2010 22nd International Conference on Tools with Artificial Intelligence

Genetic Algorithm with Local Search for Community Mining in Complex Networks

Di Jin, Dongxiao He, Dayou Liu
College of Computer Science and Technology
Jilin University
Changchun, China
{jindi.jlu, hedongxiaojlu}@gmail.com;
liudy@jlu.edu.cn

Carlos Baquero
CCTD/DI
University of Minho
Braga, Portugal
cbm@di.uminho.pt

***Abstract*—Detecting communities from complex networks has triggered considerable attention in several application domains. Targeting this problem, a local search based genetic algorithm (GALS) which employs a graph-based representation (LAR) has been proposed in this work. The core of the GALS is a local search based mutation technique. Aiming to overcome the drawbacks of the existing mutation methods, a concept called marginal gene has been proposed, and then an effective and efficient mutation method, combined with a local search strategy which is based on the concept of marginal gene, has also been proposed by analyzing the modularity function. Moreover, in this paper the percolation theory on ER random graphs is employed to further clarify the effectiveness of LAR presentation; A Markov random walk based method is adopted to produce an accurate and diverse initial population; the solution space of GALS will be significantly reduced by using a graph based mechanism. The proposed GALS has been tested on both computer-generated and real-world networks, and compared with some competitive community mining algorithms. Experimental result has shown that GALS is highly effective and efficient for discovering community structure.**

Keywords-complex network; community mining; network clustering; genetic algorithm; local search

I. INTRODUCTION

Many complex systems in the real world exist in the form of networks, such as social networks, biological networks, Web induced networks, etc., which are also often classified as complex networks. Complex network analysis has been one of the most popular research areas in recent years due to its applicability to a wide range of disciplines [1, 2, 3]. While a considerable body of work addressed basic statistical properties of complex networks such as the existence of "small world" structuring [1] and the presence of "power laws" in the link distribution [2], another property that has attracted particular attention is that of "community structure": the nodes 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 [3]. The community mining problem (CMP), which is also called network clustering problem, is to detect and interpret community structures from various complex network data sets. So far there are lots of practical application problems which can be modeled as CMP, such as terrorist organization recognition, organization management, biological network analyzing, Web community mining, search engine design, link prediction, etc [4].

The research on community mining problems is of fundamental importance. At present, there are lots of community mining algorithms which have been developed. In terms of the basic strategies adopted by them, they mainly fall into two main categories: heuristic and optimization based methods.

The heuristic methods solve community mining problem based on some intuitive assumptions. For example, there are Girvan-Newman (GN) algorithm [3], Clique Percolation Method (CPM) [5], Finding and Extracting Communities (FEC) [6], Label Propagation Algorithm (LPA) [7], Community Detection with Propinquity Dynamics (CDPD) [8], Opinion Dynamics with Decaying Confidence (ODDC) [9], etc.

In contrast, the optimization based methods solve community mining problem by transforming it into a combinatorial optimization problem and trying to find an optimal solution for a predefined objective function, such as the network modularity (Q) [10] employed in several algorithms. For example: Fast Newman (FN) algorithm [11], Simulated Annealing (SA) algorithm [12], Iterated Tabu Search (ITS) [13], Modularity-Specialized Label Propagation Algorithm (LPAm) [14, 15], etc. As maximizing the modularity Q has been proven to be a nondeterministic polynomial time (NP)-complete problem [16], the above methods are also approximation algorithms. At present, genetic algorithm (GA) has been becoming a type of competitive method in the community mining area due to its effectiveness for solving NP-complete problems. Recently, two genetic representation strategies which are string-of-group encoding [17, 18, 19] and locus-based adjacency representation (LAR) [20, 21] are mainly employed by GA for solving community mining problems. However, string-of-group encoding is not suitable for a crossover operator in community mining application; LAR presentation is well-suited for traditional crossover operators, but designing an efficient and effective mutation operator is a new challenging work for this setting.

