Community Detection in Complex Networks: A Novel Approach Based on Ant Lion Optimizer

Maninder Kaur(✉) and Abhay Mahajan

Computer Science and Engineering Department,
Thapar University, Patiala 147004, India
manindersohal@thapar.edu

Abstract. The problem of community detection in complex networks has established an increased amount of interest since the past decade. Community detection is a way to discover the structure of network by assembling the nodes into communities. The grouping performed for the communities encompasses denser interconnection between the nodes than community's intra-connections. In this paper a novel nature-inspired algorithmic approach based on Ant Lion Optimizer for efficiently discovering the communities in large networks is proposed. The proposed algorithm optimizes modularity function and is able to recognize densely linked clusters of nodes having sparse interconnects. The work is tested on Zachary's Karate Club, Bottlenose Dolphins, Books about US politics and American college football network benchmarks and results are compared with the Ant Colony Optimization (ACO) and Enhanced Firefly algorithm (EFF) approaches. The proposed approach outperforms EFF and ACO for Zachary and Books about US politics and produces results better than ACO for Dolphins and EFF for American Football Club. The conclusion drawn from experimental results illustrates the potential of the methodology to effectively identify the network structure.

Keywords: AntLion optimization · Community detection · Modularity · Social networks

1 Introduction

In the past decade, the research on complex networks has become more eye catching in the fields of mathematics, sociology, physics, biology (Ferrara and Fiumara 2011; Newman 2003; Clauset et al. 2004). The topological structure of complex systems can simply be represented as a complex network with connected nodes. The existing networks of well-known social media and online social networking websites like Twitter, Facebook, and Google+ (Ferrara and Fiumara 2011), characterize the system by means of links and nodes. The nodes signify the systems and links represent the relationship between the connecting or interrelating nodes. The network links in different type of areas represent different types of relationship e.g. animal's physical proximity, interconnectivity of infrastructures, human friendship, organizational structures, web hyperlinks and abstract relationships like similarity between data points.

K. Deep et al. (eds.), *Proceedings of Sixth International Conference on Soft Computing for Problem Solving*, Advances in Intelligent Systems and Computing 546, DOI 10.1007/978-981-10-3322-3_3

The presence of communities shows the structure of the networks existing in nature. Communities, named as modules/clusters, are the groups of relatively connected nodes, and are said to be inherent arrangement of the networks present in nature (Newman 2003). Nodes of the same community or cluster typically share common interesting features such as a function, purpose and interest. For this reason, one of the most crucial problems in network analysis is community detection.

Various contributions have been anticipated by the researchers in the field of community detection in recent years to detect communities in complex networks, with each methodology being classified according to its algorithm type. Many authors dug into the field of community detection by proposing various analytical approaches. The authors (Newman and Girvan 2004) proposed one of the first successful betweenness based divisive algorithms for community detection. The proposed approach determined the communities but it could not determine the strength of communities formed. Later on the author (Newman 2004) proposed an algorithm based on agglomerative clustering which used the modularity function determining the strength of communities. This algorithm was efficient in case of speed but in practice, the modularity produced by this algorithm was not high. The authors (Clauset et al. 2004) observed that the Newman's approach was not efficient for sparse networks as well as it was also inefficient with respect to time and memory. They improved the Newman's original algorithm with the help of max-heap. Their algorithm was the first algorithm used for analyzing large networks (about 10^6 nodes). The algorithm has a drawback that it might form large communities in the early phase at the cost of existing small communities. The author (Newman 2006) proposed a new method for optimizing the fitness function with the help of associated matrix eigenvectors and eigenvalues. The authors (Schuetz and Callisch 2008) proposed a variation of (Clauset et al. 2004) algorithm by using the "Touched-Community-Exclusion-Rule" (TCER) in the implementation. The algorithm had same complexity as (Clauset et al. 2004) algorithm but instead of creation and maintenance of the max-heap, it required the computation of pairwise gains that increases the computational cost. The authors (Lambiotte et al. 2008) proposed a greedy hierarchical clustering algorithm named as Louvain method considering modularity as objective function. The proposed algorithm is fast but on multiple core architecture the sequential corrections make it slow and it is also inefficient to be applied for very large networks. The authors (Le Martelot and Hankin 2013) proposed a new method based on local and global criteria, with global criteria algorithm almost similar to Louvain method but with an advantage of application for multiscale detection of communities. The local criterion has its advantage that it can be used for overlapping communities.

