Applied Intelligence (2018) 48:3081–3093
https://doi.org/10.1007/s10489-017-1135-5

A multiobjective discrete bat algorithm for community detection in dynamic networks

Xu Zhou[1,2] · Xiaohui Zhao[2] · Yanheng Liu[3,4]

Published online: 7 February 2018

Abstract
Some evolutionary based clustering approaches for community detection in dynamic networks need an input parameter to control the preference degree of snapshot and temporal cost. To break the limitation of parameter selection and improve the quality of detecting communities in dynamic network further, a multiobjective discrete bat algorithm (MDBA) is proposed to detect community structure in dynamic networks in this paper. In the proposed algorithm, the bat location updating strategy is designed in discrete form. In addition, turbulence operation and mutation strategy are presented to guarantee the diversity of the population. The non-dominated sorting and crowding distance mechanism are used to keep good solutions during the generation. The experimental results both on synthetic and real networks show that MDBA algorithm is competitive and will get higher accuracy and lower error rate than the compared algorithms.

Keywords Community detection · Multiobjective bat algorithm · Swarm intelligence

1 Introduction

The nodes in complex network can reveal some regular structure features such as small world effect, while community structure is another interesting property which is revealed in the study of networks [1, 2]. In complex networks, communities are groups of nodes with relatively denser edges within groups than those between them [3]. Community structure has been considered primarily in the context of static networks, community detection is one of the most popular topics in the field of network analysis, and it helps to understand the potential social structure of users. However, complex systems are not static in reality. Entities and their interactions can be created or disappeared resulting in dynamic effects [4]. The dynamic nature of the network also makes it hard for traditional community detection algorithm to deal with it. Understanding the formation and evolution of communities is a long standing research topic, identifying and detecting communities in the dynamic networks not only has particular prominence but also has immediate applications. Detecting community can be widely applied in terrorist organization identification, protein function prediction, public opinion analysis and other areas.

The dynamic community detection algorithm is more and more concerned. In order to discover community structure in dynamic networks, there are some ways to solve it by multiobjective optimization method, properly speaking, these algorithms still have room for improvement in accuracy. In this paper, we focus on bio-inspired algorithm named bat algorithm, it is a novel optimization method that models prey hunting behaviors of bats. It has lower computational complexity and fewer adjustable parameters. Though it has been employed to detect community structure in static networks [5, 6], a discrete bat algorithm is proposed for community detection in dynamic networks here. The primary contributions of our algorithms are listed as follows (1) effective bat location updating strategy is designed in discrete form (2) mutate operator and turbulence operation are incorporated to guarantee the diversity of the population. We conduct experiments on both synthetic and real networks to evaluate community detection algorithm. In all, the proposed algorithm can get better accuracy and lower error rate than other algorithms.

✉ Yanheng Liu
yhliuj@163.com

1 Center for Computer Fundamental Education, Jilin University, Changchun, Jilin 130012, China

2 College of Communication Engineering, Jilin University, Changchun, Jilin 130012, China

3 College of Computer Science and Technology, Jilin University, Changchun, Jilin 130012, China

4 Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun, Jilin 130012, China

The remainder of this paper is structured as follows. A brief review of community detection in dynamic networks and introduction of bat algorithm are shown in Section 2. In Section 3, the proposed discrete algorithm is described in detail. Additionally, the experiments and their corresponding analysis are shown in Section 4. In Section 5, conclusions and future work are explained.

2 Related works and background knowledge

2.1 Related work

There are many literatures solving the problem of dynamic community detection. The current method of finding community changing over time can be divided into two categories. One category of algorithms is to detect static communities on each snapshot independently, and to match the communities detected with the communities found on the previous one for each snapshot. These methods put community detection and evolutionary analysis separately [7, 8]. The results obtained by these methods will lead to unexpected fluctuation. That is to say, for two similar networks with tiny modifications, the algorithm can provide very different results. To overcome the instability of this kind of approach, the other category takes a unified framework to detect communities and analyze their evolutions, it detects static communities on the first snapshot, and detects communities on snapshot t+1 using network at t +1 and communities at time t . In this line of research, Sun proposed an incremental density-based clustering algorithm IncOrder for detecting communities in dynamic networks [9]. It constructed a core-connected chain for dynamic network, and the result is independent of the traversal order. Chakrabarti first proposed a framework called temporal smoothness to solve community detection in dynamic networks [10]. $cost = \alpha \times SC + (1-\alpha) \times TC$. It requires maximizing the accuracy of the clustering and minimizing the difference of clustering from one time step to the successive one, a parameter used by the user to represent the weight of snapshot cost(SC) and temporal cost(TC). Chi proposed an evolutionary spectral clustering approach and this approach took the graph cut as a metric for measuring community structure in the real blog network [11]. A dynamic community detection algorithm based on the dynamic programming and exhaustive searching proposed by Tantipathananandh [12], it has a better performance on the synthetic datasets. Lin proposed a framework for analyzing communities and evolutions in dynamic networks (FacetNet) [13]. It employs the nonnegative matrix factorization method, but it needs number of clusters in advance and it use a parameter controlling the preference degree of the snapshot cost and temporal cost.

To avoid a parameter controlling the preference degree of the snapshot cost and temporal cost. With the development of research methods, naturally, dynamic network community detection can be modeled as a multiobjective optimization problem to simultaneously the two conflicting objectives of snapshot cost and temporal cost. The first classic method using multiobjective optimization is introduced by Pizzui, it is mainly based on multiobjective genetic algorithm to optimize snapshot quality and temporal quality simultaneously [14]. They adopted a multiobjective genetic algorithm to optimize the two objectives Community Score (CS) and Normalized Mutual Information (NMI). It can get good results on small datasets. Zhou presented a multiobjective Biogeography-based optimization algorithm with decomposition (MBBOD) for community detection in dynamic networks [15]. It designed new migration and mutation operators in BBO algorithm. It automatically provides a solution representing the best trade-off between the accuracy of the clustering obtained, and the deviation from one time step to the successive. Ma proposed a decomposition-based multiobjective community algorithm to reveal community structure and its evolution in dynamic networks [16]. It employed the framework of multiobjective evolutionary algorithm based on decomposition. The experiment results are good. Gong proposed a community detection algorithm in dynamic networks based on the non-dominated neighbor immune algorithm [17], the experimental results proved the effectiveness of this algorithm. Pizzui proposed a novel method which optimizes two functions simultaneously [18], it is realized under the framework of NSGA-II [19]. Multiobjective evolutionary algorithms existing basically adopt genetic algorithm with NSGA-II or decomposition for solving community detection in dynamic network, but other evolutionary algorithms should also be well studied and it still has room to use other evolutionary algorithm to improve the accuracy of detecting community structures in dynamic networks.

2.2 Introduction of bat algorithm

The bat algorithm (BA) is a bioinspired metaheuristic based on the echolocation system of bats. It is derived from the principle of echolocation in bats during prey capture and avoidance of obstacles [20, 21]. Yang designed the algorithm based on the nature that bats emit ultrasonic pulses to the surrounding environment with hunting and navigation purposes. This ability aids the bats in finding prey in the dark. Inspired by this characteristic of bats, the optimal value searching process can be regarded as the echolocation of bats. BA combines the advantages of PSO and GA algorithm. BA exhibits excellent performance in solving optimization problems. It has been successfully used for optimization of job-shop scheduling problem

[22], feature selection [23] and so on. A multiobjective bat algorithm is shown for simplicity [24]. Yang used a weighted sum to combine all objectives f_k into a single objective $f = \sum_{k=1}^{K} w_k f_k$. The pseudo code of the multiobjective bat algorithm can be described in Table 1.

