Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 161670, 7 pages
http://dx.doi.org/10.1155/2013/161670

Research Article

A New Multiobjective Evolutionary Algorithm for Community Detection in Dynamic Complex Networks

Guoqiang Chen,[1,2] Yuping Wang,[1] and Jingxuan Wei[1]

[1] *School of Computer Science and Technology, Xidian University, Xi'an 710071, China*
[2] *School of Computer and Information Engineering, Henan University, Kaifeng, Henan 475004, China*

Correspondence should be addressed to Guoqiang Chen; xidian20120222@163.com

Received 26 March 2013; Accepted 19 May 2013

Academic Editor: Tingwen Huang

Community detection in dynamic networks is an important research topic and has received an enormous amount of attention in recent years. Modularity is selected as a measure to quantify the quality of the community partition in previous detection methods. But, the modularity has been exposed to resolution limits. In this paper, we propose a novel multiobjective evolutionary algorithm for dynamic networks community detection based on the framework of nondominated sorting genetic algorithm. Modularity density which can address the limitations of modularity function is adopted to measure the snapshot cost, and normalized mutual information is selected to measure temporal cost, respectively. The characteristics knowledge of the problem is used in designing the genetic operators. Furthermore, a local search operator was designed, which can improve the effectiveness and efficiency of community detection. Experimental studies based on synthetic datasets show that the proposed algorithm can obtain better performance than the compared algorithms.

1. Introduction

Many real-world complex systems take the form of networks. Acquaintance networks, Internet, power grids, and neural networks are some examples. Networks could be modeled as graphs, where the individual objects are represented by nodes and the interactions among these objects are represented by edges. Community structure, that is, the vertices in networks are often found to cluster into tightly knit groups with a high density of within-group edges and a lower density of between-group edges [1], is one important property of these networks. The detection of such a community structure has great practical meaning.

Traditional analysis of community detection treats the network as a static graph, where the static graph is either derived from aggregation of data over all time or taken as a snapshot of data at a particular time. Researchers have successively proposed many effective detection methods of static network. In fact, dynamic networks capture the modifications of interconnections over time, which allow tracing the changes of network structure at different time steps. Community detection in dynamic networks is attracting increasing interest.

Recently, a framework called temporal smoothness is applied to solve community detection in dynamic networks. In this framework, it is not desirable that the significant changes of clusters structure in a short time period [2]. In order to smooth each community over time, it needs to trade off two competing objectives of snapshot quality, which is that the clustering should reflect as accurately as possible the data coming during the current time step, and temporal cost, which is that each clustering should not shift dramatically from one time step to the successive one. Using this idea, Folino and Pizzuti proposed a multiobjective approach named DYN-MOGA to discover communities in dynamic networks by employing genetic algorithms [3]. In DYN-MOGA, Community Score (CS) and Normalized Mutual Information (NMI) were selected as two objectives to be optimized simultaneously. At the end of each timestamp, *DYN-MOGA* returns a set of solutions contained in the Pareto front. They adopted modularity as a criterion to automatically select one solution with respect to another. Following this

work, Gong et al. introduced a novel multiobjective immune algorithm with local search to solve the community detection problem in dynamic networks [4]. They adopted Modularity and NMI as two objectives to optimize.

But, the modularity has been exposed to resolution limits [5–7]. Fortunato and Barthélemy [5] have recently found that modularity optimization may fail to identify modules smaller than a scale which depends on the total size of the network and on the degree of interconnectedness of the modules, even in cases where modules are unambiguously defined. To overcome the limitations of modularity function, a new measure, named by modularity density, was presented in [8].

In this paper, we present a novel multiobjective algorithm, named DNCD-MOEA, to detect community in dynamic networks. The algorithm adopts modularity density as one objective to measure how well the clustering found represents the data at the current time, and it adopts NMI as another objective to measure the distance between two clusterings at consecutive time steps. Based on the problem-specific knowledge, the new genetic operators are designed to solve the proposed model. In order to improve the quality of the solutions, a local search operator is designed.