Addressing the drawbacks of current genetic algorithms for solving community mining problem, a local search based genetic algorithm (GALS) which employs modularity Q as objective function and LAR as genetic presentation is proposed in this paper. GALS firstly adopts Markov a random walk based method to generate initial population, and then it detects network community structure by iteratively executing the

1082-3409/10 $26.00 © 2010 IEEE
DOI 10.1109/ICTAI.2010.23

following three genetic operators: uniform crossover, local search based mutation and $\mu+\lambda$ selection. The major innovation in this paper is a local search based mutation method. With regard to the drawbacks of current mutation methods, a concept which is called marginal gene is newly proposed in this paper, and then an effective as well as efficient mutation method combined with local search strategy which is based on the concept of marginal gene is proposed by analyzing modularity Q. Moreover, the genetic operators which are employed by GALS make each LAR chromosome in the population correspond to a spanning subgraph of the complex network. Thus the solution space of GALS can be significantly reduced, which makes both the search efficiency and convergence rate of this algorithm expressively improved.

II. ALGORITHM

A. Problem Definition

In 2004, Newman and Girvan proposed an important quality metric for assessment of partitioning a network into communities, which is called network modularity or function Q [10]. Though this function Q suffers from resolution limit problems, as shown by Fortunato and Barthelemy [22], it still has been widely accepted by the scientific community [11-15, 17-21]. Our genetic algorithm GALS also employs modularity Q as objective function which is to be maximized.

The idea of network modularity is taken from the intuition that a network with community structure is different from a random network. Therefore, this function Q is defined as the difference between the fraction of edges that fall within communities and the expected value of the same quantity if edges fall at random without regard for the community structure.

Given an unweighted and undirected network $N = (V, E)$ and supposing that the nodes are divided into communities such that node i belongs to community $c_{r(i)}$ in which $r(i)$ denotes the label of node i, then function Q is defined as

$$Q = \frac{1}{2m}\sum_{ij}\left(\left(A_{ij} - \frac{k_i k_j}{2m}\right)\times\delta(r(i),r(j))\right), \qquad (1)$$

where $A = (A_{ij})_{n\times n}$ is the adjacency matrix of network N. $A_{ij} = 1$ if nodes i and j connect with each other, $A_{ij} = 0$ otherwise. The δ function $\delta(u,v)$ is equal to 1 if $u = v$ and 0 otherwise. The degree k_i of any node i is defined as $k_i = \sum_j A_{ij}$, and $m = \frac{1}{2}\sum_{ij} A_{ij}$ is the total number of edges in network N.

B. Genetic Representation

Algorithm GALS in this paper adopts the locus-based adjacency representation (LAR) which was proposed by [23]. At present, this representation (or encoding) schema was also employed by [24, 25] for multi-objective clustering problems and by [20, 21] for community mining problems. In this graph-based representation, an arbitrary individual (or chromosome) g in the population consists of n genes, in which each gene corresponds to a node in complex network N and n denotes the total number of nodes in this network. Each gene i can take an arbitrary allele value j in the range of $\{1, \ldots, n\}$, which can be interpreted as a link between nodes i and j existing in the corresponding graph G of individual g. This also means node i will be in a same community with node j in the network clustering solution denoted by this individual. The decoding process for a LAR individual is to identify all the components from graph G, and the nodes belonging to the same component are assigned to the same community. This decoding process can be done in a linear time as shown by [26]. The LAR representation is illustrated in Fig. 1.

In order to further clarify the effectiveness of LAR presentation for community mining problem, in this paper we try to analyze the characteristic of LAR presentation by employing percolation theory on ER random graphs. Erdös and Rényi proposed the classical ER random graphs model and the related analysis in percolation theory in 1960 [27]. They assume that, given a probability value $p\in(0\sim1)$, two arbitrary nodes in a random graph are connected under this probability p. Furthermore percolation theory is related to the percolation transition taking place at $p = 1/n$, where, as stated, p is the probability value that two nodes are connected by an edge and n is the total number of nodes in this graph. This also means that the threshold link probability of the appearance of a *giant component* in ER random graph is $p = 1/n$. It's obvious that any randomly generated LAR chromosome can be regarded as an ER random graph by the threshold link probability. Known from percolation theory on ER random graphs, starting from threshold link probability $p = 1/n$, the random graph will rapidly become a completely connected graph with the increase of p, while with the decrease of p there will rapidly appear many small connected components in this random graph. It's obvious that the link probability values which are greater or less than the threshold link probability value are all not suitable for a graph-based presentation, while $p = 1/n$ is fittest for it. Thus the LAR presentation used in this paper is very effective for our community mining application.