3 Description of the proposed method

A dynamic network is a series of network snapshots at two or more time points. Dynamic community detection is the procedure to find the community structure at every time step in the continuous changed network along with time. The definition of the dynamic community detection is described as follows: Given a dynamic network with T time steps $G = \{G_1, G_2, ..., G_T\}$. The network at time i can be modeled as a graph $G_i =< V_i, E_i > (1 \leq i \leq T)$, where V_i is node set and E_i is edge set. The final partition result of network with T time steps is $P = \{P_1, P_2, ..., P_T\}$, where $P_i (1 \leq i \leq T)$ is the partition result of a network G_i at time i, and it is composed of m communities $P_i = \{C_1, C_2, ..., C_m\}$.

In this section, the proposed method MDBA for community detection in dynamic network is described. Firstly, the framework of MDBA by integrating the non-dominated sorting and the crowding distance method is elaborated. Then two objective functions we adopt are introduced. Next a bat location representation scheme is introduced. In the end the bat location updating strategy is designed. In addition, turbulence operation and mutate operator are given to guarantee the diversity of the population.

3.1 Framework of MDBA

The framework of MDBA by integrating the nondominated sorting and the crowding distance method is put forward in this section. The specific framework of the algorithm is shown in Algorithm 1. First, a hierarchical agglomeration algorithm (CNM) [25] is used to get the community partition result at the first time step. To be pointed, bat algorithm has the advantages of simple structure, few adjustable parameters, easy to understand and realize. To make it available for solving dynamic community detection problem, it is necessary to design a discrete version of bat algorithm. From the second time step, a number of bat population is created based on representation schema which is introduced in Section 3.3, and then nondominated solution are obtained after calculating the fitness of bats. During the iterative procedure (line 7 to line 12), The new population is generated according to Algorithm 2 which is a discrete bat algorithm(DBA) with redefined location updating rules. Next we will make use of crowding distance and non-dominated mechanism to update the nondominated solutions. At last, selecting the individual with the highest modularity value in the Pareto optimal solutions at the end of each generation, and outputting the network partition at the current time step. Repeat the above-mentioned operations until we get the community structure at every time step.

Table 1 The pseudo code of the multiobjective bat algorithm

1.Initialize k objective functions $f_1(x), ..., f_k(x)$ and $x = (x_1, x_2, ..., x_d)$
2.Initialize the bat population $x_i (i = 1, 2, ..., n)$,velocity v_i, pulse frequency at x_i,pulse rate r_i and loundness A_i
3.For j=1 to n
4. Form a single objective $f = \sum_{k=1}^{K} w_k f_k$ and $\Sigma_{k=1}^{K} w_k = 1$
5. While (t< max number of iterations)
6. Generate new solutions by adjusting frequency and update velocities and locations
7. if $rand > r_i$
8. Random walk around a selected best solution
9. End if
10. Generate a new solution by flying randomly
11. If $rand < A_i \& f(x_i) < f(x_*)$
12. Accept new solution, increase r_i and reduce A_i
13. End if
14. Rank the bats and find the current best
15. End while
16.End For
17.Postprocess results and visualization

Algorithm 1 Framework of MDBA

Input: A dynamic network $DN = \{G_1, G_2, ..., G_T\}$
Output: Network partition at each time $P = \{P_1, P_2, ..., P_T\}$
1: Initialize the basic parameters of the algorithm: the size of the population N, the number of time steps T, the maximum number of generations $genmax$
2: Get the partitioned result $P_1 = \{P_1^1, P_1^2, ..., P_1^m\}$ of G_1 by adopting the CNM algorithm
3: **for** $t = 2 : T$ **do**
4: Initialize a fixed number N of bat according representation schema, $pop = \{p_1, p_2, ..., p_N\}$
5: Calculate the fitness value
6: Find the current Pareto optimal solutions rep
7: Set $gen = 0$
8: **while** $gen < genmax$ **do**
9: Generate new bat population $npop$ according to Algorithm 2
10: Use non-dominated sorting and crowding distance to select the top N bats from all $(pop+npop)$ bats and update Pareto optimal solutions rep
11: $gen = gen + 1$
12: **end while**
13: Choose an optimal bat
14: Decode the bat to get partition result at time step t, $P_t = \{P_t^1, P_t^2, ..., P_t^m\}$
15: $t = t + 1$
16: **end for**

3.2 Objective functions

As stated in [10], a cost function in the framework of temporal smoothness composed by two parts, one is snapshot cost and the other one is temperal cost. Here modularity (Q)is employed for snapshot cost to measure the quality of current time and Normalized Mutual Information (NMI) is employed for temporal cost to evaluate the similarity between current time and previous time. The modularity criterion is widely used in accessing the goodness of the partition, the definition is written as follows in (1):

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_i, c_j) \tag{1}$$

where k_i is the degree of node i, $k_i k_j/2m$ is the connected probability between node i and node j, δ is Kronecker function. A_{ij} is adjacent matrix of graph, if there is an edge between node i and node j, then $A_{ij} = 1$, else A_{ij}=0. c_i represents the community that node i belongs to. If $c_i = c_j$ then $\delta(c_i, c_j) = 1$, else $\delta(c_i, c_j) = 0$. Q is defined as the difference between the fraction of edges inside the community and the expected fraction of edges that would be in the random graph with the same community division. In general, Q value is between 0 and 1, and the value approaching to 1 indicates a strong community structure.

The NMI criterion is used in measuring the similarity between the true community structure and the detected community structure [26]. Given two partitions A and B of a network in communities, let C be the confusion matrix whose element C_{ij} is the number of nodes in the i-th community of the partition A that are also in the j-th community of the partition B. The normalized mutual information of partition A and B is defined as shown in (2)

$$NMI = \frac{-2\sum_{i=1}^{c_A}\sum_{i=1}^{c_B} C_{ij}\log\left(C_{ij}N/C_{i.}C_{.j}\right)}{\sum_{i=1}^{c_A} C_{i.}\log\left(C_{i.}/N\right) + \sum_{j=1}^{c_B} C_{.j}\log\left(C_{.j}/N\right)} \tag{2}$$

where c_A is the number of groups in the partition A, c_B is the number of groups in the partition B, $C_{i.}$ is the sum of the elements of C in row i and $C_{.j}$ is the sum of the elements of C in column j, and N is the number of nodes. If $A = B$, the NMI value is 1. If partition A and B are completely different,then the value of NMI is 0.

3.3 Bat location representation

It is noteworthy that the original bat algorithm has been applied to solve continuous problems. But community detection is a kind of discrete problem. Therefore, some modification of the original bat algorithm is necessary in order to prepare it for addressing community detection. Firstly, it is necessary to construct a reasonable representation schema to represent the solutions. Each bat should be encoded to represent a possible and feasible solution in algorithm. Here we encode the bat based on locus-based adjacency representation. Suppose each bat $x_i^t = \left(x_{i1}^t, x_{i2}^t, ..., x_{id}^t\right)$ consists of d dimensions, each value of the j-th dimension x_{ij} is set to one node k randomly selected from the neighbors of node j, $nei(j)$, where $nei(j) = \left\{k \in V, A_{jk} = 1\right\}$, it is interpreted as a link between node j and k. The decoding phrase can be accomplished in linear time. The main advantage of this representation is the number of cluster will be automatically determined during decode phrase.

An example of encoding and decoding schema is shown in Fig. 1. It is easy to implement and it guarantees the feasibility of calculating the fitness value. Figure 1a is a small network of 8 nodes, Fig. 1b is a feasible encoding pattern. Figure 1c is the decoding structure corresponding to Fig.1b, it partitions the network into two communities, one community has 4 nodes and the other has 4 nodes.

