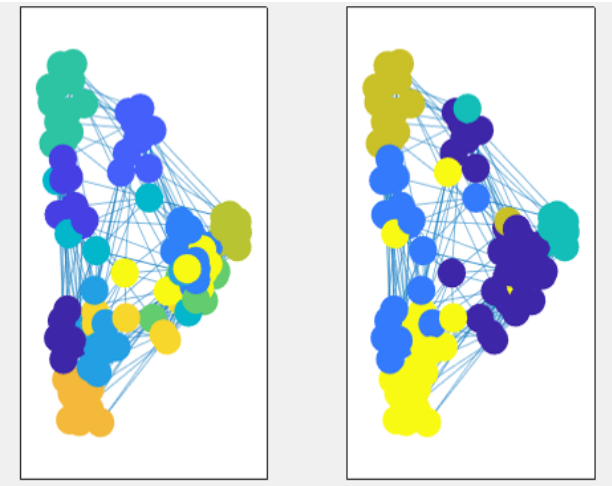


Fig. 16: Comparison of community detection with BSOGA algorithm in football dataset.

Conclusion

In this paper, we proposed a novel algorithm for community detection in social networks (we named it BSOGA). Community detection is described as an optimization problem in the context of genetic algorithm

where the search operation takes place using bee colony algorithm. As the naturality of the resulting communities is a challenge in such a problem, we exploit balanced modularity as the fitness measure to provide a more natural result when compared to the real-world communities. The proposed algorithm exploits label propagation and locus-based adjacency representation to generate and represent the initial population. The combined optimization of bee colony and genetics not only provides globally optimal solution but also it does not need prior information about the number as well as the structure of communities. We conducted extensive experimentations on real-world network datasets which demonstrate the performance of BSOGA in terms of quality of the resulting communities compared to the state-of-the-art algorithms. Moreover, the proposed algorithm could partially provide parallel operations to accelerate the efficiency of the method. The proposed algorithm could be extended/adapted to be used in many applications, including targeted product recommendation in e-commerce networks, effective routing in telecommunication networks, real-word transportation networks, and fraud detection in financial transactions. As the real-world social networks are of dynamic nature, extending the current method on dynamic and evolving networks is of great interest in the literature. We intended to tackle such a problem as future work. Another research direction could be the investigation of different meta-heuristic optimization methods for community detection problem.

Author Contributions

F. Akbari methodology, preparing the draft. E.Khanjari Supervision, reviewing and finalizing the manuscript.

Acknowledgment

The authors would like to thank the editor and anonymous reviewers.

Conflict of Interest

The authors declare no potential conflict of interest regarding the publication of this work.

Abbreviations

BSOGA	Community detection in social networks based on bee swarm optimization using genetic algorithm
PBSOGA	Parallel Community detection in social networks based on bee swarm optimization using genetic algorithm
CD	Community Detection

References

[1] D. Abdulhadi Abduljabbar, S. Z. Mohd Hashim, R. Sallehuddin, “Nature‐inspired optimization algorithms for community detection in complex networks: A review and future trends,” Telecommun. Syst., 74: 225-252, 2020.

[2] A. Karataş, S. Şahin, “Application areas of community detection: A review,” in Proc. International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), 2018.

[3] A. I. Hafez, H. M. Zawbaa, A. Ella Hassanien, A. A. Fahmy, “Networks community detection using artificial bee colony swarm optimization,” in Advances in Intelligent Systems and Computing, 303, 2014.

[4] F. Dabaghi Zarandi, M. Kuchaki Rafsanjani, “Community detection in complex networks using structural similarity,” Physica A, 503: 882-891, 2018.

[5] D. Jin, Z. Yu, P. Jiao, S. Pan, D. He, J. Wu, P. S. Yu, W, Zhang , “A survey of community detection approaches: From statistical modeling to deep learning,” IEEE Trans. Knowl. Data Eng., 35(2): 1149-1170, 2023.

[6] C. C. Aggarwal, H. Wang, “Managing and mining graph data,” in Advances in Database Systems, 40: 275–301, 2010.