The outline of the paper is as follows. Section 2 introduces related background. In Section 3 we present our algorithm and explain its steps. Experimental studies are presented in Section 4. Section 5 concludes this paper.

2. Background

2.1. Notation. We define a static network N_t at time t as $N_t = (V_t, E_t)$, where V_t is a set of objects, each $v_i \in V_t$ denotes a node, and E_t is a set of links, each $v_{ij} \in E_t$ represents an edge that connects two nodes v_i and v_j at time t. The dynamic network N can be defined as a sequence of static networks $N_t = (V_t, E_t)$; that is, $N = \{N_1, N_2, \ldots, N_T\}$, where each N_t represents the snapshot of nodes and edges at time t. Let $C_t = \{C_t^1, C_t^2, \ldots, C_t^k\}$ be the set of all communities in network N_t at time t where C_t^i denotes the ith community at time t.

2.2. Evolutionary Clustering. Chakrabarti et al. first introduced evolutionary clustering in [2] as the problem of clustering data coming at different time steps to produce a sequence of clustering. We adopt this framework to analyze communities and their evolutions by use of the community structure at time $t-1$ to regularize the community structure at time t. At each time step, two conflicting criteria must be simultaneously optimized to produce a new clustering. The first criterion is that the clustering at any point in time should remain faithful to the current data as much as possible. The second is that the clustering should not shift dramatically from one time step to the next. A framework called temporal smoothness is defined in order to satisfy the second property to smooth out each community over time. This framework assumes that the dramatic change of clustering in a short time is not desirable; thus the cost function consisting of two parts is defined as follows:

$$\text{cost} = \alpha \times \text{SC} + (1-\alpha) \times \text{TC}, \tag{1}$$

where SC denotes snapshot cost which measures how well a community structure C_t represents the data at time t and TC denotes temporal cost which measures how similar the community structure C_t is with the previous community structure C_{t-1}. The input parameter α is used by the user to control the level of emphasis on each part of the two objectives. When $\alpha = 1$, it returns the clustering without temporal smoothing. When $\alpha = 0$, however, the framework produces the same clustering structure with the previous time step, that is, $C_t = C_{t-1}$. Thus it can control the preference degree of each subcost by changing the value of parameter α between 0 and 1.

3. Proposed Algorithm

As [3], we adopt the same multiobjective representative frameworks for dynamic community detection which treat snapshot cost SC and temporal cost TC as two competing objectives. Modularity density and NMI are employed to denote snapshot cost SC and temporal cost TC, respectively. The main advantage of this method is that it does not need to fix the control parameter α.

The proposed algorithm DNCD-MOEA is realized under the framework of NSGA-II [9]. The details of objective functions and representation method are given as follows.

3.1. Objective Functions. To denote snapshot cost SC, we adopt modularity density as an objective function to measure the quality of the community. The modularity density is defined as follows:

$$D = \sum_{i=1}^{m} d(G_i) = \sum_{i=1}^{m} \frac{L(V_i, V_i) - L\left(V_i, \overline{V_i}\right)}{|V_i|}. \tag{2}$$

In detail, given an undirected network $G = (V, E)$ consisting of the vertex set $V = \{v_1, v_2, \ldots, v_n\}$ (n is the cardinality of V) and the edge set E. Where $\{V_c\}_{c=1}^m$ is a partition of the vertex set V into m groups, $\overline{V_c}$ is the complement of V_c with respect to V, $L(V_c, \overline{V_c}) = \sum_{i \in V_c, j \in \overline{V_c}} A_{ij}$ ($A = (A_{ij})$ is the adjacent matrix of G), and $|V_c|$ is the cardinality of V_c.

Normalized Mutual Information (NMI) was employed to denote the second objective function, that is, temporal cost. Danon et al. have proved NMI to be a reliable similarity measure [10].