3.4 Discrete bat algorithm

In the original bat algorithm, the main equation of updating bat location based on velocity and frequency is shown in (3)–(5),

$$f_i = f_{min} + (f_{max} - f_{min})\beta \tag{3}$$

$$v_i^t = v_i^{t-1} + \left(x_i^{t-1} - x_*\right) f_i \tag{4}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{5}$$

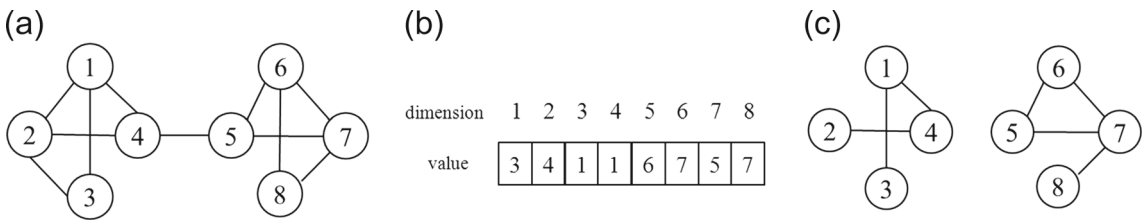

Fig. 1 Example of bat location encoding and decoding **a** a network of 8 nodes **b** encoding pattern **c** decoding result

Where β is a random number, f_{min} is minimum frequency, f_{max} is maximum frequency. Where v_i^t is the new velocity, v_i^{t-1} is the previous velocity of the i-th bat, x_i^t is the current position of the i-th bat, x_i^{t-1} is the previous position of the i-th bat.

Next, since the location of bat is composed of an integer vector, the mathematical updating rules in continuous bat algorithm no longer fit the discrete situation. Therefore, we redefine them because of the discrete nature of community detection problem. First of all, a new velocity formula in proposed in formula(6).

$$v_i^t = v_i^{t-1} + \left(x_i^{t-1} \oplus x^{best}\right) f_i \tag{6}$$

For discrete version of bat algorithm, we can not get the difference by subtraction of current bat and current best bat directly. It is necessary to propose a new velocity formula. The velocity at step t designed is given in (6), where v_i^t is the new velocity of the i-th bat, $v_i^{t-1}$is the previous velocity of the i-th bat, x_i^{t-1} is the current position of the i-th bat, x^{best} is the current best position, f_i is the current frequency, $\oplus$ is XOR. Here, we employ XOR operation to represent the difference between the current bat and the current best individual. In fact, bat will adjust its velocity by learning from the current best. The learning process is actually a comparison between the positions.

It is known that a new location of bat is based on velocity and old bat location. Since f is continuous value and the bat location is integer vectors in our approach, it is necessary to transform the continuous velocity value obtained in (6) to discrete value. In order to get the discrete values, we map v_{id}^t to the solution domain that is composed of 0 and 1 firstly as shown in the (7–8). After using sigmoid function to map the velocity, the new bat location formula could be calculated as shown in (9). If the velocity of the d-th dimention of i-th bat is equal to zero $v_{id}^t = 0$, the new value of d-th dimension keeps the original one, else the new value is equal to the value of the d-th dimension of the best bat. The value in each dimension of the new bat location is effective, each bat is safe enough to avoid the meaningless divisions of the original network.

$$sig(v_i^t) = 1/(1 + exp(-v_i^t)) \tag{7}$$

$$v_{id}^t = \begin{cases} 0, & if rand() < sig(v_{id}^t) \\ 1 & otherwise \end{cases} \tag{8}$$

$$x_{id}^t = \begin{cases} x_{id}^{t-1} & if v_{id}^t = 0 \\ x_d^{best} & if v_{id}^t = 1 \end{cases} \tag{9}$$

The procedure of DBA is given in Algorithm 2. A new bat population is obtained after Algorithm 2. First of all, it is to initialize pulse rates, the loudness and pulse frequency at each bat. Then it is to generate a new location according to (7)–(9) defined. If a random number is larger than pulse rates, a new position is generated according to Algorithm 3, else a new position is generated based on Algorithm 4, both Algorithm 3 and Algorithm 4 are designed to keep the diversity of population and avoid algorithm into local optimum at the same time. When some condition is satisfied, we keep the new solution and reduce the value of loudness.

Algorithm 2 The framework of DBA

Input: old population pop,nondominated solutions rep
Output: new population $npop$
1: Define pulse frequency f_i at pop_i, $0 \le f_i \le 2$
2: Initialize pulse rates r_i and the loudness A_i
3: **for** $i = 1$ in N **do**
4: Randomly select a bat from rep as best
5: Generate a new solution x by adjusting frequency and updating velocities and solutions according to (7)–(9)
6: **if** $rand > r_i$ **then**
7: Generate a new bat nx according Algorithm 3
8: **else**
9: Get a new bat nx according Algorithm 4
10: **end if**
11: **if** $rand < A_i$ and nx dominates x
12: Accept the new solution and $A_i = 0.95 * A_i$
13: **end if**
14: **end for**

Algorithm 3 is a turbulence operation. It aims to improve the quality of population to escape from local optima. The specific implement step of Algorithm 3 is depicted as follows. Firstly, to get label of each dimension of the bat after decoding, and then to get a new value of each dimension based on the strategy. The strategy includes three steps (line 3 to line 5). It is mainly based on the guidelines that if most neighbors belong to a cluster, theses neighbors form a set, and to change the value of current dimension to a neighbor from the set.

Algorithm 3 Turbulence operation

Input: bat $b1$
Output: new bat $b2$ after turbulence operator
1: decode bat $b1$ to get cluster label of each dimension, and compute objective values
2: **for** each dimension d of $b1$ **do**
3: find all neighbors $nei(d)$ of d and record cluster label of neighbors
4: find cluster l that most neighbors belong to it, and the most neighbors in cluster l is recorded as set s
5: select one neighbor r randomly from set s as new dimension value of $b1$, $b2(d) = r$
6: **end for**

Algorithm 4 is introduced to mutate the bat and the procedure is to be depicted as follows. For each dimension of the bat, to find the neighbor of each dimension to form a set and select one neighbor randomly from neighbor set when the condition is satisfied. Otherwise, the dimension value keeps unchanged. Mutation strategy can avoid algorithm into local optimum, it can keep the diversity of population at the same time.

Algorithm 4 Mutation strategy

Input: bat $b1$
Output: new bat $b2$ after mutation operator
1: **for** each dimension $j = 1$ in d **do**
2: **if** $rand() < 0.2$ **then**
3: $t = nei(j)$ //get the neighbor set of node j
4: generate a value r in t that is different from current value, then $b2(j) = r$
5: **else**
6: $b2(j) = b1(j)$
7: **end if**
8: **end for**

3.5 Computational complexity of MDBA

The time complexity of MDBA algorithm is analyzed. For each iteration, the non-dominated sorting is $O(k(2N)^2)$ and the crowding distance assignment is O(k(2N)log(2N)), where N is number of population and k is the number of objectives.Here, k is equal to 2. The complexity of fitness computation consists of two parts. Modularity computation needs $O(m)$, where m is the number of edges, NMI computation needss $O(n)$, Supposing the total number of iteration is I, then the time complexity of the algorithm is $O(IN^2 * (m + n))$.