Community detection being an NP hard problem (Fortunato 2010) various heuristic approaches have been anticipated that assist in detecting communities of the complex networks. There are the widely used properties to calculate the quality factors of the clustered structure of networks. Amongst them, the most well-liked method is reliant on the optimization of the profit function recognized as "modularity" over the feasible partitions of a network. It is one approach to community detection that has been set primarily competent. A growing number of evolutionary approaches (Pizzuti 2008; Honghao et al. 2013; Shang et al. 2013; Hafez et al. 2014; Ma et al. 2014) dependent on modularity optimization in detecting communities have been published in past few years. The author (Pizzuti 2008) presented a (GA) algorithm for uncovering communities in large networks named GA-Net. They introduced the concept named community score. The author also introduced the Safe individual criteria in GA to avoid useless computation making the algorithm efficient. The algorithm showed effective results in finding the communities in networks for synthetic as well as real world data set. The author (Pizzuti 2012) further modified his work by presenting a new GA to discover optimal communities in complex networks named MOGA-Net i.e. multiobjective GA basically a Non-dominated Sorting GA (NSGA-II). The algorithm used modularity criteria as objective function and Normalized Mutual Information (NMI) for measuring the performance of algorithm. The algorithm produced promising results in implementation for synthetic as well as real world data set. The authors (Amiri et al. 2013) proposed a multicriteria optimization approach that utilizes EFF algorithm Fuzzy- based grouping and mutation techniques for the detection of network communities. The implementation and testing of EFA on real world and other synthetic data networks showed its efficiency in finding different communities in large networks. The authors (Shang et al. 2013) proposed a community detection approach named MIGA on the basis of modularity function and an improved GA. The authors also used prior information regarding the number of detecting communities. The results shown a lesser computational complexity of MIGA in comparison to memetic algorithm (ME) and GA for both real-world and computer-generated data networks. The authors (Honghao et al. 2013) proposed an ACO method for detecting communities in networks using max-min ant system method for community detection. The algorithm was tested on four real-life network and LFR bench mark and results showed the great potential of algorithm in finding communities in networks. The authors (Hafez et al. 2014) used Artificial bee colony (ABC) optimization procedure to solve the community detection problem. The algorithm has advantage of automatically detecting the count of communities. The algorithm shown efficient results in terms of accuracy and detection of communities when applied on real-world data as well as online social network. The authors (Ma et al. 2014) proposed fast multi-level memetic algorithm for community detection named MLCD that uses GA with multi-level learning strategies. The results showed a lesser computation time in comparison with original memetic algorithm.

Various evolutionary approaches aforementioned have been applied in the field of community detection for identification of network communities. There is still a gap between the results obtained by existing evolutionary approaches for solving community detection in comparison to original ones. The current work focuses on development of a novel nature-inspired algorithmic approach based on Ant Lion Optimizer (Mirjalili 2015) that is aimed at maximizing the benefit function modularity to produce good partitions of a network into different set of communities through exploration and exploitation, by searching through the possible candidates for ones with high modularity.

2 Problem Statement

Given a graph $G(V, E)$ with a set of n nodes/vertices $V = \{\mathrm{v1}, \mathrm{v2}, \ldots..\mathrm{vn}\}$ and a set of m interconnections/links $E = \{\mathrm{e1}, \mathrm{e2}, \ldots.\mathrm{e}m\}$, the graph reflecting the social structure corresponding to the given community is represented with an adjacency matrix A of size $V \times V$ such that for any pair of vertices i and j

$$A_{i,j} = \begin{cases} 1 & \text{if an edge between i and j} \\ 0 & \text{Otherwise} \end{cases} \tag{1}$$

Where the adjacency matrix A is symmetric as shown in Eq. (1) (Assuming the graph as undirected graph).

The problem of community detection relies on finding the subgraphs i.e. partitioning the graph G into n subgraphs $G_1, G_2, \ldots\ldots..G_n$ and $V = G_1 \cup G_2 \cup \ldots\ldots..G_n$ such that all $G_i \forall i \in n$ correspond to the communities of densely linked nodes, with the nodes belonging to different communities being only sparingly connected based on the criteria of optimizing modularity value. The modularity function evaluates the quality of cluster signifying the extent to which a given community partition is distinguished by high number of intra-community connectivity in comparison to inter-community ones (Newman and Girvan 2004).