C. Population Initialization

Here we try to further analyze the idea of safe individuals proposed by [20, 21], and then give a Markov random walk based individual generation method which can produce safe as well as accurate and diverse initial individuals.

If an individual is randomly generated, some components in its corresponding graph G may be disconnected in the original complex network N, which also means G may be not a subgraph of N. For example, an individual could contain an allele value j in the ith position, which means there is a link between nodes i and j in its corresponding graph G, but there may be no connection existing between these two nodes in network N. However, in complex networks with community

(a) (b) (c)

Figure 1. An illustration for the locus-based adjacency presentation. (a) A complex network consisting of eleven nodes; (b) one out of many possible chromosomes; (c) the corresponding graph of this chromosome.

structure, there is an obvious intuition that any node should be in a same community with one of its neighbors. Thus the solution space of chromosomes can be reduced according to the above heuristics, which makes any individual in the population be a spanning subgraph of the original network N. Then the individual which satisfies the above condition is called a safe individual, the population which is composed of safe individuals is called a safe population, and the solution space of safe individuals is called a safe solution space. The whole solution space of LAR presentation is n^n, while the safe solution space is $\prod_{i=1}^{n} k_i$, where n is the total number of nodes and k_i is the degree of node i in network N. As most complex networks are sparse graphs, k_i can be regarded as a constant. Thus it's obvious that the safe solution space is much smaller than the whole solution space. Therefore if the search region of our algorithm GALS can be restricted in the range of the safe solution space, its search efficiency and convergence rate can be both significantly improved.

In order to improve the performance of algorithm GALS, we propose a Markov random walk based individual generation method (MRW) which is based on the natural features of community structure in complex networks. First of all, the related theory is given as follows.

In a network, let p_{ij} be the probability that an agent freely walks from any node i to its neighbor node j within one step, which is also called the transition probability of one-step random walk. In terms of the adjacency matrix of N, $A = (a_{ij})_{n\times n}$, p_{ij} is defined by

$$p_{ij} = \frac{a_{ij}}{\sum_r a_{ir}}. \tag{2}$$

From the view of a Markov random walk, when a complex network has community structure, a random walk agent should be found it difficult to move outside its own community boundary, whereas it should be easy for it to reach other nodes within its community, as link density within a community should be high, by definition. In other words, the probability for remaining in the same community, that is, a random walk agent starts from any node and stays in its own community, should be greater than that for going out to a different community. Based the above idea, we make arbitrary gene i in a chromosome select its allele value j in the range of $\{1, \ldots, n\}$ by using one-step transition probability p_{ij} in algorithm MRW. It's obvious that the individuals generated by MRW are not only safe but also accurate and diverse, which can improve the performance of our genetic algorithm GALS.

D. Selection and Crossover Operator

Selection operator plays a role of global search in genetic algorithm (GA). In order to retain the fittest individuals from each generation and improve the convergence speed of GA, $\mu+\lambda$ selection strategy [28] which is preferred by GA for solving combinatorial optimization problems is adopted by this paper. The process of $\mu+\lambda$ selection can be described as follows. Let the size of parent population be μ, and λ offspring be generated from randomly chosen parents, then we single out μ best individuals among parents and offspring as the population of next generation.