Let $A = \{A_1, A_2, \ldots, A_m\}$ and $B = \{B_1, B_2, \ldots, B_n\}$ denote two partitions of a network in communities, and C denotes the confusion matrix whose element C_{ij} is the number of nodes of the community $A_i \in A$ that are also in the community $B_j \in B$. The NMI(A, B) is defined as follows:

$$\text{NMI}(A, B) = \frac{-2 \sum_{i=1}^{m} \sum_{j=1}^{n} C_{ij} \log\left(C_{ij} N / C_{i.} C_{.j}\right)}{\sum_{i=1}^{m} C_{i.} \log\left(C_{i.}/N\right) + \sum_{j=1}^{n} C_{.j} \log\left(C_{.j}/N\right)}, \tag{3}$$

where $C_{i.}$ ($C_{.j}$) is the sum of the elements of C in row i (column j) and N is the number of nodes. If $A = B$, set NMI$(A, B) = 1$. If A and B are completely different, set NMI$(A, B) = 0$. In this paper, the two objectives of $D(C_t)$ and NMI(C_{t-1}, C_t) will be maximized simultaneously.

3.2. Solution Selection. In fact, the proposed algorithm DNCD-MOEA returns Pareto front at the end of each timestamp, which contains a set of solutions. Each of these solutions corresponds to a different tradeoff between the two objectives and thus to a diverse partitioning of the network consisting of various number of clusters. It needs to establish a criterion to automatically select which solution denotes the optimal partitioning of the current network at each time step. Community score introduced in [11] has been proved to be very effective in detecting communities. In this paper, we adopt community score as a criterion to select, among the solutions found, the solution having the highest value of community score. It is defined as follows.

Let $C_t = \{C_t^1, C_t^2, \ldots, C_t^k\}$ be the set of all communities in network N_t at time t where C_t^i denotes the ith community at time t. The community score of C_t is defined as

$$\mathrm{CS}\left(C_t\right) = \sum_{i=1}^{k} \mathrm{score}\left(C_t^i\right), \tag{4}$$

where

$$\mathrm{score}\left(C_t^i\right) = \frac{\sum_{i \in C_t^i} \mu_i}{\left|C_t^i\right|} \times \sum_{i,j \in C_t^i} A_{ij}. \tag{5}$$

The parameter $\mu_i = (1/|C_t^i|) \sum_{j \in C_t^i} A_{ij}$ denotes the fraction of edges which connect each node i of C_t^i to the nodes in the same community C_t^i. The community score takes into account both the fraction of interconnections among the nodes and the number of interconnections contained in the module C_t^i. It gives a global measure of the network division in communities by summing up the local score of each module found. Thus the larger community score indicates that the community structure is stronger. So, we select the maximum community score value in the set of solutions as the best solution.

3.3. Genetic Representation. The locus-based adjacency representation proposed by Park and Song [12] is used in our community detection algorithm, similar to [11].

As a graph $G = (V, E)$ has N vertices, an arbitrary individual of the population consists of N genes, which can be represented by a number of strings as follows:

$$\left(g_1, g_2, \ldots, g_N\right), \quad g_i \in \{1, 2, \ldots, N\} \quad (i = 1, 2, \ldots, N), \tag{6}$$

where $g_i = j$ denotes that there exists a link between the nodes i and j. This means that thenodes i and j will be in the same community. It is necessary to identify all of the components of the corresponding graph in the decoding step. The nodes participating to the same component are assigned to the same community. Adopting this representation, the main advantage is that the decoding step can be done in linear time and it has been verified as an effective encoding schema for community detection as shown in [3]. Additionally, the number k of clusters is automatically determined by the number of components contained in an individual and is also determined by the decoding step [11].