The Modularity (Q) value can be mathematically stated as in Eq. (2)

$$Q = 1/2\mathrm{m} \sum\nolimits_{i,j} \left(A_{ij} - \frac{d_i d_j}{2\mathrm{m}} \right) \delta(\mathrm{i}, \mathrm{j}) \tag{2}$$

Where, $A_{i,j}$ is the adjacency matrix, m is the number of edges in network, d_i, d_j are the degree (or strength) of nodes i, j and $\delta(i, j)$ is the function which return 1 when both i, j are in same community, 0 otherwise. The modularity value of a community ranges from -1 to 1 that computes the degree of cohesiveness within community as compared to interconnections between communities (Newman 2004; Pizzuti 2012). More the modularity value better is the quality of the communities detected.

3 The Proposed Algorithm

The work presents a novel nature-inspired algorithmic approach based on Ant Lion Optimizer for solving the Community Detection named ALOCD approach. The underlying algorithm proposed by Seyedali Mirjalili (Mirjalili 2015) utilizes the unique hunting behavior of antlion. The algorithm is inspired by of hunting behavior of victim such as random walk of ants, constructing traps, entrapment of ants inside traps, catching preys, and re- constructing traps. These features allow the antlions to take the positions of ants, making the antlions move towards better fit ants to achieve optimal solution. The main components of the proposed algorithm are as follows:

3.1 Solution Representation

In the proposed approach each solution/antlion in the population is encoded as a collection of n antlion positions, $s = [a_1, a_2, \ldots.a_n]$ such that each value $a_i \in [1, n]$ interprets the community to which the ith node belongs. The node i and node j belong

Communities / Ant Positions	2	1	3	1	2	3	3	1	2	1
	1	2	3	4	5	6	7	8	9	10

Fig. 1. Solution representation

to same cluster if the value a_i equals a_j for any set of i and j nodes. One antlion stands for one solution that divides given community structure into sub-community partitions. Figure 1 illustrates a solution representation of a social network with 10 nodes using array data structure with set of nodes {2,4,8,10} belonging to the 1st sub-community/cluster and set {1,5,9}, {3,6,7} belonging to 2nd and 3rd sub-community respectively.

3.2 Random Walk of Ants

In each iteration, the position of each ant is updated with respect to elite (best antlion obtained so far) and a selected antlion based on roulette wheel selection operator. This updation of position is performed with the help of two random walks i.e. random walk on the basis of roulette selected antlion and elite antlion.

For a given community of size say ‘n’, pick value at random position [1:n] from candidate solution(elite/roulette selection) which represents sub-community number to which that indexed vertex belongs to. Figure 2 shows the representation of Candidate Solution before random walk.

Communities / Ant Positions	1	3	2	1	2	1	3	1	2	3
	1	2	3	4	5	6	7	8	9	10

Fig. 2. Candidate solution (before random walk)

Figure 3 shows one step in random walk procedure by taking a random position say 4th in candidate solution and generates new solution after merging 1st and 3rd communities. If the newly generated solution gives better modularity than candidate solution, the candidate solution is updated. This step of random walk is applied recursively depending upon the size of the trap. For assuring exploitation of search space, the radius of updating ant’s positions is contracted. This step is modeled by decreasing the random walk rate as the iteration value approaches Num_of_gen value.

Communities / Ant Positions	1	1	2	1	2	1	1	1	2	1
	1	2	3	4	5	6	7	8	9	10

Fig. 3. Solution after merging 1^{st} and 3^{rd} communities

3.3 Updating the Position of Ants

During each iteration of the algorithm, the movements of all the ants are affected by elite as well as selected antlion as every ant randomly walks around a selected antlion by the roulette wheel and the elite concurrently. This concurrent effect of both selected and elite antlion on the movement of ant is modeled using Update_Ant_pos() procedure as shown in Fig. 4. The steps shown in Fig. 4 are repeated for all the possible communities while retaining better solutions in every repetition.