As another global search operator in GA, uniform crossover (UC) [29] is adopted in this paper. Given two randomly chosen parents A and B, and a randomly generated binary vector v, uniform crossover then selects the genes where v is 1 from parent A, and selects the genes where v is 0 from parent B, and then combines the genes to form a new child C. Mathematically speaking, there is $C = A.*v + B.*(1-v)$, where (.*) denotes array multiplication. Lets consider that parents A and B are both safe individuals, which means that if a gene i contains a value j, the edge $<i, j>$ will exist in the initial network N. Any gene i containing a value j in child C comes from one of its two parents, thus this child is forcibly a safe individual. Therefore we say that uniform crossover will not violate the safety of the population in GA. Furthermore, we choose the uniform crossover in favor of one-point or two-point crossover because it is unbiased with respect to the ordering of genes and can generate any combination of alleles from the two parents within a single crossover operation.

E. Mutation Operator

Mutation operator is the most important part in this paper. For community mining problems, we first introduce a concept called *marginal gene* with regard to the drawbacks of existing mutation methods, and then we propose an efficient and effective local search based mutation algorithm.

1) Marginal gene: Recently, a random mutation strategy was adopted as local search operator in most genetic algorithms which employed LAR presentation [20, 21, 24, 25]. However this type of mutation operator is not well-suitable for community mining problems.

In the opinion of Guimera and Amaral, when solving community mining problems, it's an effective method to generate a new candidate solution by continuously executing the following three types of operations on current candidate solution, which includes moving single nodes from one community to another, merging multi-communities and splitting single communities [12]. In genetic algorithms, the crossover operator is regarded as a macroscopic operation on individuals, while the mutation operator is regarded as a microcosmic operation on individuals. Thus, in a genetic algorithm for solving community mining problems, if the crossover operator can achieve its global search function by merging and splitting communities, and the mutation operator can achieve its local search function by moving single nodes between communities, this genetic algorithm can express a strong ability for searching. Each individual in this paper corresponds to a graph, and each component in the graph corresponds to a community. Thus it's obvious that the uniform crossover in this paper can effortlessly achieve its function of merging and splitting communities, and then the mutation operator in this section should effectively achieve its function of moving single nodes between communities. However, our study shows that traditional mutation operators [20, 21, 24, 25] often result in merging or splitting communities, which make it not amenable to effectively achieve the local search function, thus leading to an overall inefficacy of genetic algorithm. For example, in an individual (or chromosome) g, let nodes i and j belong to different communities (or components), and gene i select the allele value j by a single mutation operation, then this mutation operation will very likely lead to the combination of

these two communities, which also means the two components denoted by nodes i and j may be merged into a bigger one by building a new link <i, j> between them. It's obvious this situation is undesirable. Our research for this problem is presented as follows.

Definition 1. (Marginal Gene) Given arbitrary LAR chromosome g, if the allele values of all genes in g are not equal to j, gene j is called a marginal gene in g, which is also called a marginal node.

An arbitrary LAR chromosome g corresponds to a directed graph G. All nodes in G will not point at marginal nodes known from Definition 1. Thus a single mutation operation on a marginal node can implement this node's movement from one community to another, while it will not result in merging or splitting communities. It's obvious that the mutation operator can successfully achieve its local search function with the aid of marginal nodes.

Now we have to pay attention to the proportion of marginal genes appearing in a LAR chromosome. Here we first research the case for a randomly generated chromosome g, and then generalize it to a more general case. Let there be n genes in g, then the probability that an arbitrary gene i takes some value j is $p = 1/n$; on the contrary, the probability that gene i doesn't take value j is $1-p$; and then the probability that all genes in g don't take value j is $\beta = (1-1/n)^n$. Therefore, the probability that the marginal genes appear in chromosome g should be β. By mathematical analysis we know that, $\beta(n)$ is a monotone increasing function, and $\lim_{n\to+\infty}\beta(n) = 1/e \approx 0.3679$, where e is the Napierian base. Because the total number of nodes in almost all concrete instantiations of complex networks is greater than 10, and $p(10) = 0.3487$, thus the proportion of marginal genes appearing in a chromosome g should be $\beta(n) \in (0.3487\sim0.3679)$, $n > 10$. Moreover, our experiments show that the above conclusion is also fit for more general chromosomes in a universal situation. In this sense, if we execute mutation operations on all marginal genes of a chromosome, it's equivalent to that of executing mutation operations on this chromosome by mutational rate β.