[7] S. Jia, L. Gao, Y. Gao, J. Nastos, Y. Wang, X. Zhang, H. Wang, “Defining and identifying cograph communities in complex networks," New J. Phys., 17, 2015.

[8] L. Yang, X. Cao, D. He, C. Wang, X. Wang, W. Zhang, “Modularity based community detection with deep learning,” in Proc IJCAI: 2252-2258, 2016.

[9] M. Newman, M. Girvan, “Finding and evaluating community structure in networks,” Phys. Rev. E., 69, 2004.

[10] P. Zhang, C. Moore, “Scalable detection of statistically significant communities and hierarchies, using message passing for modularity,” PNAS, 111(51): 18144-18149, 2014.

[11] M. Fanuel, C. M. Ala´ız, J. A. K. Suykens, “Magnetic eigenmaps for community detection in directed networks,” Phys. Rev. E, 95, 2017.

[12] Y. Li, K. He, D. Bindel, J. E. Hopcroft, “Uncovering the small community structure in large networks: A local spectral approach,” in Proc. the 24th International Conference on World Wide Web: 658-668, 2015.

[13] A. Anandkumar, R. Ge, D. J. Hsu, S. M. Kakade, “A tensor approach to learning mixed membership community models,” J. Mach. Learn. Res., 15(1): 2239-2312, 2014.

[14] H. Faris, I. Aljarah, M. Azmi Al-Betar, S. A. Mirjalili, “Grey wolf optimizer: A review of recent variants and applications,” Neural Comput. Appl. 30: 413-435, 2017.

[15] E. Jokar, M. Mosleh, M. Kheyrandish, “GWBM: an algorithm based on grey wolf optimization and balanced modularity for community discovery in social networks,” J. Supercomput., 78: 7354–7377, 2022.

[16] M. A. Javeda, M. S. Younisa, S. Latifa, J. Qadirb, A. Baigc, “Community detection in networks: A multidisciplinary review,” J. Network Comput. Appl., 108: 87-111, 2018.

[17] L. Zhang, Q. Ye, Y. Shao, C. Li, H. Gao, “An efficient hierarchy algorithm for community detection in complex networks,” Math. Probl. Eng., 874217, 2014.

[18] V. D. Blondel et al., “Fast unfolding of communities in large networks,” J. Stat. Mech. Theory Exp., 2008.

[19] T. Li et al., “Hierarchical community detection by recursive partitioning,” J. Am. Stat. Assoc., 117(538): 951-968, 2020.

[20] W. E. Donath, A. J. Hoffman, “Lower bounds for the partitioning of graphs,” IBM J. Res. Dev., 17(5): 420-425, 1973.

[21] M. Fiedler,”Algebraic connectivity of graphs,” Czech. Math. J., 23(2): 298-305, 1973.

[22] A. Pothen, “Graph partitioning algorithms with applications to scientific computing,” in Parallel Numerical Algorithms: 323-368, 1997.

[23] B. W. Kernighan, S. Lin, “An efficient heuristic procedure for partitioning graphs,” Bell Syst. Tech. J., 49(2): 291-307, 1970.

[24] A. Said, R. Ayaz Abbasi, O. Maqbool, A. Daud, Na. Radi Aljoha, ”CC-GA: A clustering coefficient based genetic algorithm for detecting communities in social networks,” Appl. Soft Comput., 63: 59-70, 2018.

[25] M. Guerreroa, F. G. Montoyab, R. Ba˜nosb, A. Alcaydeb, C. Gila, “Adaptive community detection in complex networks using genetic algorithms,” Neurocomputing, 266: 101-113, 2017.

[26] Y. Belkhiri, N. Kamel, H. Drias, “Multi-swarm BSO algorithm with local search for Community Detection Problem in complex environment,” in Proc. Comput. Collective Intelligence (ICCCI 2019): 321-332, 2019.

[27] H. Drias, S. Sadeg, S. Yahi, “Cooperative bees swarm for solving the maximum weighted satis ability problem,” in Proc. Computational Intelligence and Bioinspired Systems (IWANN 2005): 318-325, 2005.