The ALOCD approach is illustrated by the pseudo code as shown in Table 1. A solution is encoded as a random permutation of n positions of ant/antlion representing the communities of given input net list file pertaining to the social network. The initial solutions are preprocessed by randomly picking a vertex Vi and finding its adjacent vertex say Vj. The community value of Vi is allocated to that of Vj. This process is repeated for rest of vertices until sufficient number of solutions for initial population are generated. After initialization, the best antlion (elite) is chosen from the generated population on the basis of modularity value of antlion. More modularity

Communities / Ant Positions	1	3	2	1	2	1	3	1	2	3
	1	2	3	4	5	6	7	8	9	10

Solution1 (From Random walk around elite)

Communities / Ant Positions	3	1	3	1	1	1	3	2	2	1
	1	2	3	4	5	6	7	8	9	10

Solution2 (From Random walk around selected antlion)

(Assuming Solution1 has better modularity than Solution2)
Vertices belonging to 1st community in Solution1 {1,4,6,8}
Vertices belonging to 1st community in Solution2 {2,4,5,6,10}
X= Vertices_Solution1 U Vertices_solution2 {1,2,3,4,5,6,8,10}
C= X-Solution1 = { 2,3,5,10}
Modified Solution1 using C value

Communities / Ant Positions	1	1	1	1	1	1	3	1	2	1
	1	2	3	4	5	6	7	8	9	10

Modified_ Solution1
If Modified_ Solution1 _Modularity > Solution1 _Modularity
Then
Solution1= Modified_ Solution1

Fig. 4. One step of *Update_Ant_pos()* procedure

Table 1. Pseudocode of proposed ALOCD algorithm

```
    Community_AntLionDetector (Input_File, Num_Nodes, Num_nets, Numclass , class,
Pop_size, Num_of_gen)

    Input : Read the Benchmark files of communities
    Variables: pop_ant, pop_antlion - Array of structures of solution of size (Pop_size X
    Num_Nodes), Merge_solution -Array of structures of solution of size ( (2 X Pop_size)
    X Num_Nodes)
    Class- True community of community structure, Numclass -Total no .of true communi-
    ties
    Output: Set of Best Communiites

    Begin
    [adj_array]=Create Netlist(Input_File, Num_Nodes, Num_nets)
    [pop_ant]=Initalise(Pop_size,Num_Nodes,adj_array)
    [pop_antlion]=Initalise(Pop_size ,Num_Nodes,,adj_array)
    Set iteration:=1
     While (iteration <= Num_of_gen)
               [Elite]=Findbest(pop_antlion,Pop_size,Num_Nodes)
                 For every ant  i=1:Pop_size
                    For j=1:Pop_size
                                 w(j)=pop_antlion(j).fit
                        EndFor
                          choice = RouletteWheelSelection(1./w,Pop_size)
                    If choice==-1
                                 choice=1
                          Endif
                    roulete_antlion=pop_antlion(choice)
                    RA = Random_walk (Elite,Num_Nodes,adj_array)
                           RB = Random_walk (roulete_antlion,Num_Nodes,adj_array)
                           Cross_R= Update_Ant_pos(adj_array,RA,RB,Num_Nodes)
                           pop_ant(i)=Cross_R
                           If pop_ant(i).fit>Elite.fit
                                 Elite= pop_ant(i)
                           EndIf
             EndFor
             Merge_solution =Merge_Sort(pop_ant,pop_antlion, Pop_size,Num_Nodes)
             [pop_antlion ]=Update_solution(Merge_solution, Pop_size,Num_Nodes)
     Endwhile
    [C]= pop_antlion(1).bit
    [NMI] = Compute_NMI(class, C)
    Print: 'nmi ', NMI
    Print: 'Optimized Community ', pop_antlion(1).bit
    End
```

contributes to better antlion. The quality of solutions/antlions is improved through random walk. After random walk, if the new ants have high fitness than antlions, then antlion new positions will become positions of the ants for imitating the process of catching the prey, the antlion is required to change its position to the most recent position of the hunted ant to boost its chance of catching new prey. In each iteration, the antlion with highest fitness is substituted as best antlion (elite). The process is repeated until optimal solution is obtained. At final stage, the NMI value of the first solution of the population with best modularity value is calculated.

4 Simulation Results

The proposed ALOCD approach is implemented using matlab 7.11.0 (R2012a) on intel core i5 processor, with 4 GB RAM under 64-bit Operating System. The work is tested on Zachary's Karate Club, Bottlenose Dolphins, American college football and Books about US politics network (Newman 2009) benchmarks and results are compared with the ACO and EFF (Amiri et al. 2013; Honghao et al. 2013). The characteristics of these benchmarks are shown in Table 2.

