2019 5th International Conference on Web Research (ICWR)

Memetic Based Online Community Detection

Mohammad Foad Abdi
Department of Mathematics and Computer Science
Amirkabir University of Technology
Tehran, Iran
mohamadfoadabdi@aut.ac.ir

Kasra Farrokhi
Computer and Information Technology Engineering Department
Amirkabir University of Technology
Tehran, Iran
farrokhi.Kasra@aut.ac.ir

Maryam Amir Haeri
Computer and Information Technology Engineering Department
Amirkabir University of Technology
Tehran, Iran
haeri@aut.ac.ir

***Abstract*— Community detection is one of the most important tasks in social networks analysis. This problem becomes more challenging when the structure of the network changes during the time. It is very important to update the structures of the community in a dynamic network without time-consuming procedures. This paper suggests a hybrid evolutionary algorithm for online community detection. The proposed algorithm called Memetic Based Online Community Detection (MBOC) is based on a memetic algorithm with new genetic operators and a novel stochastic local search to assign new nodes to communities and another local search called dense search to modify communities after new assignments. The method is evaluated over several well-known benchmark networks. The results show that the proposed approach outperforms the previous methods in most cases.**

Keywords*— *Online community detection; Social network; Memetic algorithm; Density Preserving Crossover; Dense search mutatio; Stochastic local search.

I. INTRODUCTION

Today, social networks are everywhere they get people closer to each other, inform them about news and it is also used for commercials. These are just a few parts of social networks benefits. Social networks analysis can lead to extract a huge amount of knowledge. Analyzing social networks are mostly based on graph processing. One of the most well-known tasks in social networks analysis is community detection. Community detection is used to find the relation among the nodes and find the highly connected nodes. The information about the community can be used in many applications. For example, some social networks use them to suggest the people you might know to add them to your friend list or detect groups that are more connected and has a common attribute between them for recommendations. We can analyze social interactions that have been happening like posts and comments, tweets, short messages, phone calls and etc., which are produced during the time streaming networks, these network's communities will change during the time steps. Besides that, some of the online shopping companies use community detection in their recommendation systems to find a set of customers that have the same appetite for buying things to propose their purchase to another.

In most social networks, the network changes during the time. Since the community detection process is time-consuming, it is not possible to re-perform the community detection algorithm by each change occurs in the network. So, it is very important to find new generated communities in an online manner. Moreover, as the network data is very huge and the memory is limited the online community detection methods should work with a limited memory and has low space complexity. The changes in a social network can be occurred by following or unfollowing users or creating new accounts or deleting existing accounts. These occurrences change the structure of communities. Hence, the communities must be updated and for this, dynamic community detection algorithms are needed.

Community detection is an optimization problem. Here, the objective function can be any goodness features of communities such as maximizing modularity, minimizing ratio-cut or maximizing conductance. Thus, evolutionary algorithms such as genetic or memetic algorithms can be used in community detection. Many researchers utilized the evolutionary algorithms for clustering problems. Their results indicated that evolutionary algorithms are appropriate methods for these purposes such as finding communities [9]. There are some researches related to the online community detection (or dynamic community detection) which used evolutionary computations. However, the results of some of them such as Pizzuti's research [15] showed that evolutionary methods have the potential to become as the most appropriate methods for finding communities in dynamic networks.

In this paper, a new evolutionary community detection algorithm for streaming dynamic networks is proposed. The method assumes that the edges or nodes of the graph can be removed or added. This means that the structure of graphs such as nodes and edges can be changed, added or deleted and this changes the Communities. Hence, the method has to update the communities in each time steps, in an efficient manner. There are different parameters to measure community detection performance.

The structure of the rest of this paper is as follows: First, The previous related works are reviewed in Section 2. Section 3 explains the proposed method called MBOC and Section 4 is devoted to evaluating the proposed method. Finlay, Section 5 concludes the paper.

II. RELATED WORKS

Community detection is an important problem and has many applications in different areas, such as physics, sociology, biology, and computer science. There are many approaches for community detection in the literature. However, most of them are related to the community detection in static networks. These methods are based on linear algebra, data clustering, evolutionary algorithms and so on.