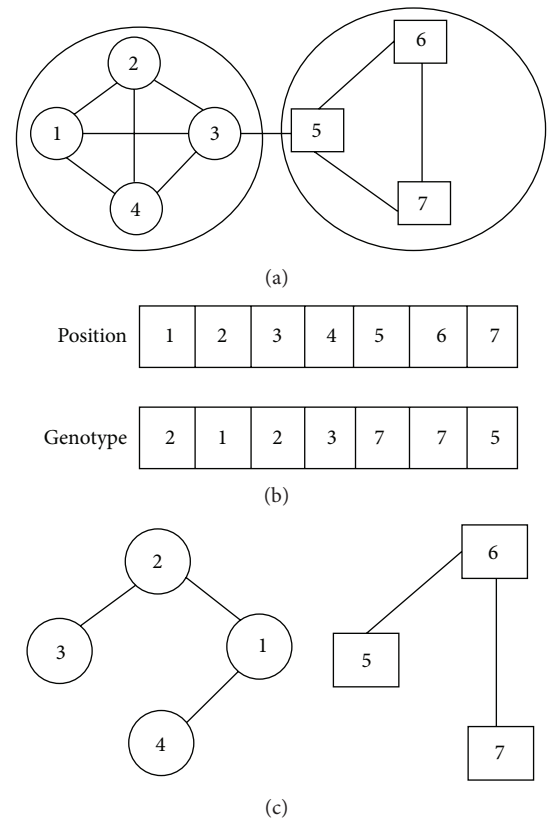

Figure 1: (a) A graph modeled as a network, (b) the representation of a possible genotype, and (c) the graph structure of the genotype to be decoded.

An example of the encoded genotype and corresponding network is shown in Figure 1. As shown in Figure 1(a), the supposed network consists of seven nodes numbered from 1 to 7. It is obvious that the network can be partitioned into two groups visualized by different shape. A possible genotype as shown in Figure 1(b), which corresponds to the optimal solution, is translated in the graph structure given in Figure 1(c). Each connected component provides a grouping of nodes that corresponds to the partitioning of the network in Figure 1(a).

3.4. Initialization. If an individual is created randomly, it may not be a feasible solution. In fact, a randomly generated individual could contain an allele value j in the ith position, but no connection exists between the two nodes i and j; that is, the edge (i, j) is not present. In order to overcome this limitation, we should check whether the individual is safe after an individual is created. When the individual is not safe, that is, a gene i contains a value j, but link (i, j) does not exist, it needs to repair to ensure whether the individual is safe or not. Safe individuals improve the convergence of the method because the space of the possible solutions is restricted.

3.5. Crossover and Mutation. As in [3], we use uniform crossover which can guarantee the maintenance of the effective connections of the nodes in the network in the child individual. Given two arbitrary safe parent individuals and that

a random binary vector is created, the genes are selected from the first parent if the vector element is 1, and the genes are selected from the second parent if the vector element is 0. Then the genes are combined to form a child. Because of the biased initialization, the child created from the two safe parents is safe also. In the child, if a position i contains a value j, then the edge (i, j) exists. The uniform crossover is shown in Figure 2.

Crossover operator is regarded as a macroscopic operation on individuals, while the mutation operator is regarded as a microcosmic operation on individuals. The mutation operator that randomly changes the value j of the ith gene causes a useless exploration of the search space. In order to guarantee the mutated child is safe as the crossover operation, the possible value of an allele after mutating is restricted to be one of the replaced gene's neighbors.

3.6. The Pseudocode of the Proposed Algorithm. The pseudocode of our DNCD-MOEA algorithm is described in Algorithm 1.

3.7. Local Search Strategy. Local search is proved to be an effective algorithm. The mutation operator is regarded as a microcosmic operation on individuals and can achieve its local search function by moving single nodes between communities. Inspired by this idea, we adopt mutation operator in our local search algorithm.