4 Experimental results and analysis

In this section, the experimentation performed in this study is detailed. The results obtained by proposed algorithm are shown and compared with the ones obtained by recent algorithms including MDPSO, MBBOD and DYNMOGA. MDPSO is based on particle swarm optimization algorithm. We employ the main framework in [27], but take different objectives to solve community detection in dynamic networks. DYNMOGA is based on genetic algorithm, MBBOD is a biogeograph based optimization algorithm. All the experiments conducted have been performed on an Intel core i5 computer with 2.3 GHz and a RAM of 4 G. Matlab has been used as the programming language. The parameters setting are as follows after conducting a series of experiments.The population size N is 100 and the number of maximum generations gen is 20. The pulse rates r_i is 0.5. The loudness A_i is 0.8. To make the experiments more accurate, every algorithm has been run 20 times. An average result of the 20 independent runs is obtained and compared. The criterions of evaluating the performance of this algorithm are NMI and the error rate. The error rate measures the distance between matrix G which stores the real community partition and matrix Z which stores the communities obtained by the algorithm. The specific calculating method is shown in (10), where matrix Z has n rows and k columns. Every node in n belongs to only one community. In the same way, matrix G represents the nodes in the network and the real community partitions of the network.

$$Error = \left\| ZZ^T - GG^T \right\| \tag{10}$$

The smaller the error rate value, the better the performance of the algorithm. If the error rate is 0, then this illustrates that the experimental results are the same with the ground truth.

4.1 Synthetic datasets

Two different synthetic datasets are used for testing algorithms. Synthetic dataset 1 (SYN1) is composed of 10 time steps. It consists of 128 nodes and 4 commnities at each time step. The average degree of each node is 16 and each node connects to other z nodes in the network. In order to reflect the dynamic of the network, from the second time step, 3 members from each community are randomly selected and put to the other 3 communities. At the same time, the edges connected with these 3 nodes are also randomly put into the other 3 communities. The connecting possibility inside the community is higher than the connecting possibility outside the community. The value of parameter z depends on the fuzzy degree of the network.

The comparative average error value of four algorithms on SYN1 is shown in Fig. 2. Figure 2a is the result for $z = 3$ and Fig. 2b is the result for $z = 5$. It is obviously seen that MBBOD and MDPSO perform better than DYNMOGA at most time steps. MDBA performs best because it nearly gets zero error when $z = 3$ and $z = 5$.

Next, we use box plot to depict the distribution of error values for four algorithms with $z = 3$ and $z = 5$ in order to show the statistical decentralization of data at each time step. As seen from Figs. 3 and 4, it is worth noticing that even there are some outliers at each time step for those algorithms, the performance of MDBA is superior to other three algorithms, mostly the error value it gets is zero, and the outliers of MDBA is smaller than those of other algorithms.

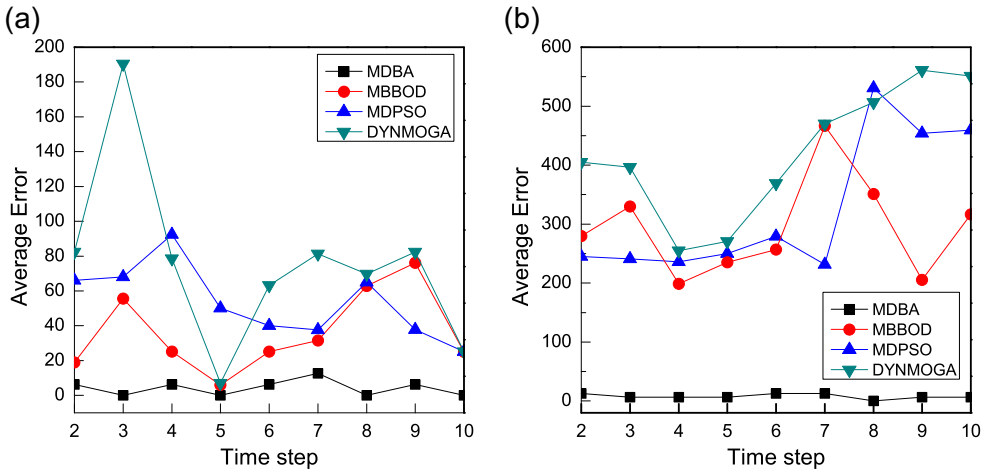

Fig. 2 Comparative average error value of four algorithms on SYN1 **a** $z = 3$ **b** $z = 5$

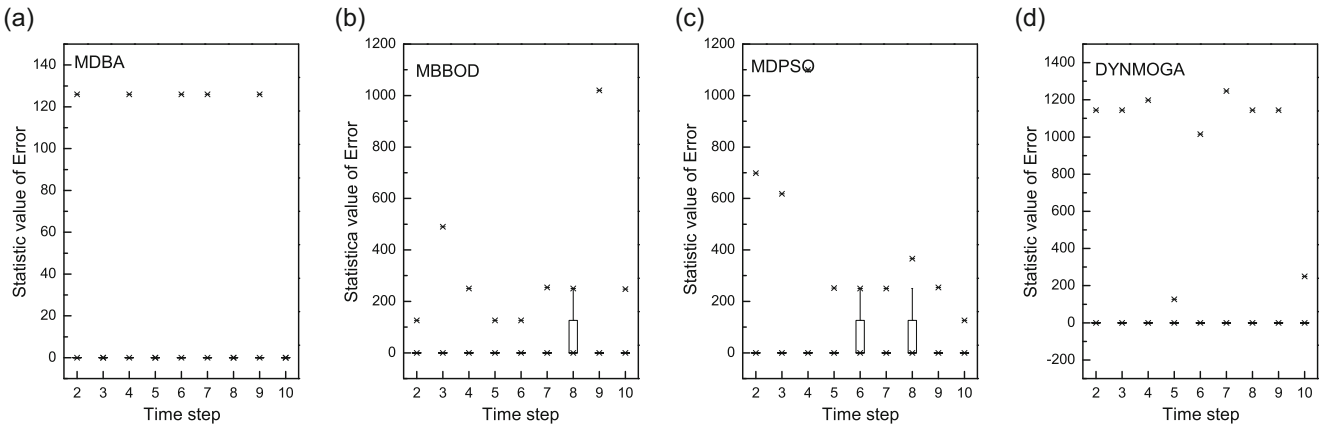

Fig. 3 Statistic value of error of four algorithm on Syn1 of $z = 3$ **a** MDBA **b** MBBOD **c** MDPSO **d** DYNMOGA

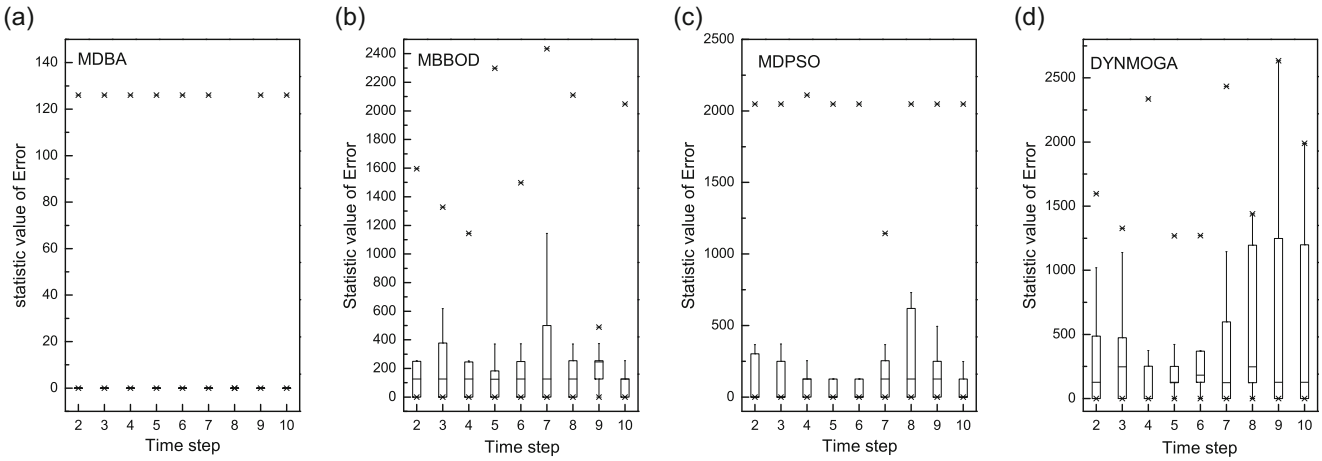

Fig. 4 Statistic value of error of four algorithm on Syn1 of $z = 5$ **a** MDBA **b** MBBOD **c** MDPSO **d** DYNMOGA