For example, Lin et al. [12] presented a community detection method with non-negative matrix factorization called FacetNet. Tian et al. [19] suggested a new community detection strategy to split the graph into communities by using the similarity between the attributes of nodes. Another

978-1-7281-1431-6/19/$31.00 ©2019 IEEE

method was suggested by Radicchi et al. [16] which uses the in-degree and out-degree notions to split communities to weak and strong groups and use this definition to cluster the graph nodes. Moreover, Cascia et al. [4] used the label propagation method to detect communities from ego-networks. Some other methods such as CFINDER [14] try to find cliques as a base for network structures and community detection.

Network clustering can be defined as an optimization problem [13] and this optimization problem can be solved by evolutionary methods. For example, Pizzuti [15] developed an approach based on genetic algorithm for network clustering and Gong et al. [8] proposed a memetic algorithm for network community detection. They used the memetic algorithm to maximize modularity for static social networks. The Crossover function used in their method has been inspired by [5] and an improved version of one-way crossover function called two-way crossover, in order to reach to graph portioning with higher modularity. At last, one kind of local search algorithm like hill climbing search is used in order to improve the communities. This works showed that memetic algorithm can be a proper solution for community detection in static networks. Thus, in this paper, we try to develop a memetic based method for dynamic community detection in online social networks.

When the structure of the complex (social) network changes dynamically, we need efficient methods for community detection. Such methods should change the structure of the communities in a fast and efficient manner.

For online community detection, some solutions have been proposed, recently.

Kim and Han [10] are one of the first computer scientists who introduced dynamic social networks and proposed a density-based clustering method based on the evolutionary algorithms in dynamic networks.

Moreover, Alvari et al. [1, 2] used a dynamic game theoretic approach for dynamic community detection. Wang et al. [19] developed an online approach for community detection which stores the nodes and their previous communities to consider the new changes in the graph and in the communities. Thus, they can find the community changes by utilizing a huge amount of storage. Rosseti et al. [17] presented an algorithm that produces overlapping communities in streaming graphs. *Intrinsic Longitudinal Community Detection* [3] is an approach for dynamic graphs that re-evaluate new communities at each iteration generated by LFR (Lancichinetti–Fortunato–Radicchi) [11] streaming source. LFR is a synthetic graph generator that used to evaluate community detection approaches. Gong et al. [7] method uses label Propagation and simulated annealing and elitism strategy to find communities with higher normalized mutual information (NMI) in dynamic social networks.

Evolutionary computations are also used for online community detection. For example, Folino et al. [6] suggested DYNMOGA (Dynamic Multi-Objective Genetic Algorithm) for community detection in dynamic social networks. They attempted to achieve temporal smoothness by multi-objective optimization, i.e. maximization of snapshot quality (community score is used) and minimization of temporal cost.

In this paper, we aim to introduce a new evolutionary method for online community detection that updates the communities very fast and accurate during the time steps.

III. Memetic Based Online Community Detection

The proposed approach for detecting communities is based on density. Modularity is a measure that has been defined to determine density and the strength of a division of a network. High modularity represents that connections between nodes belonging to the same community are dense and connections between nodes belonging to different communities are sparse. For each graph G, with n nodes and m edges modularity is calculated by the following equation:

$$Q(G,S)=\frac{1}{2m}\sum_{s\in S}\sum_{i\in s}\sum_{j\in s}(A_{ij}-\frac{k_i k_j}{2m}) \qquad (1)$$

In this formula, $\frac{k_i k_j}{2m}$ represents the expected number of connections between nodes v_i and v_j; where k_i and k_j are the degree of nodes v_i and v_j, respectively. Hence, for each group or community of nodes $\sum_{i\in s}\sum_{j\in s}(A_{ij}-\frac{k_i k_j}{2m})$ is the strength of the relation and density between nodes and if a network is partitioned into s communities, modularity of network is $\sum_{s\in S}\sum_{i\in s}\sum_{j\in s}(A_{ij}-\frac{k_i k_j}{2m})$ where $\frac{1}{2m}$ is defined to normalize the formula in the interval of [-1,1].

In this paper, this must be considered that the online (dynamic) community detection problem is more challenging than the offline one, as the network is changing at each time step. So, the structures of communities are changing during the time. In these networks, the edges and nodes are going to be added, deleted or altered during the time. It is important to find the community structures at each time step without computing everything from the beginnings because community detection of a huge graph could be time-consuming. Thus, in the online community detection, the structures of the communities in the previous step should be updated by the changes in the current time step.