In local search algorithm, it needs to convert multiple objectives into a single objective function. In our study, we select a weighted objective as follows [3]:

$$G(x) = \sum_{i=1}^{2} r_i g_i(x), \tag{7}$$

where $g_i(x)$ is the objective function which is described in (2) and (3), respectively, and r_i is the nonnegative weights for the two objectives. The weights are calculated in a special way as follows:

$$r_i = \frac{\left(g_i(x) - g_i^{\min}\right) / \left(g_i^{\max} - g_i^{\min}\right)}{\sum_{j=1}^{2} \left(g_j(x) - g_j^{\min}\right) / \left(g_j^{\max} - g_j^{\min}\right)}, \tag{8}$$

where $g_i^{\max}$ or $g_i^{\min}$ is the maximum or minimum value of each objective function $g_i(x)$ in the obtained dominant population and $\sum_{i=1}^{2} r_i = 1$. In Algorithm 2, the detailed pseudocode of the local search algorithm is given.

3.8. Computational Complexity. The computational complexity of DNCD-MOEA algorithm is investigated in this subsection. The DNCD-MOEA algorithm contains two major components: the main program based on the NSGA-II framework and the subprogram of local search operation. The time consuming of the main program mainly consists of three parts: generating nondominated sorting, calculating crowding distance assignment, and constructing partially ordered set. Let S denote the size of population and r denote the number of objects. The computational complexity of the three parts is $O(r(2S)^2)$, $O(r(2S)\log(2S))$, and $O(2S\log(2S))$, respectively. So the total time complexity of the main programming is $O(rS^2)$. The local search operation mainly contains two loops and its computational complexity is $O(MN)$. So, the total computational complexity of DNCD-MOEA algorithm is $O(rS^2) + O(MN)$. Because $M < S$, $r = 2$, and in practice $N < S$, the time complexity can be simplified to $O(S^2)$.

Parent1	2	1	2	3	6	7	5
Parent2	2	1	4	3	6	5	6
Binary mask	1	1	0	1	0	0	1
Offspring	2	1	4	3	6	5	5

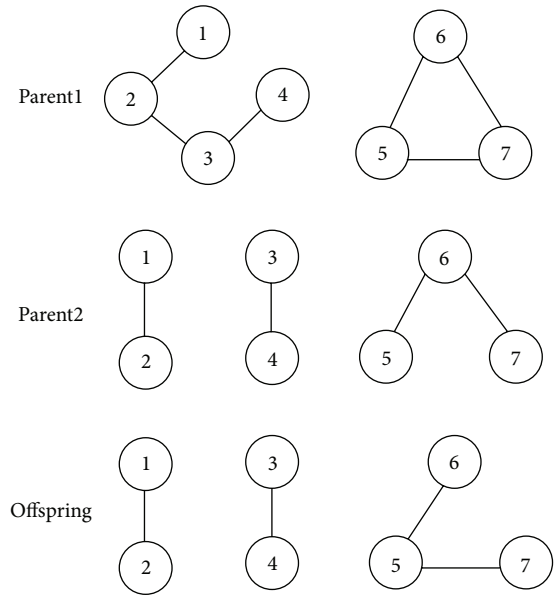

Figure 2: Illustration of the uniform crossover.

4. Experiments

In order to check the ability of our approach on a dynamic network, we adopt the method proposed as [3] to generate data simulating dynamic network. Firstly, we generate synthetic datasets by following the procedure suggested by Girvan and Newman [1]. The datasets are generated using the software package designed by Lancichinetti et al. [13]. The data have 128 nodes, which are divided into 4 communities of 32 vertices each. Every node has an average degree of 16 and shares a number Z_{out}, which represents the average number of edges from a node to nodes in other communities. If we increase Z_{out}, then the noise level of network is augmented. On the other hand, if we decrease Z_{out}, then the noise level in the network decreases.

A parameter μ, which represents the average ratio of external degree/total degree for each node, is used to control the noise level in the dynamic networks. If the value of μ is increased, then the network will become more noisy in the sense that the community structure becomes less obvious and hard to detect. In this study, by setting $\mu = 0.1$ and $\mu = 0.3$, the datasets under two different noise levels are generated.