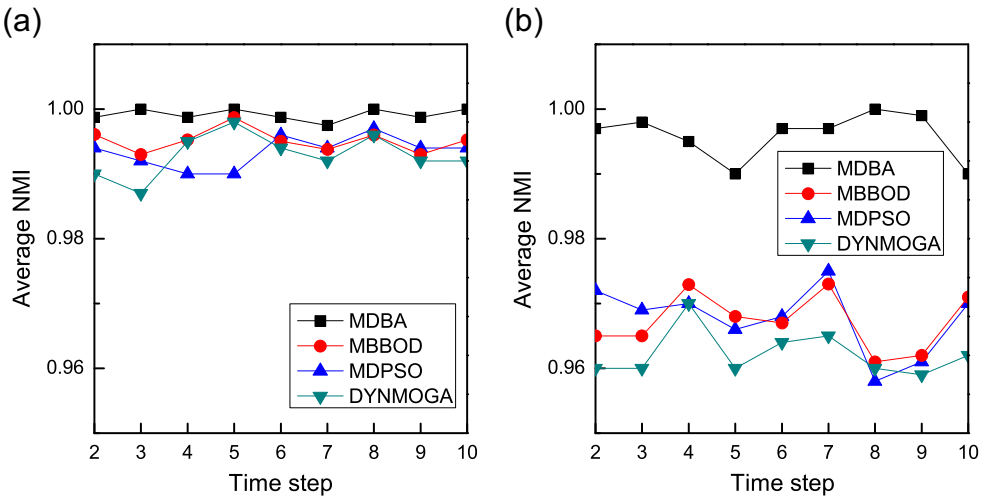

Fig. 5 Comparative average NMI of four algorithms on SYN1 **a** $z = 3$ **b** $z = 5$

Figure 5 shows comparing result for the average NMI obtained by four algorithms. Figure 5a is the result for $z = 3$ and Fig. 5b is the result for $z = 5$. It is easy to find that the performance of MDBA is superior to DYNMOGA, MBBOD and MDPSO. The results obtained by MBBOD,MDPSO and DYNMOGA are similiar when $z = 3$, the average NMI values of MDPSO is lower than DYNMOGA when $z = 3$ at some time steps. Furthermore, the NMI value of MDBA is about 0.99 when $z = 5$, but the NMI value of other three algorithms are between 0.96 and 0.98 when $z = 5$. It means when z is large and the network is fuzzy, our algorithm can discover the community structure more accurately.

In order to show the statistical decentralization of data at each time step, the box plot depicting the distribution of NMI values for four algorithms with $z = 3$ and $z = 5$ are shown in Figs. 6 and 7 separately. From Fig. 6, it is seen that MDBA performs best, it nearly gets the true partition results at each time step. Figure 7 clearly shows that the NMI value distribution of MDBA is more intensive than other three algorithms, and outliers of MDBA get higher NMI value at each time step than those of MBBOD, MDPSO and DYNMOGA.

The difference between synthetic dataset 2 (SYN2) and synthetic dataset 1 is that the number of communities in the synthetic dataset 2 is changing over time. The synthetic dataset 2 has 256 nodes at each time step, it includes 4 communities at the first time step, and each community has 64 nodes. From second time step to fifth time step, randomly select 8 nodes from each community, use these 8 nodes to construct a new community, and for the other time steps, those 8 nodes return to their original communities. In this way, the number of communities at ten timestamps is 4,5,6,7,8,8,7,6,5,4. Moreover, the average degree of each node in one cluster is equal to the half size of the cluster. At each time step, 16 nodes are deleted arbitrarily, and then 16 new nodes are added randomly in order to guarantee the

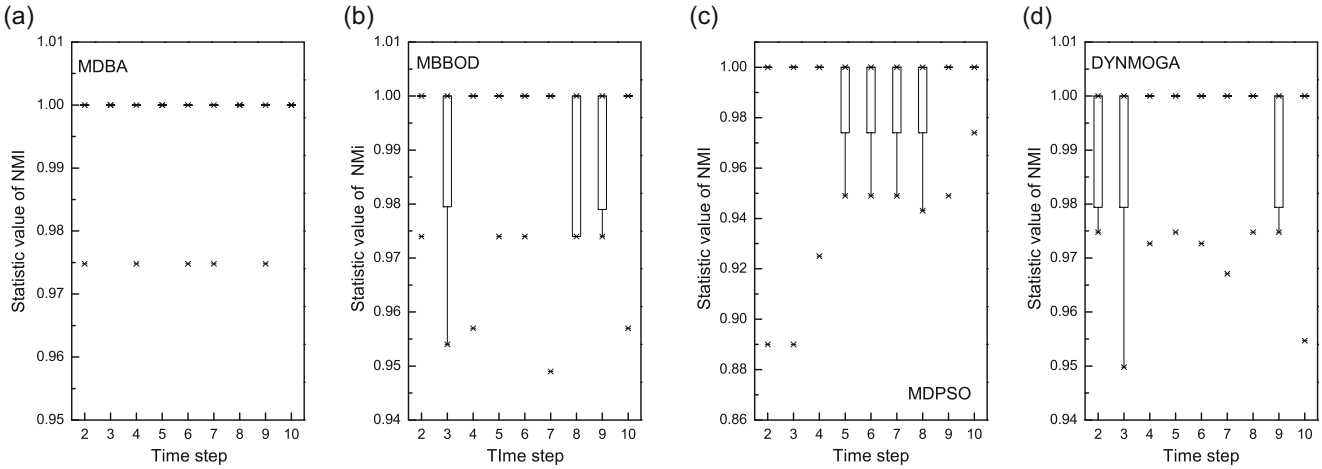

Fig. 6 Statistic value of NMI of four algorithm on Syn1 of $z = 3$ **a** MDBA **b** MBBOD **c** MDPSO **d** DYNMOGA

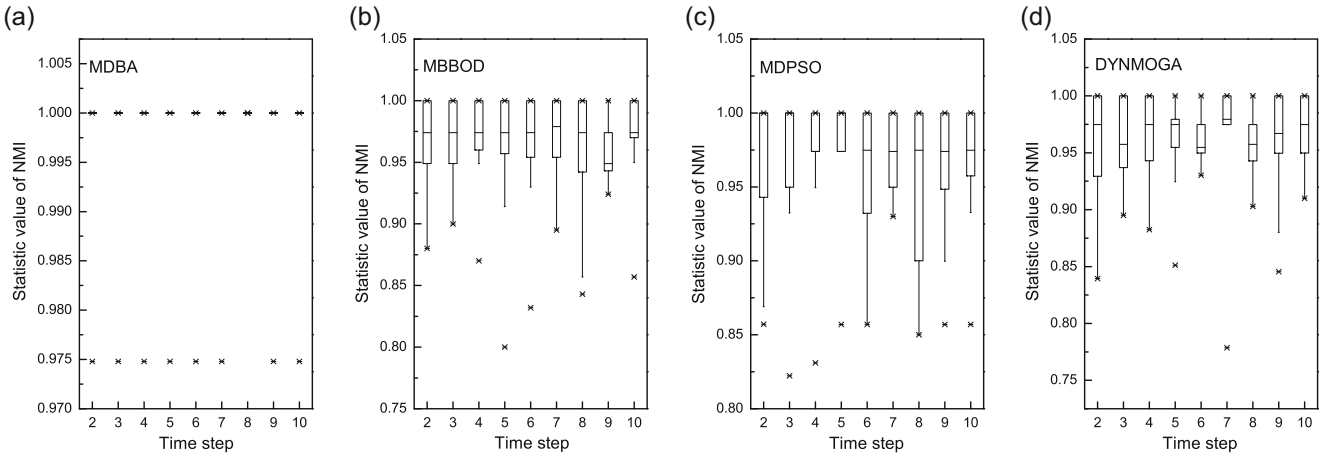

Fig. 7 Statistic value of NMI of four algorithm on Syn1 of $z = 5$ **a** MDBA **b** MBBOD **c** MDPSO **d** DYNMOGA