Table 2. Shows the basic features of the real world networks and their true number of community structures

Benchmark networks	Nodes	Edges	True communities
Zachary's Karate Club	34	78	2,4
Bottlenose Dolphins	62	159	2,4
Books about US politics	105	441	3
American College Football	115	613	12

The performance of ALOCD approach is evaluated using Normalized mutual information (NMI) (Le Martelot and Hankin 2013; Pizzuti 2008; Pizzuti 2012; Amiri et al. 2013) that quantifies the similarity between the detected and true community structure. NMI denoted as I (X,Y) is calculated using the following formula as shown in Eq. (3).

$$I(X,Y) = -\frac{2\sum_{i=1}^{C_X}\sum_{j=1}^{C_Y} C_{ij}\log\left(\frac{C_{ij}N}{C_i}\cdot C_j\right)}{\sum_{i=1}^{C_X} C_i\cdot\log(C_i/N)+\sum_{j=1}^{C_Y} C_{.j}\log(C_{.j}/N)} \tag{3}$$

Where X and Y denotes two network structures, C - the confusion matrix; C_{ij} - the count of nodes present in community i of X as well as in community j of Y; C_X, C_Y - the number of classes in part X and Y; $C_i, C_{.j}$ - the count of elements in row i and column j of C; N - the total count of nodes in networks.

The larger value of NMI reflects more similarity between true and detected communities leading to better solution quality. For both communities being same, the NMI value equals 1 and NMI = 0 signifies different communities.

Table 3 shows the results of implementation of the proposed approach over 10 different runs for a given set of benchmark networks. The algorithm computes the NMI value and determines the total number of communities in each run, as shown in Table 3.

From the tabulated values as given in Table 3, it is concluded that the proposed approach is competent to identify 100% community structure for Zachary's karate network (Newman 2009). Figure 5 reveals NMI value equal to 1 at 5th and 8th run of the program execution with the number of communities same as that of true community structure for Zachary's karate network.

Figures 6, 7 and 8 shows the NMI values of ten runs of ALOCD on Bottlenose Dolphins, Books about US politics and American Football Club benchmark networks

Table 3. Represents the NMI values and number of communities for multiple runs of ALOCD approach

Benchmark networks		Number of runs									
		1st	2nd	3rd	4th	5th	6th	7th	8th	9th	10th
Zachary's Karate Club	NMI	0.732	0.732	0.592	0.494	**1.000**	0.732	0.516	1.000	0.732	0.732
	NC	2	2	4	4	**2**	2	8	2	2	2
Bottlenose Dolphins	NMI	0.502	0.534	0.699	0.798	0.698	0.835	**0.887**	0.802	0.513	0.756
	NC	5	3	2	3	4	2	**2**	3	6	4
Books about US politics	NMI	0.696	0.508	0.449	0.506	0.726	**0.733**	0.431	0.668	0.705	0.534
	NC	4	3	10	15	4	**3**	7	5	3	9
American College Football	NMI	0.537	0.608	0.503	0.804	0.775	0.795	0.665	**0.850**	0.611	0.759
	NC	17	16	14	13	12	13	22	**12**	20	14

NC-represents the number of Communities and bold faced values represent best results of ALOCD approach for respective benchmark networks.

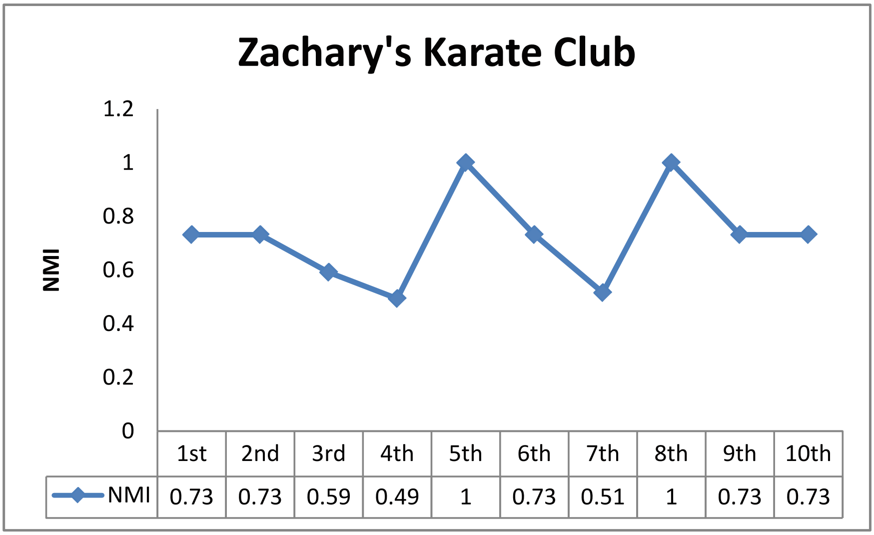

Fig. 5. NMI values of ten runs of ALOCD on Zachary's Karate Club

respectively. In 7th run of the implementation of ALOCD approach, the no. of communities found for bottlenose dolphins is equal to true number of community structures with NMI value of 0.887 as shown in Fig. 6.