2) Local search based mutation algorithm: Based on Definition 1, this section tries to propose a fast and effective local search based mutation algorithm (LSMA) aiming to marginal nodes by a focused study on the community mining problem, which is totally different from the existing mutation methods.

In order to effectively implement single nodes' movements between communities in algorithm LSMA, here we firstly give some theoretical analyses on our objective function Q from each node's local view. We convert (1) to (3), which takes function Q as the sum of function f of all nodes. It's obvious that from each node's local point of view, function f can be regarded as the difference between the number of edges that fall within communities and the expected number of edges that fall within communities. Therefore, function f of each node can measure whether a network division indicates a strong community structure from its local perspective. Some Propositions and Theorems on function f are given as follows.

$$Q = \frac{1}{2m}\sum_i f_i, \quad f_i = \sum_{j\in c_{r(i)}}\left(A_{ij} - \frac{k_i k_j}{2m}\right) \tag{3}$$

Proposition 1. For $\forall\, i \in V$, the local function f_i of any node i in a complex network is only related to its own community $c_{r(i)}$.

Proof. Proposition 1 is obvious, according to (3).

Theorem 1. For $\forall\, i \in V$, if the label of node i changes under the condition that the labels of all other nodes don't change, function Q of a complex network is monotonically increasing with function f_i.

Proof. Given a network $N = (V, E)$ and its community structure C. For $\forall\, i \in V$, let the label of node i be varied from $r(i)$ to $r(j)$, which makes the community structure become C'. Given $r(i) \neq r(j)$, in community structure C', the original community of node i becomes $c'_{r(i)} = c_{r(i)} - \{i\}$, and its current community becomes $c'_{r(j)} = c_{r(j)} \cup \{i\}$.

Known from (3), if the community of any node changes, its own function f will also change. Therefore the variety of the label of node i will result in the variety of function f of all nodes in node set $c = c_{r(i)} \cup c_{r(j)}$. We divide the nodes in c into three categories, and give the equation about the variety of function f for the nodes in each category respectively.

1. For $\forall\, s \in c'_{r(i)}$, the variety of its function f_s which is defined as Δf_s is given by

$$\begin{aligned}\Delta f_s &= f_s(C') - f_s(C)\\ &= \sum_{t\in c'_{r(i)}}\left(A_{st} - \frac{k_s k_t}{2m}\right) - \sum_{t\in c_{r(i)}}\left(A_{st} - \frac{k_s k_t}{2m}\right)\\ &= -\left(A_{si} - \frac{k_s k_i}{2m}\right)\end{aligned} \tag{4}$$

2. For $\forall\, p \in c_{r(j)}$, the variety of its function f_p which is defined as Δf_p is given by

$$\begin{aligned}\Delta f_p &= f_p(C') - f_p(C)\\ &= \sum_{q\in c'_{r(j)}}\left(A_{pq} - \frac{k_p k_q}{2m}\right) - \sum_{q\in c_{r(j)}}\left(A_{pq} - \frac{k_p k_q}{2m}\right)\\ &= A_{pi} - \frac{k_p k_i}{2m}\end{aligned} \tag{5}$$

3. For node i, the variety of its function f_i which is defined as Δf_i is given by

$$\begin{aligned}\Delta f_i &= f_i(C') - f_i(C)\\ &= \sum_{e\in c'_{r(j)}}\left(A_{ie} - \frac{k_i k_e}{2m}\right) - \sum_{e\in c_{r(i)}}\left(A_{ie} - \frac{k_i k_e}{2m}\right)\end{aligned} \tag{6}$$

Thus, the variety of function Q of the whole network, which is caused by the variety of the label of node i, can be

deduced as follows. Here the variety of Q-value is defined as ΔQ.