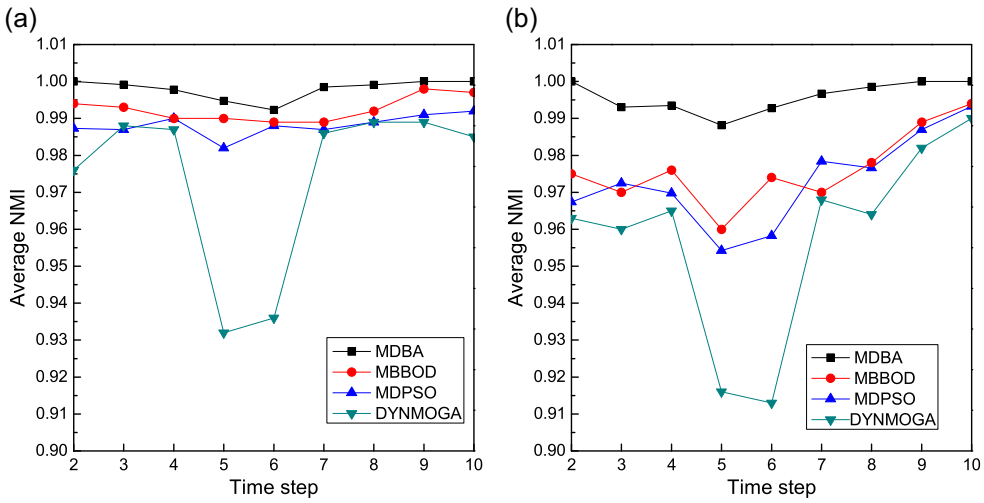

Fig. 8 Comparative average NMI value of four algorithms on SYN2 **a** $z = 3$ **b** $z = 5$

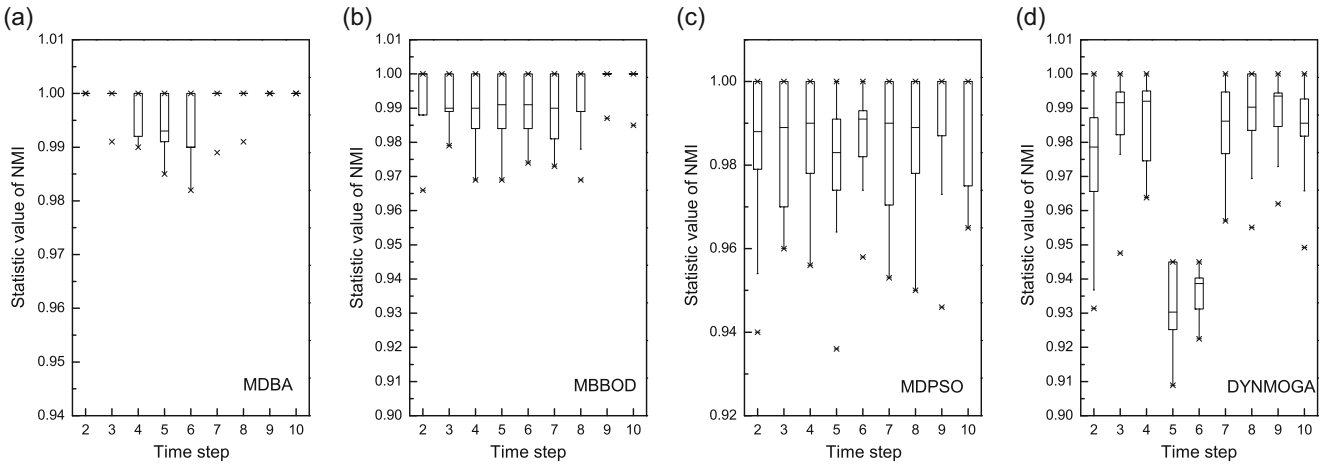

Fig. 9 Statistic value of NMI of four algorithm on SYN2 of $z = 3$ **a** MDBA **b** MBBOD **c** MDPSO **d** DYNMOGA

number of nodes in the network is fixed. The meaning of the parameter z in synthetic dataset 2 is the same as in the synthetic dataset 1, the bigger the value of z becomes, the more confusing the network structure is and the more difficult to detect partitions of network. It simulate the splitting and merging of communities.

We compare the algorithm proposed with DYNMOGA, MBBOD and MDPSO in this paper. The comparative average NMI value of four algorithms on SYN2 is presented in Fig. 8. The accuracy at each time step when $z = 3$ is almost approaching to 100% which is much higher than MBBOD MDPSO and DYNMOGA algorithm. The NMI value of DYNMOGA and MDPSO at time 5 and time 6 is much lower than MDBA and MBBOD. When $z = 5$ the values obtained by ours are above 0.99 which are still higher than other algorithms. The values obtained by MDPSO algorithm are between MBBOD and DYNMOGA. So we can get the conclusion that the results on the average values after 20 independent runs show that the partition results obtained by our algorithm are very closely approaching the real results. Generally speaking, MDBA could catch the merging and splitting of communities.

Figures 9 and 10 show the box plot to illustrate the distribution of the value of NMI on SYN2 datasets when $z = 3$ and $z = 5$. From Fig. 9 it can be seen clearly that the distribution of NMI values obtained by MDBA performs best, it nearly gets the true partition results at each time step. From Fig. 10 the distribution of MDBA is more intensive than other three algorithms, the median, lower quartile, upper quartile even though outliers of MDBA get higher NMI value at each time step than those of MBBOD,MDPSO and DYNMOGA. The outlier NMI value at some time steps

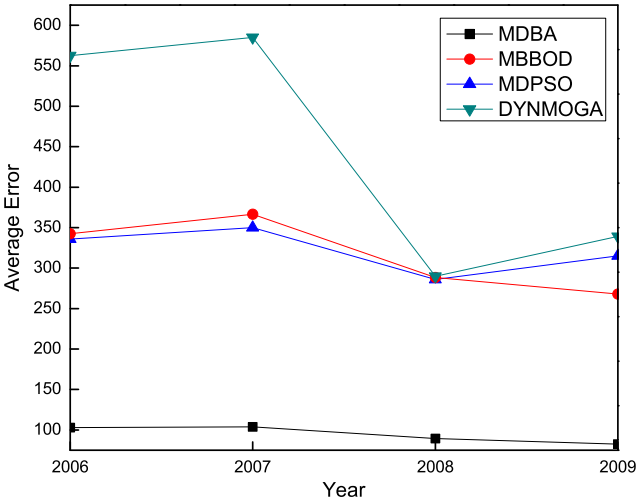

Fig. 11 Comparative average error value of four algorithms on football dataset

obtained by MDBA is still higher than DYNMOGA and MDPSO.

4.2 Real datasets

The Football dataset is the National Collegiate Athletic Association (NCAA) Football Division 1–A games, it can be downloaded in http://www.jhowell.net/cf/scores. The nodes in the network represent the teams, and the edges represent the regular season games between the two teams. We select 119 teams with 12 conferences from 2005 to 2009 as the dynamic datasets over 5 time steps.