Here, we represent a network at time step t, by G_t. Each graph consists of some communities. Each community is a partition of graph, let $CR_t=\{C_1,C_2,C_3,\ldots,C_k\}$ be the set of all communities of G_t; where, C_i is the ith partition of the network. Clearly, the union of vertices of all communities in time step t is the set of vertices of G_t ($\cup C_i=V$). Also for each different i, j ($i\neq j$) we have $C_i\cap C_j=\phi$, and each network G_t is the snapshot of the network at time step t and its change for next time step i is ΔG_i that would be some communities that are created, removed, grew or being small at each time step. In order to have an efficient method for community detection in dynamic networks, at the first time step the communities are found by a community detection method, after that the network is observed for each change in any part of its structure and by each change, the community structures are updated. It is important that each change in the network do not affect all parts of it and only affects locally. Hence, each change may only affect a few numbers of communities and thus, it is not needed to compute everything.

Evolutionary algorithms and particularly the memetic algorithm is a method for achieving the near-optimal results during the evolution process. What we have in the memetic algorithm is a population and several genetic operators for generating new chromosomes from every one or two chromosomes of the population. For example, consider the chromosomes A_1, A_2, …, A_n every two chromosomes are selected and processed by the method and generate better chromosomes B_1, B_2, …, B_n. This algorithm could round in m times to produce desirable results.

Our presented solution for online community detection is based on, memetic algorithm. This method helps us cluster the graph for each time step T based on the last community structure in $T-1$. A chromosome in memetic algorithms is a set of parameters that determine the answer of the algorithm. In this paper, each plan (solution) of clustering the graph represents as a chromosome P_i of the population. Each chromosome of this algorithm is represented by a $1\times V$ matrix, where V is the number of nodes and each row (element) i of this matrix is the community number of ith node.

A. Population Initialization

At the first step of community detection, an initialization of population is needed to run the memetic algorithm and make the result more and more accurate, during the evolution. In our approach, the initial population is generated randomly. To generate each chromosome of the initial population, the following procedure should be done:

Similar to [18], a threshold α is considered by this threshold at the first step at most we have $\alpha\times V$ communities. Then $\alpha\times V$ nodes are randomly selected from the network. Then each of these selected nodes is considered as the seed of a cluster. After that, other nodes of the graph are assigned to the cluster of their neighbor. If a node is not connected to any of the seed nodes, it assigned to a random cluster. Thus, each chromosome of the algorithm is a vector of size V which one of its elements v_i is the cluster number assigned to the node i. In the experiments of this paper, α is set to 0.2. The pseudo-code of the initialization process is demonstrated in Fig. 1.

B. Memetic Operators for Online Community Detection

Memetic algorithms mostly have three main genetic operators, the crossover, mutation and local search. Crossover method takes two chromosomes and swaps the parts of the chromosomes by each other to generate one or

Algorithm 1 Population initialization Method

Generate population P of size S_{pop} , each chromosome X has size V and generated as follows:
for all chromosome $X = \{x_1, x_2, \ldots, x_V\}$ in P **do**
  Randomly select seed vertex set V_s of size $\alpha \times V$
  for all $v_i \in V_s$ **do**
    for all vertex u that $(u, v_i) \in E$ **do**
      $x_u = x_v$
    end for
  end for
end for

Fig. 1. Pseudo-code of population initialization in MBOC

individual and change a part of it randomly to a generated new chromosome.

Another function that could lead to better exploitation of the search space is local search. Local search is a procedure that checks near states of the current solutions (chromosomes) to find the most accurate solution. In the case of the community detection, it checks every node or community of the graph to enhance the results, by some local changes in the communities.

1) Density Preserving Crossover

Since in MBOC algorithm the objective is finding high-density communities, the genetic operators work based on density.

The new crossover considers two community partitioning solutions (chromosomes) and generates two other community partitioning solutions (offsprings) each of which contains the densest communities of both parents.

Consider the two parent chromosomes X_a and X_b. At the first step, the densest communities in X_a and X_b are found. We represent them as D_a and D_b respectively. Then considering the D_a and X_b it is possible to generate an offspring X'_b in which all the community assignments are the same as X_b except that all the nodes of D_a are assigned to the same community. This means that the densest community of X_a is copied in X_b and generates offspring X'_b. In the same manner, a new offspring X'_a is generated by copying the densest community D_b in X_a.