$$\begin{aligned}\Delta Q &= \frac{1}{2m}\left(\sum_{s\in c'_{r(i)}}\Delta f_s + \sum_{p\in c_{r(j)}}\Delta f_p + \Delta f_i\right)\\ &= \frac{1}{2m}\left(-\sum_{s\in c'_{r(i)}}\left(A_{si}-\frac{k_s k_i}{2m}\right)+\sum_{p\in c_{r(j)}}\left(A_{pi}-\frac{k_p k_i}{2m}\right)+\Delta f_i\right)\\ &= \frac{1}{2m}\left(\begin{array}{l}-\left(\sum_{s\in c_{r(i)}}\left(A_{si}-\frac{k_s k_i}{2m}\right)-\left(A_{ii}-\frac{k_i k_i}{2m}\right)\right)+\\ \left(\sum_{p\in c'_{r(j)}}\left(A_{pi}-\frac{k_p k_i}{2m}\right)-\left(A_{ii}-\frac{k_i k_i}{2m}\right)\right)+\Delta f_i\end{array}\right)\\ &= \frac{1}{2m}\left(\sum_{p\in c'_{r(j)}}\left(A_{pi}-\frac{k_p k_i}{2m}\right)-\sum_{s\in c_{r(i)}}\left(A_{si}-\frac{k_s k_i}{2m}\right)+\Delta f_i\right)\\ &= \frac{1}{2m}\left(\Delta f_i+\Delta f_i\right)=\frac{1}{m}\Delta f_i\end{aligned} \tag{7}$$

Then $Q(C')-Q(C)=\frac{1}{m}\left(f_i(C')-f_i(C)\right)$. As we can see, given $f_i(C')>f_i(C)$, there is $Q(C')>Q(C)$. Thus concluding this proof.

There is an obvious intuition in that any node in a complex network which has community structure should have a same label with one of its neighbors, thus any marginal node i can take an allele value j in the range of NS_i instead of V in our mutation algorithm, where NS_i is the neighbor set of node i and V is the whole node set of the network. Furthermore, as function Q of a complex network is monotonically increasing with any node's function f known from Theorem 1, we select node j from NS_i, which can maximize function f_i, as the allele value of marginal node i. Because we only consider marginal nodes here, after each mutation operation, node i will take the label of node j as its label, meanwhile, the labels of all other nodes will not change. Thus this also satisfies the condition requested by Theorem 1. Moreover, it's obvious that any safe individual is still be a safe one after executing the above mutation method. Here we design a fast and effective local search based mutation algorithm LSMA which is described in Fig. 2.

As we can see, according to the request of algorithm LSMA, the variety of any node's allele value denotes a node's movement from its original community (or component) to another one in the corresponding graph of chromosome g, while it can't result in merging or splitting communities. It's obvious that only marginal nodes can satisfy this request. On the contrary, the mutation operation on non-marginal nodes will necessarily lead to merging or splitting communities for g, thus they can't satisfy the request of algorithm LSMA. Therefore we can say that LSMA is totally designed for marginal nodes.

In order to further explain the effectiveness as well as efficiency of algorithm LSMA, some Propositions are given as follows.

```
Procedure  LSMA
Global: g /* a chromosome to be mutated */
Begin
1   C ←decode chromosome g // C is the community structure denoted by g
2   For i=1: n   // n is the total number of genes in g
3     If  node i is a marginal node
4       NS_i ←attain all the neighbors of node i
5       labels ← get unique labels from NS_i
6       max ← −∞
7       For each r∈ labels
8         f_i ←compute function f_i when node i takes label r
9         If  f_i > max
10          max ← f_i
11          label_i ← r
12        End
13      End
14      g(i) ← randomly select a node from NS_i whose label is label_i
                                          // update chromosome g
15      C(i)← label_i // update community structure C
16    End
17  End
End
```

Figure 2. The algorithm flow of LSMA.

Proposition 2. For arbitrary chromosome g which adopts LAR presentation, the value of fitness function Q will not decrease after executing LSMA algorithm.

Proof. Known from the algorithm flow of LSMA, any marginal gene i in chromosome g will necessarily take one of its neighbors j which can maximize its function f_i as its allele value after LSMA mutation. This means node i will move to the community of node j, while it will not result in merging or splitting communities, which also means the label of node i changes under the condition that the labels of all other nodes don't change. Known from Theorem 1, if the variety of a node's label makes its function f increase under the condition that the labels of all other nodes don't change, this variety will also cause the increase of Q-value of the complex network. Thus this mutation operation on any marginal gene in chromosome g will not make the Q-value of the complex network decrease. Thus concluding the proof..