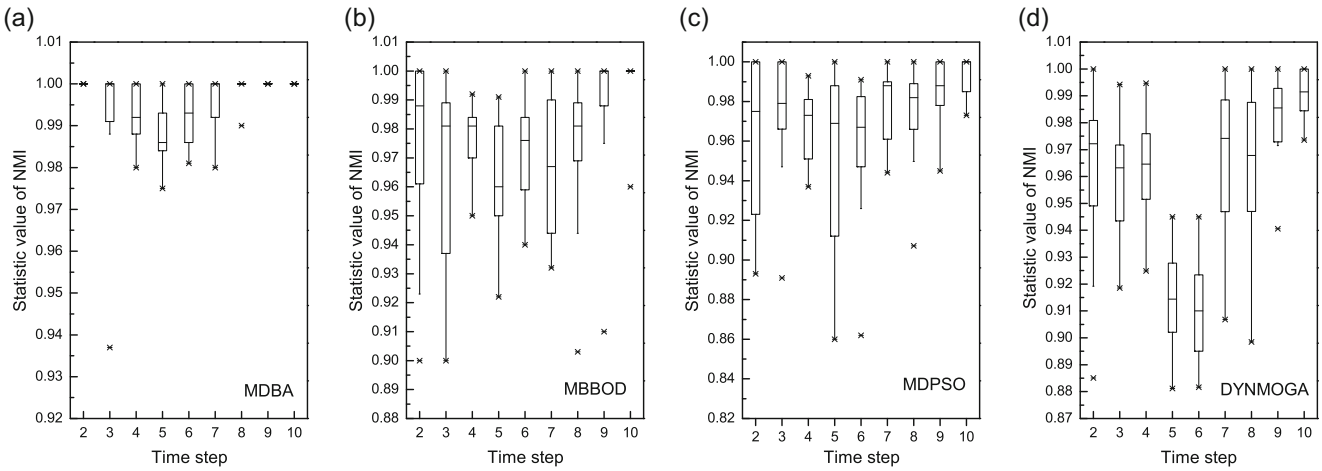

Fig. 10 Statistic value of NMI of four algorithm on SYN2 of $z = 5$ **a** MDBA **b** MBBOD **c** MDPSO **d** DYNMOGA

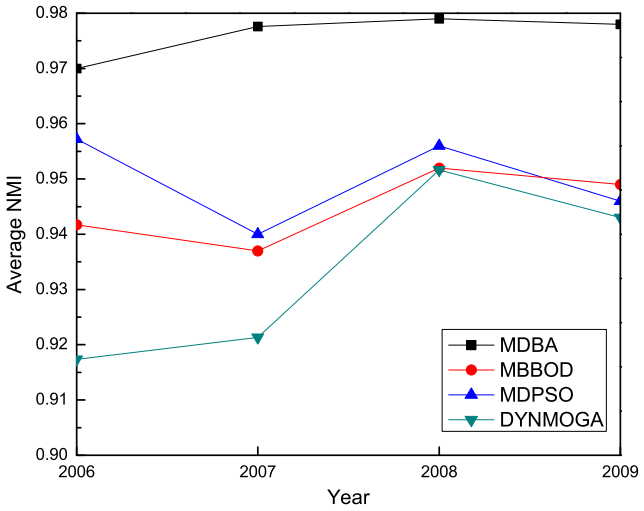

Fig. 12 Comparative average NMI value of four algorithms on football dataset

Here, NMI is employed to evaluate the accuracy of algorithms. Figure 11 shows the average error values of year from 2006 to 2009 obtained by four algorithms. Comparing with DYNMOGA, MDPSO and MBBOD algorithm, it is obvious to realize that MDBA algorithm has lowest error. The error of MBBOD is close to that of MDPSO. Figure 12 shows the average NMI values of year from 2006 to 2009 obtained by four algorithms. The accuracy at each time step is almost approaching to 98% which is much higher than other three algorithms.

Next, we visually the community partition with the best NMI value. The best results(0.981) founded by MDBA in 2009 of 20 runs is shown in Fig. 13. It is drawn by using Pajek software [28]. As shown in Fig. 13, MDBA can find 12 different communities. Almost all teams can be classified into true communities to which they really belong. It can be clearly seen that only two teams, Washington state and army, are mistakenly divided to the conferences Independent and MAC respectively. It is difficult to cluster the indepedent teams,

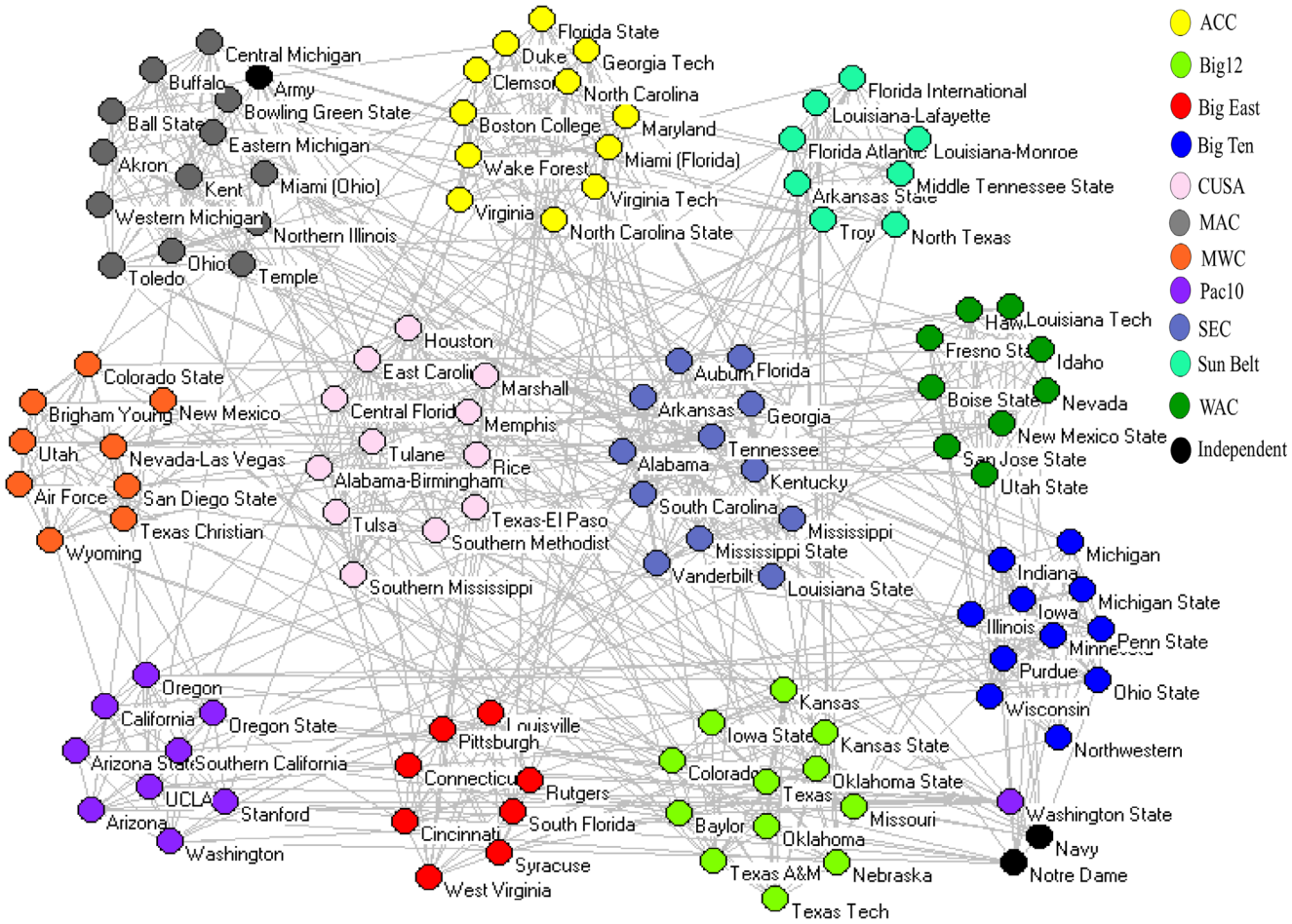

Fig. 13 Football partition result obtained by MDBA in 2009 year

because the indepedent teams (black circle)have more games with the teams in other communities than between them. Here we cluster 2 of them and Washington state into a community. In a word, this partition result is also acceptable.