An illustrative example of the proposed crossover is depicted in Fig. 2.

2) Dense Search Mutation

Mutation helps the exploitation of the search space and they can be considered as a local search. Here, the mutation operator provides a local search called dense search. In the mutation function of MBOC, a node v_i with probability P_{mutex} is randomly selected. Then for each neighbors v_j of v_i which is located in different community than v_i; the community of node v_i is assigned to node v_j. After that, the modularity of the communities is recalculated. If the modularity is increased, the new assignment is accepted. This procedure is called dense search function.

If all the neighbors of v_i are in the same community with v_i, then the dense search function is performed on each neighbor of v_i, in the same manner.

The pseudo code of this process is shown in Fig 3. This function gets a chromosome X and a number C that is the C^{th} community of the chromosome X (C_C: C^{th} community) then for each adjacent node V'_i of C_C, it checks that if V'_i added to the community C_C, makes the modularity of graph much bigger than previous state. If this change leads to better modularity the community of V'_i is changed to C_C and by this change a new chromosome is generated.

3) Stochastic Local Search

Here we introduce a local search to find better communities. This local search is called stochastic local search. At first, this local search chooses several nodes randomly and puts them in the search list. Then For each

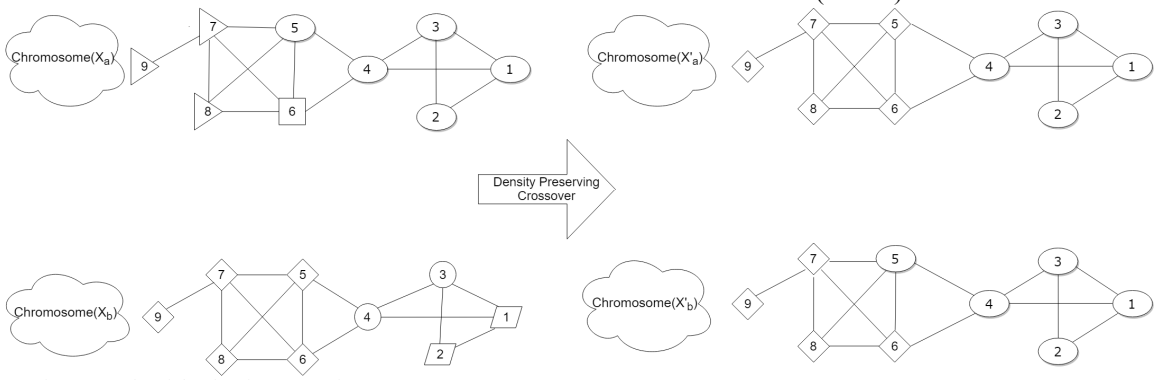

Fig. 2. Illustrative example of the density preserving crossover operator.

Algorithm 2 Pseudo-code of Dense Search(X, C)

```
MaxModularity = modularity(Graph, X)
for all node V'i adjacent to CC do
  if V'i ∈ CC then
    A new chromosome X-new is generated
    if modularity(Graph, X-new) > MaxModularity then
      X = X_new
      MaxModularity = modularity(Graph, Xnew)
    end if
  end if
end for
```

Fig. 3. Pseudo-code of dense search mutation in MBOC.

node v_i in the search list, the following operation is done:

Every adjacent community of v_i is determined. If any edge of v_i is linked to any node of the community C_j, the community C_j is called an adjacent community of v_i.

Here, a threshold $0 < \theta < 1$ is considered. Then, for any adjacent community C_j of v_i, the proportion ρ of the edges connected to C_j is calculated (ρ = number of edges between v_i and C_j divided by degree of v_i).

If ρ is greater than the θ then vertex v_i is assigned to the cluster C_j. Due to increasing of the density; If $\rho > \theta$, then C_j it is better to assign v_i to community C_j.

On the other hand, if $\rho < \theta$ then the local search assigns v_i to one of the adjacent communities that makes the graph more dense in terms of modularity. If this method could increase the total modularity of the graph, then this process will be checked for one of the adjacent nodes of vertex v_i. Similarly, in the next step, this procedure performed over one of the adjacent nodes of v_i, and If modularity does not change, the process will be stopped.

The stochastic local search will be executed R times to ensure that the algorithm performs perfectly.