In complex network N, let the total number of nodes be n, the average degree of all the nodes be k, and the average community size in the network clustering solution denoted by chromosome g in algorithm LSMA be c. It's obvious that c is much smaller than n. As most complex networks are sparse graphs, in order to raise efficiency, all the algorithms in this paper are implemented by using a sparse matrix, stored by a linked list. The time complexity analysis of algorithm LSMA is given as follows.

Proposition 3. The time complexity of algorithm LSMA is $O(cn)$.

Proof. The decoding step for a LAR chromosome can be done in a linear time, thus the time complexity of the 8th step is the highest in LSMA. Step 8 computes function f_i of each

marginal node i for all the labels of its neighbors. Because the labels of any node's neighbors are likely to overlap, the average number of evaluations of function f for each node i in step 8 can't be greater than k. Known from Proposition 1, the average time that function f is computed once can't be greater than c. The total number of marginal nodes is about βn in a LAR chromosome. Thus the time complexity of LSMA can't be greater than $O(\beta nkc)$. Furthermore, complex networks are always sparse graphs, which means k is a constant, and β is also a constant, thus the time complexity of LSMA can be also given by $O(cn)$.

F. Algorithm GALS

Based on the discussion in above sections, the description for algorithm GALS is given as Fig. 3.

Algorithm GALS firstly adopts a Markov random walk based individual generation method (MRW) to produce initial population, and then it detects network community structure by iteratively executing the following three genetic operators: uniform crossover, local search based mutation and $\mu+\lambda$ selection. It's obvious that: the individuals in initial population produced by MRW are not only safe but also accurate and diverse; our crossover operator can achieve its global search function by merging and splitting communities for LAR chromosome; our mutation operator can effectively achieve its local search function by cleverly moving single marginal nodes between communities for LAR chromosome; our selection operator can achieve its global search function by making the best individuals enter the next generation, mimicking on the Darwin's idea of "survival of the fittest". Moreover, as the population produced by MRW is a safe one, meanwhile, the two genetic operators (uniform crossover and local search based mutation) which can change the structure of LAR chromosome will not violate the safety of the safe population, thus algorithm GALS can detect network community structures in the range of the safe solution space which is much smaller than the whole solution space. Therefore the search efficiency and convergence speed of algorithm GALS can be also significantly improved from this approach.

The time complexity of GALS is given as follows. In complex network N, let the total number of nodes be n, and the average community size in all of the LAR chromosomes during the execution process of algorithm GALS be c. Note that this community size c doesn't denote the average community size of the final network clustering solution got by GALS.

Proposition 4. The time complexity of algorithm GALS is $O(cn)$.

Proof. It's obvious that the time complexity of the 4th step is the highest in GALS, and the time of other steps are all equal to or less than $O(n)$. The number of running algorithm LSMA in step 4 can't be greater than $L\lambda$, and the time of running LSMA once is $O(cn)$ from Proposition 3. Thus the time of running step 4 in GALS can't be greater than $O(L\lambda cn)$. Because all the parameters in GALS can be regarded as a constant, the time of running step 4 in GALS can be also given by $O(cn)$. Therefore the overall time complexity of GALS is $O(cn)$.

It's worth mentioning that, the time complexity of most community mining algorithms at present is not less than $O(n^2)$, even though the time of Newman's fast algorithm (FN) is $O(n^2)$, whereas the time of our algorithm GALS is $O(cn)$, in which c is much smaller than n.

```
Procedure  GALS
Input: N, L, μ, λ /* N denotes the network, L denotes the iteration number of GALS,
μ denotes the size of parent population, λ denotes the size of offspring population */
Output: C /* network community structure, or called network clustering solution */
Begin
1   P← produce initial population by running MRW for μ times
2   For i=1: L
3      P(new)← ∅
4      For j=1: λ
5         g← apply uniform crossover on two arbitrary parents from P
6         g← apply LSMA mutation on g // the mutation rate is about β
7         P(new)←P(new) ∪ {g}
8      End
9      P(u)←P ∪ P(new)
10     P← single out μ best individuals from P(u) // μ+λ selection strategy
11  End
12  I← select the fittest individual from P
13  C← decode chromosome I
End
```

Figure 3. The algorithm flow of GALS.