[28] Y. Belkhiri, N. Kamel, H. Drias, S. Yahiaoui, “Bee swarm optimization for community detection in complex network,” in Proc. Recent Advances in Information Systems and Technologies (WorldCIST 2017): 73-85, 2019.

[29] x. Guo, J. Su, H. Zhou, C. Liu, J. Cao, L. Li, “community detection based on genetic algorithm using local structural similarity,” IEEE Access, 2019.

[30] T. Yang, Y. Chi, S. Zhu, Y. Gong, R. Jin, “ Detecting communities and their evolutions in dynamic social networks a Bayesian approach,” Mach. Learn., 82: 157-189, 2011.

[31] Raghavan UN, Albert R, Kumara S, “Near linear time algorithm to detect community structures in large-scale networks,” Phys. Rev. E., 76(3): 036106, 2007.

[32] Z. Beldi, M. Bessedik, “A new brainstorming based algorithm for the community detection problem,” in Proc. 2019 IEEE Congress on Evolutionary Computation (CEC): 2958-2965, 2019.

[33] S. Katoch, S. Singh Chauhan, V. Kumar, “A review on genetic algorithm: past, present, and future,” Multimedia Tools Appl., 80: 8091–8126., 2020.

[34] k. Singh Kaswan, S. Chodhary, K. Sharma , “Application of artificial bee colony theqnique: survey,” in Proc. 2nd International Conference on Computing for Sustainable Global Development (INDIACom): 1660-1664, 2015.

[35] W. Zachary, “An information flow model for conflict and fission in small groups,” J. Anthropol. Res. 33(4): 452–473, 1997.

[36] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, S. M. Dawson, “The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations,” Behav. Ecol. Sociobiol. 54(4): 396–405, 2003.

[37] L. Danon et al., “Comparing community structure identification,” J. Stat. Mech. Theory. Exp., 2005.

[38] X. Zhou, K. Yang, Y. Xie ,C. Yang, T.Huang, “A novel modularity-based discrete state transition algorithm for community detection in networks,” Neurocomputing, 334: 89-99, 2019.

[39] Z. Bu, H. J. Li , J. Cao, Z. Wang , G. Gao, “Dynamic cluster formation game for attributed graph clustering,” IEEE Trans. Cyber., 49(1): 328-341, 2017.

[40] H. Banati, N. Arora, “TL-GSO: - A hybrid approach to mine communities from social network” in Proc. 2015 IEEE International Conference on Research in Computational Intelligence and Communication Networks, 2015.

[41] S. Bilal, M. Abdelouahab, “Evolutionary algorithm and modularity for detecting communities in networks,” Physica A, 473: 89-96, 2017.

[42] S. Chand, S. Mehta, “Community detection using nature inspired algorithm,” in Hybrid Intelligence for Social Networks: 47-76, 2017.

Biographies

Fatemeh Akbari was born in Sari city of Mazandaran province. She completed her bachelor's degree in Computer Software Engineering at Noshirvani University of Technology in Babol and graduated in 2020. Also, she has studied a master's degree in Computer Software Engineering at the University of Science and Technology and will graduate in 2022. She became a computer teacher in 2022 and teaches in a conservatory. Her areas of interest include data mining and social network analysis.

- Email: fatemeh.akbari74@yahoo.com
- ORCID: 0009-0008-79381753
- Web of Science Researcher ID: NA
- Scopus Author ID: NA
- Homepage: NA

Eynollah Khanjari is an Assistant Professor of Computer Science at the Iran University of Science and Technology, Tehran, Iran. He received his Ph.D. from University of Tours, France, in Software Engineering, M.Sc. degree in Software Engineering from Iran University of Science and Technology, and B.S. from Sharif University of Technology, Iran. His areas of research interest and specialization include Knowledge Discovery and Data Mining, Database and Information Systems, and Social and Complex Networks Analysis. He is the director of data engineering research group at the school of computer engineering.

- Email: Khanjari@iust.ac.ir
- ORCID: 0000-0002-2304-1804
- Web of Science Researcher ID: Google Scholar
- Scopus Author ID: 57205202272
- Homepage: https://its.iust.ac.ir/profile/en/khanjari