The pseudo-code of stochastic local search is shown in Fig. 4.

IV. Experimental Results

This section is devoted to the evaluation of MBOC. The proposed method is compared with other recent approaches for online community detection. For evaluating the community detection algorithms, some evaluation measures are needed. For validating our method, modularity and

Algorithm 3 The pseudo-code of the stochastic local search.

```
input: V = v1, v2, ..., vn
for all i = 1, 2, 3, ..., R do
  r = random(1, n)
  Define an array A
  Add vr and its adjacent nodes to A
  for all vi nodes in A do
    for all adjacent communities(Cj) of vi do
      adj(vi) = number of nodes in Cj connected to vi
      if (ρ =(adj(vi) / degree(vi)) > θ) then
        community(vi) = Cj
        break
      end if
      new_ Modularity = Assign node vi to community
      Cj and calculate modularity;
      if new_ Modularity > Modularity then
        community(vi) = Cj
      end if
    end for
  end for
  if Modularity does not grow then
    break
  end if
end for
```

Fig. 4. Pseudo-code of the stochastic local search in MBOC

Normalized Mutual information (NMI) are selected to compare the results.

In order to better evaluation, as our approach is density (modularity) based we need to use some other measures to test the effectiveness of our approach; thus, in addition to modularity, the normalized mutual NMI is used.

Normalized mutual information or NMI is an information theory measure and is used for determining the quality of clustering. This measure needs the class labels of the instances and it is calculated as follows:

$$NMI(X;Y)=\frac{MI(X.Y)}{H(x)+H(Y)}$$

Where $MI(X,Y)$ is the mutual information that is the shared information between these two distributions and is calculated by $MI(X)=H(X)-H(X|Y)$. $H(X)$ is known as the entropy of *X (target class labels)* and calculated for the entire dataset and can be calculated prior to clustering, as it will not change depending on the clustering output. Moreover, $H(X|Y)$ is the entropy of community labels within each class.

978-1-7281-1431-6/19/$31.00 ©2019 IEEE

1) Datasets

In this paper, the following datasets are used to evaluate the method. Most of the research works such as [1] and [9] utilized synthetic datasets to evaluate online community detection methods. Several of the most well-known synthetics datasets are generated by [9] and called *Birth-Death, Merge-Split* and *Switch*. These datasets are explained in the following.

Birth-Death: Birth-death is a synthetic dataset that their communities appear and disappear in each time steps. So a community C can be a subset of another community D but after some time steps, it could introduce itself as an independent community and its first appearance in network. In another way, a community C can lose its independence and become a subset of a community D, when its core nodes are be empty. Usually, happen in more than one time steps.

We generate birth-death data set with 2000 nodes, 50 communities (or seeds) , maximum degree of all nodes is 150, average degree of nodes is 50, probability of switching communities is 0.1, birth events per each step is 2 and death events per each step is 4, death events per each step is 4 and the maximum modularity is about 0.4.

Merge-Split: Merge-split is a synthetic dataset that their community merges when their cores nodes are has overlap and split when core nodes of community spread in during of each time step. We generate our dataset with 2000 vertices with 50 communities (or seeds), in 10 time steps with average degree 50, the maximum degree of vertices is 150 , with 20% merges & splits in each step.

This means that if there are two different communities C_1 and C_2 at time step T they could join in a community C_m at time step T+1 and if there is a community C_s at time step T' it could be split into two separate communities $C_{1'}$ and $C_{2'}$ at time step T'+1. These merges and splits happen more often in this dataset.

Switch: Switch is synthetic data set that flips memberships between communities at each step. We generate data set with 2000 nodes, 50 communities , maximum degree of all nodes is 150, average of all nodes degree is 50, mixing parameter is 0.2 and probability of is 0.1.

2) Results

In this section, the results of MBOC in comparison with three recent methods [2], [6] and [10] are investigated. In DYNMOGA (Dynamic Multi-Objective Genetic Algorithm) [6], the authors have used a genetic algorithm based approach to dynamic community detection. They attempt to achieve temporal smoothness by multi-objective optimization, i.e. maximization of snapshot quality (community score is used) and minimization of temporal cost. Kim and Han [10] are one of the first computer scientists who introduced Dynamic Social networks and proposed a density-based clustering method based on the evolutionary algorithms in dynamic networks. Alvari et al. [2] used a dynamic game theoretic approach for solving this problem.