From Fig. 7 it is clear that the number of communities found for Books about US politics is equal to the number of true communities with NMI value of 0.733 at 6th run of program execution. Figure 8 depicts same number of communities as that of actual one at 8th run of implementation for American Football with NMI value of 0.850.

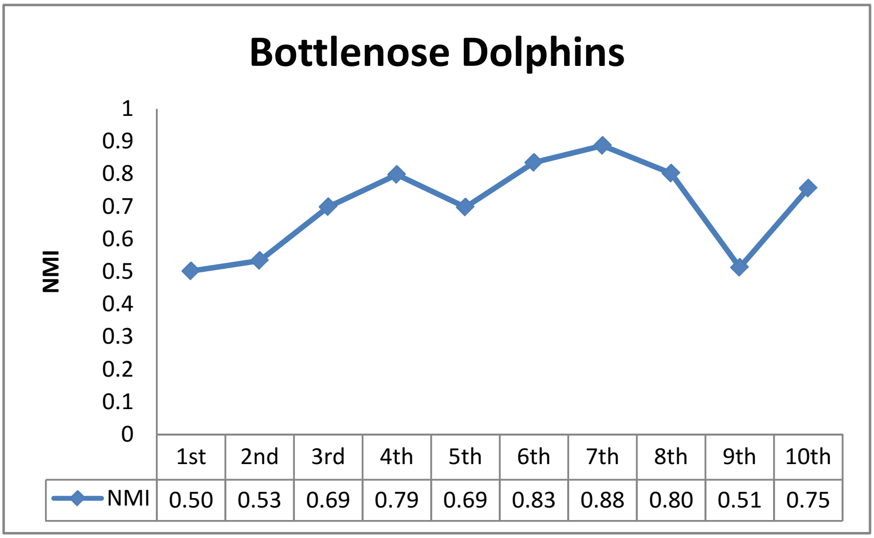

Fig. 6. NMI values of ten runs of ALOCD on Bottlenose Dolphins

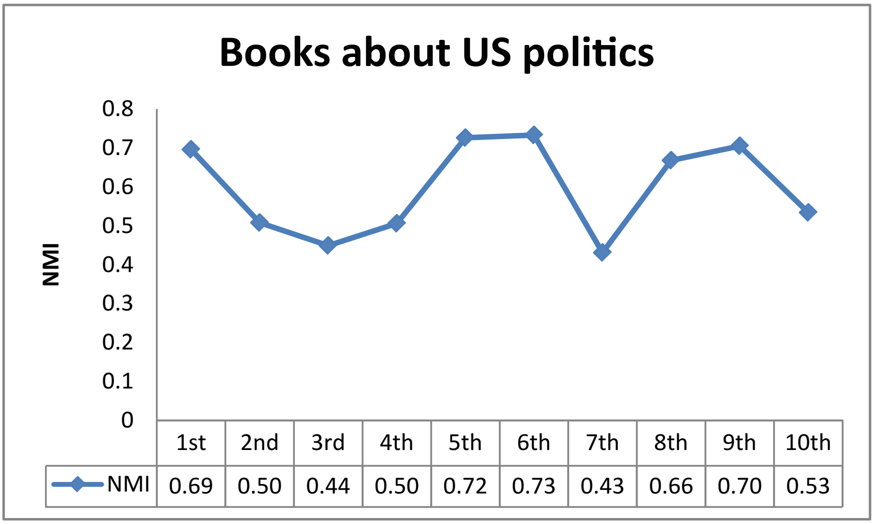

Fig. 7. NMI values of ten runs of ALOCD on books about US politics

Table 4 depicts excellent performance of ALOCD approach for Zachary's Karate Club Benchmark Networks. The solutions found by ALOCD approach split the communities into 2, 3, 12 clusters with 1, 6, 13 nodes misplaced for Bottlenose Dolphins, Books about US politics, American College Football respectively.