5 Conclusion and future work

Community detection in dynamic networks is a promising direction worth further investigation and expansion. In order to improve the accuracy of the community detection, a multiobjective discrete bat algorithm is proposed in this paper. MDBA optimizes modularity and NMI as two objective functions simultaneously. The bat location updating strategy is redefined in discrete form. In addition, turbulence operator and mutate operator is incorporated to guarantee the diversity of the population. The non-dominated sorting and crowding distance mechanism are used to keep good solutions during the generation. A comparison of simulation results reveals better solution quality and computational efficiency of the proposed algorithm. In future, we will aim at designing new method to detect overlapping community structures in dynamic networks.

Acknowledgements We would like to thank the anonymous referees for their many valuable suggestions and comments. This work is supported by the National Natural Science Foundation of China (Grant No. 61373123), Key Development Program for Science and Technology of Jilin Province, China (Grant No.20150414004GH) China Postdoctoral Science Foundation (Grant No.2017M621210).

References

1. Fortunato S (2010) Community detection in graphs. Phys Rep 486:75–174
2. Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proc Natl Acad Sci 99:7821–7826
3. Newman MEJ (2011) Communities, modules and large-scale structure in networks. Nat Phys 8:25–31
4. Holme P, Saramaki J (2012) Temporal networks. Phys Rep 519:97–125
5. Song A, Li M (2016) Community detection using discrete bat algorithm. Int J Comput Sci 43(1):37–43
6. Wang C, Pan Y (2015) Discrete bat algorithm and application in community detection. The Open Cybernet Syst J 9:967–972
7. Hopcroft J, Khan O, Kulis B, Selman B (2004) Tracking evolving communities in large linked networks. In: Proceedings of the National Academy of Sciences of the United States of America, 101(1),5249–5253
8. Greene D, Doyle D, Cunningham P (2010) Tracking the evolution of communities in dynamic social networks. In: International conference on advances in social networks analysis and mining (ASONAM), Odense. IEEE, 176–183
9. Sun H, Huang J (2004) IncOrder: Incremental density-based community detection in dynamic networks. Knowl Based Syst 72:1–12
10. Chakrabarti D, Kumar R, Tomkins A (2006) Evolutionary clustering. In: Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining, Philadelphia. ACM, 554–560
11. Chi Y, Song XD, Zhou D, Hino K, Tseng BL (2007) Evolutionary spectral clustering by incorporating temporal smoothness. In: Proceedings of the 13th International Conference on Knowledge Discovery and Data Mining, 153-162
12. Tantipathananandh C, Berger-Wolf T, Kempe D (2007) A framework for community identification in dynamic social networks. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 717-726
13. Lin YR, Chi Y, Zhu S, Sundaram H, Tseng BL (2008) Facetnet: a framework for analyzing communities and their evolutions in dynamic networks. In: Proceedings of the 17th international conference on World Wide Web, 685–694
14. Folino F, Pizzuti C (2010) A multiobjective and evolutionary clustering method for dynamic networks. In: Proceedings of the International Conference on Advances in social networks analysis and mining, 256–263
15. Zhou X, Liu YY, Li B (2015) Multiobjective biogeography based optimization algorithm with decomposition for community detection in dynamic networks. Physical A 436:430–442
16. Gong MG, Zhang LJ, Ma JJ, Jiao LC (2012) Community detection in dynamic social networks based on multiobjective immune algorithm. J Comput Sci Technol 27(4):455–467
17. Ma JJ, Liu J, Ma W, Gong MG, Jiao LC (2014) Decomposition-based multiobjective evolutionary algorithm for community detection in dynamic social networks . Sci World J 2014:402345
18. Folino F, Pizzuti C (2014) An evolutionary multiobjective approach for community discovery in dynamic networks. IEEE Trans Knowl Data Eng 26(8):1838–1852
19. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197
20. Yang XS (2010) A new metaheuristic bat inspired algorithm. In: Nature inspired cooperative strategies for optimization, studies in computational intelligence, Springer Berlin, 284, 65-74
21. Yang XS, Gandomi AH (2012) bat algorithm: A novel approach for global engineering optimization. Eng Comput 29(5):464–483
22. Xu H, Bao ZR, Zhang T (2017) Solving dual flexible job-shop scheduling problem using a Bat Algorithm. Adv Production Eng Manag 12(1):5–16
23. Jeyasingh S, Veluchamy M (2017) Modified bat algorithm for feature selection with the wisconsin diagnosis breast cancer (WDBC) dataset. Asian Pac J Cancer Prev 18(5):1257–1264
24. Yang XS (2011) Bat algorithm for multiobjective optimization. Int J Bio Inspired Comput 3(5):267–274
25. Clauset A, Newman MEJ, Moore C (2004) Finding community structure in very large networks. Phys Rev E 70:066111
26. Danon L, Diaz-Guilera A, Duch J, Arenas A (2005) Comparing community structure identification, Journal of Statistical Mechanics: Theory and Experiment, P09008
27. Li Z, He L, Li Y (2016) A novel multiobjective particle swarm optimization algorithm for signed network community detection. Appl Intell 44:621–633
28. Nooy WD, Mrvar A, Batagelj V (2005) Exploratory Social Network Analysis with pajek. Cambridge University Press, New York

Xu Zhou received the M.S. degree in computer application from Northeast Normal University, China in 2013, and the Ph.D. degree in College of Computer Science from Jilin University, China in 2016. She is currently a post doctor in College of Communication Engineering in Jilin University. Her research interests focus on data mining and network analysis.

Xiaohui Zhao received the B.E. and M.S. degrees in electrical engineering from Jilin University of Technology, China, in 1982 and 1989, respectively, and the Ph.D. degree in applied mathematics and control theory from Universite de Technologie de Compiegne, France in 1993. He did his postdoctoral research in the Institute of Automatic Control, Southeast University, China during 1994–1996, and he has been a Senior Visiting Scholar for half a year to the Laboratoire d' Informatique, Universite de Pierre et Marie Curie, France in 2006. He is currently a Professor of Communication Engineering, Jilin University. He has published over 100 papers of journals and international conferences, two books in the fields of cognitive radio and signal processing. Dr. Zhao is a Member of the IEEE Communication Society. Currently, he serves as an Editor of the Chinese Journal of Communications, and Editor of the Chinese Journal of Signal Processing, and an Editor of the Journal of China Universities of Posts and Telecommunications. His research interests include wireless communication, cognitive radio, and adaptive signal processing.

Yanheng Liu received MSc and Ph.D. degrees in computer science from Jilin University, People's Republic of China. He is currently a professor in Jilin University, People's Republic of China. His primary research interests are in network security, network management, mobile computing network theory and applications, etc. He has co-authored over 120 research publications in peer reviewed journals and international conference proceedings of which one has won "best paper" awards. Prior to joining Jilin University, he was visiting scholar with University of Hull, England, University of British Columbia, Canada and Alberta University, Canada.