These comparisons are based on two measures modularity and NMI. Fig. 5. depicts the results. As the results show, the

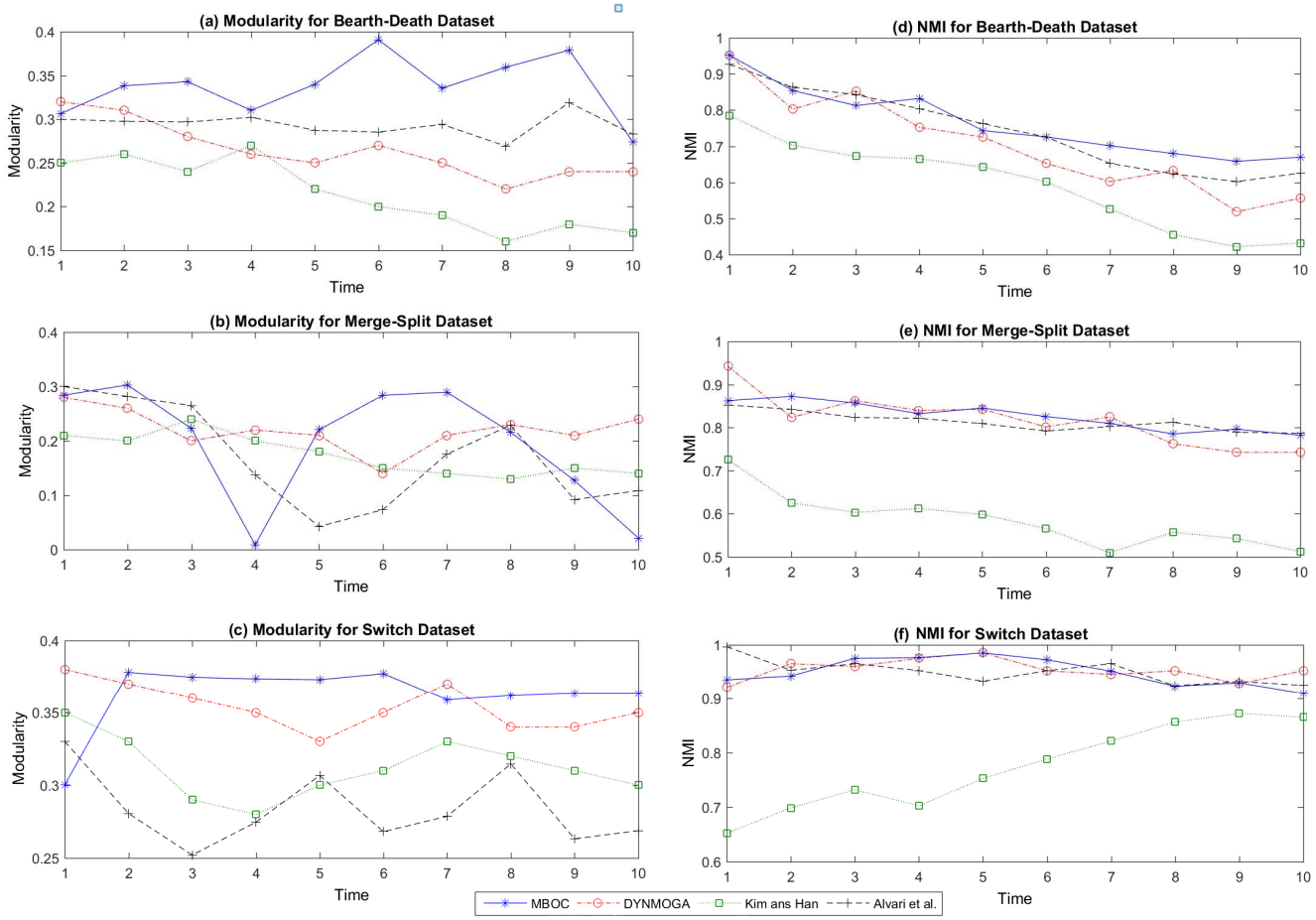

Fig. 5. The results of MBOC algorithm in comparison with three other online community detection methods.

MBOC approach can reach to better results in many cases. For example in Birth-Death and Switch dataset it reaches to higher modularity values than the previous works. Moreover, for the Merge-Split dataset, in several time steps it can find communities with high modularity values.