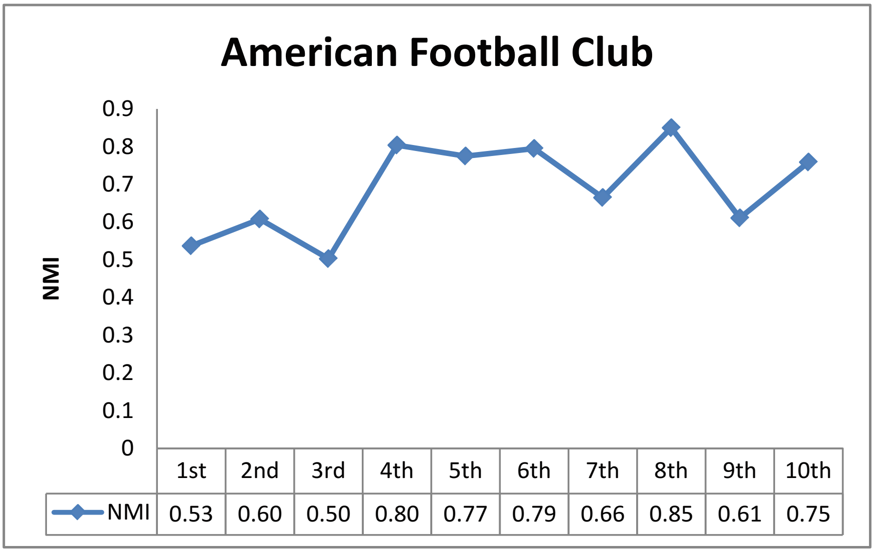

Fig. 8. NMI values of ten runs of ALOCD on American Football Club

Table 4. Shows the count and nodes of wrong communities along with best NMI of all networks

Benchmark networks	NMI	Wrong communities	Nodes in wrong communities
Zachary's Karate Club	1.000	–	–
Bottlenose Dolphins	0.887	1	[20]
Books about US politics	0.733	6	[8, 15, 28, 46, 54, 89]
American College Football	0.850	13	[10, 18, 23, 32, 39, 47, 52, 54,65, 78, 82, 86, 101]

Table 5. Shows the comparison of best NMI values of ALOCD, ACO and EFF algorithm and also shows at best NMI the respective communities found in all networks

Benchmark networks	NMI and no. of communities	ALOCD	ACO	EFF
Zachary's Karate Club	NMI	1.000	0.687	0.998
	Communities	2	2	4
Bottlenose Dolphins	NMI	0.887	0.587	0.988
	Communities	2	2	4
Books about US politics	NMI	0.733	0.560	0.599
	Communities	3	2	4
American College Football	NMI	0.850	0.890	0.798
	Communities	12	12	11

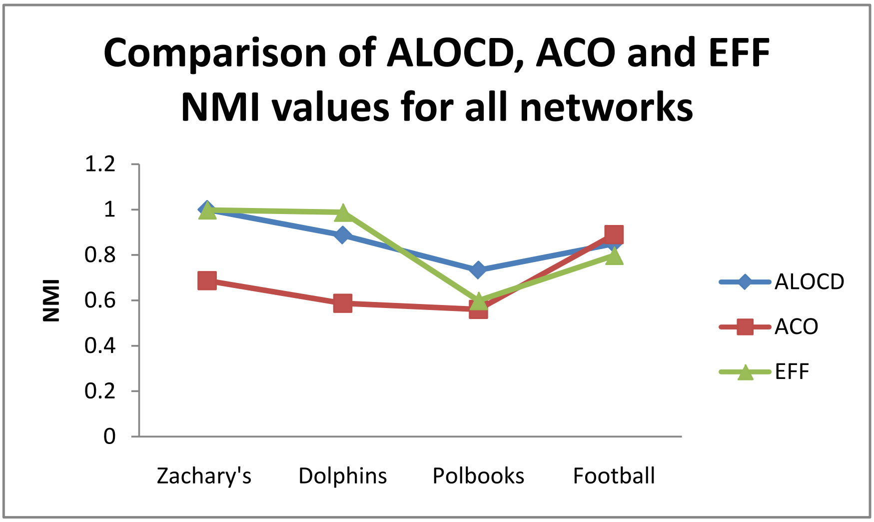

Fig. 9. Comparison of best NMI values of ALOCD with ACO and EFF algorithm's NMI Values for four real world networks

The results of the proposed approach are compared with ACO and EFF in terms of best NMI values obtained. From Table 5 and Fig. 9, it is concluded that the proposed ALOCD approach detects 100% true community structure statics for Zachary's karate network. For Dolphins networks, the algorithm obtained best normalized mutual information of 0.887 closer to EFF best NMI value, while the NMI of ACO was 0.798. For Books about US politics, the proposed approach has highest NMI value of 0.733 in comparison with other approaches. On the American College Football network, ALOCD obtained best normalized mutual information of 0.850 closer to ACO's best NMI value, while the NMI of EFF was 0.798. Consequently the proposed approach is able to detect the true count of communities with NMI value close enough to NMI value of true community structure.