Program DNCD-MOEA
Input: The number T of time steps, the sequence of dynamic network $N = \{N_1, N_2, \ldots, N_T\}$
Output: The sequence of community structure detected in the dynamic network $C = \{C_1, C_2, \ldots, C_T\}$
Begin
Step 1: Set $t = 1$. Generate the initial community structure $C_1 = \{C_1^1, C_1^2, \ldots, C_1^k\}$ of the network N_1 using GA-Net algorithm. Set $t = t + 1$.
Step 2: If $t > T$, return the sequence of community structure $C = \{C_1, C_2, \ldots, C_T\}$ as the output, algorithm stops; Else, go to Step 3.
Step 3: Set $s = 1$. Randomly generate individuals whose length equals the nodes number $n_t = |V_t|$ of network N_t as an initial population P_s;
Step 4: While termination condition is not satisfied do
Step 4.1: Create a new population Q_s of offspring by applying the variation operators on population P_s;
Step 4.2: Combine the parents P_s and offspring Q_s into a new pool R_s and;
Step 4.3: Decode each individual I of the population R_s to generate the partitioning $C_t = \{C_t^1, C_t^2, \ldots, C_t^{k_t^I}\}$ of the network N_t in k_t connected components;
Step 4.4: Evaluate the two fitness values of the translated individuals;
Step 4.5: Partition R_s into fronts, assign a rank to each individual and sort them according to nondomination rank;
Step 4.6: Select individuals based on rank and crowding length to comprise new population P_{s+1};
Step 4.7: Select the dominant individuals L_s in P_{s+1},
Step 4.8: Perform the local search algorithm on the selected individuals in L_s to generate the new dominant population L_s'. Update the dominant population L_s with L_s' in P_{s+1}.
Step 4.9: $s = s + 1$
End while
Step 5: Select the individual which has the maximum Community Score on the Pareto front. Decode the selected individual to get the community structure $C_t = \{C_t^1, C_t^2, \ldots, C_t^{k_t}\}$ of the network N_t.
Step 6: Set $t = t + 1$, go to Step 2.
End

Algorithm 1

In order to introduce dynamics into the network, we let the community structure of the network evolve in the following way. After time step 1, 5% of the nodes is randomly *choused* to leave their original community and randomly assigned to the other three communities at each time step. After the community memberships are decided, links are generated by following the parameter μ. We generate the network with community evolution in this way for 10 time steps.

Figure 3 shows the statistical average value of normalized mutual information with the ground truth over the 10 networks for the 10 timestamps when the value of $\mu = 0.1$ (Figure 3(a)) and $\mu = 0.3$ (Figure 3(b)). Both figures show that the proposed algorithm DNCD-MOEA can achieve better accuracy than the compared method. Especially, the average values of NMI at each time step obtained by DNCD-MOEA are closed to 1 when $\mu = 0.1$.

Figure 4 reports the community score obtained by the two algorithms at each time step when $\mu = 0.1$ (Figure 4(a)) and $\mu = 0.3$ (Figure 4(b)). It indicates that the corresponding network is densely connected within each subnetwork when the obtained value of community score is larger. From Figure 4, it can be found that the algorithm DNCD-MOEA outperforms

Program: Local Search Algorithm:
Input: Given a dominant population L_s before local search at the sth generation, the size M of dominant population L_s, the number N of local search.
Output: the improved population L'_s in the sth generation.
Begin
(1) for $i = 1$ to M
(2) $j = 1$; $find = false$
(3) while $j < N$ and not $find$
(4) Create a new individual applying the mutation operator on the ith individual of dominant population L_s;
(5) Calculate the objective function value of the new individual according to formula (7). If its value is greater than that before local search, add the new individual to L'_s, $find = true$;
(6) $j = j + 1$;
(7) end while
(8) if not $find$, adds the current individual to L'_s
(9) end for
END