MBOC results in the switch dataset do not deviate a lot during the time steps. The reason is that this dataset does not change during the time. The MBOC modularity in this dataset is near to the optimal result.

For the birth-death dataset, we have better results, in this dataset communities will be died or being (the core nodes of each community being removed or changed) smaller during the time. Other approaches use local methods for network changes but in MBOC we have a global search in crossover and local search for the graph. Due to this approach, we have a better modularity and NMI.

However, the synthetic Merge-split dataset is changing during each time step. It is merging two communities or splitting them more frequently this could lead MBOC to problems. For solving this problem, local search based approaches are not good. We need a global search-based approach. Because MBOC searches global solutions and local solutions in each time step, we can approximately determine community behaviors at each time step. But as you can see, the result in the modularity which is swinging due to the high differences in communities structures. Thus, in the future, we aim to focus on this problem and improve the MBOC.

V. CONCLUSION

This paper suggests a memetic approach for finding communities in a dynamic social network. In order to do that, three new operators are suggested. These operators are density preserving crossover, dense search mutation, and stochastic local search. These operators can explore and exploit the search space very fast. The method was evaluated by using three synthetic well-known online network datasets. The results show that the MBOC method can outperform the previous method in most cases.

REFERENCE

[1] H. Alvari, A. Hajibagheri, and G. Sukthankar. Community detection in dynamic social networks: A game-theoretic approach. In *Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on*, pages 101–107. IEEE, 2014.

[2] H. Alvari, A. Hajibagheri, G. Sukthankar, and K. Lakkaraju. Identifying community structures in dynamic networks. *Social Network Analysis and Mining*, 6(1):77, 2016.

[3] R. Cazabet, F. Amblard, and C. Hanachi. Detection of overlapping communities in dynamical social networks. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 309–314. IEEE, 2010.

[4] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi. Demon: a local-first discovery method for overlapping communities. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 615–623. ACM, 2012.

[5] J. Duch and A. Arenas. Community detection in complex networks using extremal optimization. *Physical review E*, 72(2):027104, 2005.

[6] F. Folino and C. Pizzuti. An evolutionary multiobjective approach for community discovery in dynamic networks. *IEEE Transactions on Knowledge & Data Engineering*, (1):1, 2013.

[7] M. Gong, Q. Cai, Y. Li, and J. Ma. An improved memetic algorithm for community detection in complex networks. In *Evolutionary Computation (CEC), 2012 IEEE Congress on*, pages 1–8. IEEE, 2012.

[8] M. Gong, B. Fu, L. Jiao, and H. Du. Memetic algorithm for community detection in networks. *Physical Review E*, 84(5):056101, 2011.

[9] D. Greene, D. Doyle, and P. Cunningham. Tracking the evolution of communities in dynamic social networks. In *Advances in social networks analysis and mining (ASONAM), 2010 international conference on*, pages 176–183. IEEE, 2010.

[10] M.-S. Kim and J. Han. A particle-and-density based evolutionary clustering method for dynamic networks. *Proceedings of the VLDB Endowment*, 2(1):622–633, 2009.

[11] A. Lancichinetti and S. Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80(1):016118, 2009.

[12] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng. Analyzing communities and their evolutions in dynamic social networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 3(2):8, 2009.

[13] M. E. Newman. Fast algorithm for detecting community structure in networks. *Physical review E*, 69(6):066133, 2004.

[14] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *nature*, 435(7043):814, 2005.

[15] C. Pizzuti. Ga-net: A genetic algorithm for community detection in social networks. In *International Conference on Parallel Problem Solving from Nature*, pages 1081–1090. Springer, 2008.

[16] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi. Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 101(9):2658–2663, 2004.

[17] G. Rossetti, L. Pappalardo, D. Pedreschi, and F. Giannotti. Tiles: an online algorithm for community discovery in dynamic social networks. *Machine Learning*, 106(8):1213–1241, 2017.

[18] M. Tasgin, A. Herdagdelen, and H. Bingol. Community detection in complex networks using genetic algorithms. *arXiv preprint arXiv:0711.0491*, 2007.

[19] P. Wang, L. Gao, and X. Ma. Dynamic community detection based on network structural perturbation and topological similarity. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(1):013401, 2017.