5 Conclusion

The proposed approach optimizes the modularity function and is able to recognize densely connected clusters of nodes bearing sparse interconnections. The performance of the algorithm is measured in terms of Normalized Mutual Information (NMI) function. The algorithm is tested on real-world networks i.e. Bottlenose Dolphins, Zachary's Karate Club, Books about US politics and American Football Club. The experimental results show 100% community structure statics detection for Zachary's Karate club. The approach also gives promising results for other networks by finding the true number of communities with NMI values close to true community structures. The ALOCD approach is also compared with ACO and EFF over given set of benchmarks. The approach outperforms EFF and ACO for Zachary and Books about

US politics and produces results better than ACO and comparable to EFF for Dolphins and also produces results better than EFF and comparable to ACO for American Football Club. The work can be further extended for other bigger real world networks like Facebook, Twitter etc. This nature inspired approach can be improved to solve the problem of community detection for overlapping communities and dynamic community detection.

References

Amiri, B., Hossain, L., Crawford, J.W., Wigand, R.T.: Community detection in complex networks: multi-objective enhanced firefly algorithm. Knowl.-Based Syst. **46**, 1–11 (2013)

Clauset, A., Newman, M.E., Moore, C.: Finding community structure in very large networks. Phys. Rev. E **70**(6), 1–6 (2004)

Ferrara, E., Fiumara, G.: Topological features of online social networks. Commun. Appl. Indus. Math. **2**(2), 1–20 (2011)

Fortunato, S.: Community detection in graphs. Phys. Rep. **486**(3), 75–174 (2010)

Hafez, A.I., Zawbaa, H.M., Hassanien, A.E., Fahmy, A.A.: Networks community detection using artificial bee colony swarm optimization. In: Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications, IBICA 2014, pp. 229–239 (2014)

Honghao, C., Zuren, F., Zhigang, R.: Community detection using ant colony optimization. In: IEEE Congress on Evolutionary Computation (CEC), pp. 3072–3078 (2013)

Lambiotte, R., Lefebvre, E., Guillaume, J.L., Blondel, V.D.: Fast unfolding of communities in large networks. J. Stat. Mech: Theory Exp. **2008**(10), P10008 (2008)

Le Martelot, E., Hankin, C.: Fast multi-scale detection of relevant communities in large-scale networks. Comput. J. **56**(9), 1136–1150 (2013)

Ma, L., Gong, M., Liu, J., Cai, Q., Jiao, L.: Multi-level learning based memetic algorithm for community detection. Appl. Soft Comput. **19**, 121–133 (2014)

Mirjalili, S.: The Ant lion optimizer. Adv. Eng. Softw. **83**, 80–98 (2015)

Newman, M.E.: The structure and function of complex networks. SIAM Rev. **45**(2), 167–256 (2003)

Newman, M.E.: Fast algorithm for detecting community structure in networks. Phys. Rev. E **69** (6), 1–5 (2004)

Newman, M.E.: Finding community structure in networks using the eigenvectors of matrices. Phys. Rev. E **74**(3), 036104 (2006)

Newman, M.E.: (2009). http://www-personal.umich.edu/mejn/netdata/

Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**(2), 1–16 (2004)

Pizzuti, C.: GA-Net: a genetic algorithm for community detection in social networks. In: Rudolph, G., Jansen, T., Beume, N., Lucas, S., Poloni, C. (eds.) PPSN 2008. LNCS, vol. 5199, pp. 1081–1090. Springer, Heidelberg (2008). doi:10.1007/978-3-540-87700-4_107

Pizzuti, C.: A multiobjective genetic algorithm to find communities in complex networks. IEEE Trans. Evol. Comput. **16**(3), 418–430 (2012)

Schuetz, P., Caflisch, A.: Efficient modularity optimization by multistep greedy algorithm and vertex mover refinement. Phys. Rev. E **77**(4), 046112 (2008)

Shang, R., Bai, J., Jiao, L., Jin, C.: Community detection based on modularity and an improved genetic algorithm. Physica A Stat. Mech. Appl. **392**(5), 1215–1231 (2013)