Algorithm 2

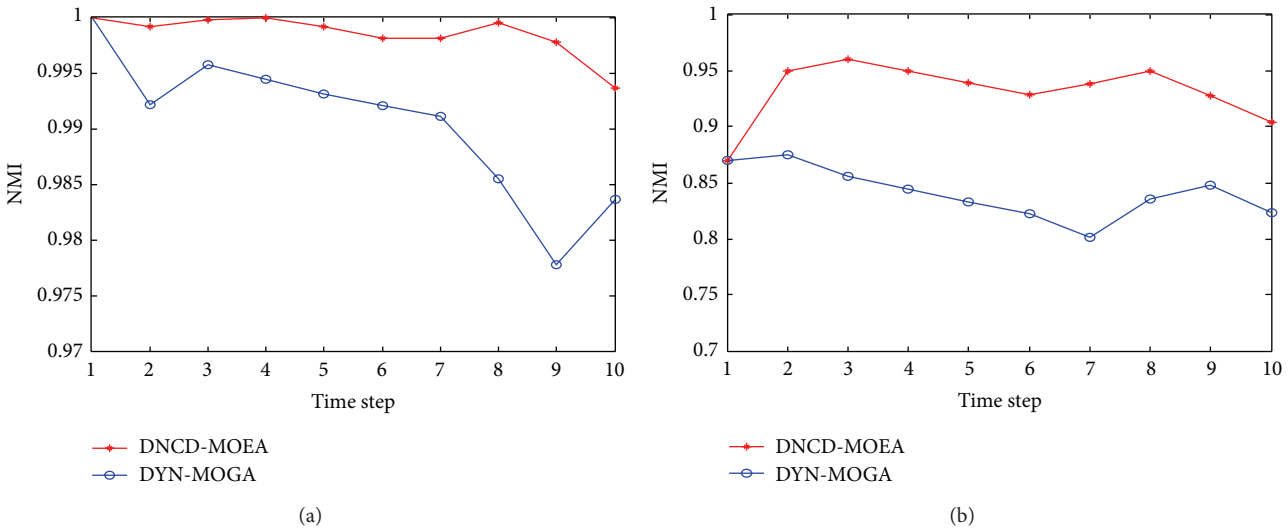

Figure 3: Normalized mutual information (NMI) of clustering results when (a) $\mu = 0.1$ and (b) $\mu = 0.3$.

the algorithm DYN-MOGA. That is to say, the community structure obtained by our algorithm is more closed to the true community structure.

5. Conclusions

In this paper, a novel multiobjective algorithm for detecting communities in dynamic networks is proposed. Based on optimizing modularity density and NMI, the algorithm automatically provides a solution representing the best tradeoff between the accuracy of the community structure obtained with respect to the data of the current time step and the deviation from one time step to the successive. To solve the proposed model, the new genetic operators are designed based on the problem-specific knowledge. A local search operator is designed in order to improve the quality of the solutions. Experimental results on synthetic datasets show that the proposed algorithm can obtain better performance than the compared algorithm. Future research will aim at improving efficiency and applying proposed algorithm to process real-life network.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grants no. 61272119 and no. 61203372.

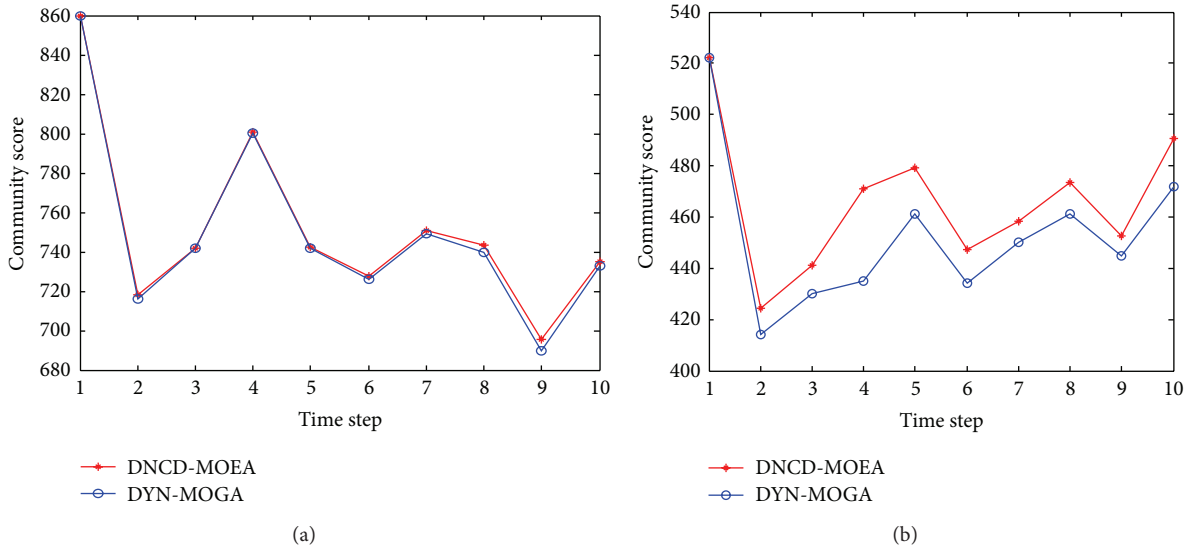

Figure 4: Community score when (a) $\mu = 0.1$ and (b) $\mu = 0.3$.

References

[1] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, pp. 7821–7826, 2002.

[2] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *Proceedings of the 12th ACM International Conference on Knowledge Discovery and Data Mining (KDD '06)*, pp. 554–560, August 2006.

[3] F. Folino and C. Pizzuti, "A multiobjective and evolutionary clustering method for dynamic networks," in *Proceedings of the International Conference on Advances in Social Network Analysis and Mining (ASONAM '10)*, pp. 256–263, August 2010.

[4] M. G. Gong, L. J. Zhang, J. J. Ma, and L. C. Jiao, "Community detection in dynamic social networks based on multiobjective immune algorithm," *Journal of Computer Science and Technology*, vol. 27, no. 3, pp. 455–467, 2012.

[5] S. Fortunato and M. Barthélemy, "Resolution limit in community detection," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 1, pp. 36–41, 2007.

[6] M. Rosvall and C. T. Bergstrom, "An information-theoretic framework for resolving community structure in complex networks," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 104, no. 18, pp. 7327–7331, 2007.

[7] S. Muff, F. Rao, and A. Caflisch, "Local modularity measure for network clusterizations," *Physical Review E*, vol. 72, no. 5, Article ID 056107, 4 pages, 2005.

[8] Z. P. Li, S. H. Zhang, R. S. Wang, X. S. Zhang, and L. N. Chen, "Quantitative function for community detection," *Physical Review E*, vol. 77, no. 3, Article ID 036109, 9 pages, 2008.

[9] N. Srinivas and K. Deb, "Multiobjective optimization using nondominatedsorting in genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.

[10] L. Danon, A. Díaz-Guilera, J. Duch, and A. Arenas, "Comparing community structure identification," *Journal of Statistical Mechanics*, vol. 2005, no. 9, Article ID P09008, 2005.

[11] C. Pizzuti, "GA-Net: a genetic algorithm for community detection in social networks," in *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature*, pp. 1081–1090, September 2008.

[12] Y. J. Park and M. S. Song, "A genetic algorithm for clustering problems," in *Proceedings of 3rd Annual Conference on Genetic Algorithms*, pp. 2–9, 1989.

[13] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical Review E*, vol. 78, no. 4, Article ID 046110, 5 pages, 2008.