III. EXPERIMENTS

In order to quantitatively analyze the performance of algorithm GALS, we tested it by using both computer-generated and real-world networks. All experiments are done on a single Dell Server (Intel(R) Xeon(R) CPU 5130 @ 2.00GHz 2.00GHz processor with 4Gbytes of main memory on Microsoft Windows Server 2003 OS). Our programming environment is Matlab 7.3.

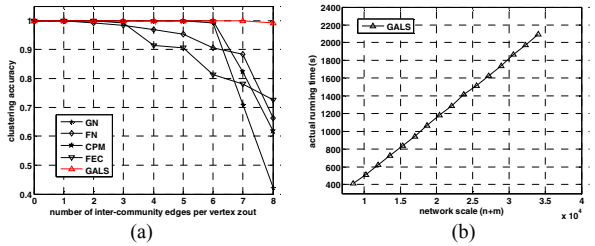
In algorithm GALS, there are three parameters: iteration number L, parent population size μ and offspring population size λ, which are all standard parameters in genetic algorithms. They can be set as: L = 200, μ = 80 and λ = 60 based on [20, 21, 24, 25, 28] as well as our own experience.

A. Computer-Generated Networks

To test the performance of algorithm GALS, we adopt random networks with known community structure, which has been used as benchmark datasets for testing community mining algorithms [3]. This kind of random network is defined as RN (a, s, k, z_{out}), where a is the number of communities, s is the number of nodes in each community, k is the degree of each nodes in the network, and each node has z_{in} edges connecting it to members of its community and z_{out} edges to members of other communities. As z_{out} increases from zero, community structures of networks become more diffused and the resulting networks pose greater and greater challenges to the community mining algorithms. Especially, a network doesn't have community structure when z_{out} is greater than 8 [3]. The clustering solution for a random network is perfect only if each node is assigned to the correct community, and no communities are further divided. This measure is used to calculate network clustering accuracy in this paper.

In order to investigate the performance of GALS in terms of clustering accuracy, this algorithm is compared with GN algorithm [3], Fast Newman (FN) algorithm [11], CPM algorithm [5] and FEC algorithm [6]. Algorithm GN and FN are both classic community mining algorithms which take function Q as objective function. Algorithm CPM and FEC are also very competitive algorithms at present, which belong to the class of heuristic methods. Fig. 4(a) shows the experimental results. Benchmark random network RN (4, 32, 16, z_{out}) is used in this experiment. In Fig. 4(a), y-axis denotes clustering accuracy, x-axis denotes z_{out}. For each z_{out}, for each algorithm, we compute the average accuracy through clustering 50 random networks. As we can see from this figure, our algorithm GALS significantly outperforms the other four algorithms in terms of clustering accuracy. Furthermore, as z_{out} becomes greater and greater, the superiority of our algorithm becomes more and more significant. Especially, when z_{out} equals 8, which means the number of within-community and between-community edges per nodes is the same, our algorithm can still correctly classify 99.22% of nodes into their correct communities, while the clustering accuracy of the other algorithms is low at this moment.

Computing speed is another very important criterion to evaluate the performance of community mining algorithms. Time complexity analysis for GALS has been given by Proposition 4 in Sec. II F, here we show the actual running time of GALS from experimental angle in order to further evaluate its efficiency. Random network RN (a, 100, 16, 5) containing 100a nodes and 1600a edges is adopted in this experiment. Though community structure of this type of network is known, the number of communities can still be changed by a. Fig. 4(b) shows the trend that the actual running time of GALS varies with the change of network scales. In this figure, y-axis denotes the actual running time (second), x-axis denotes the scales of networks (number of nodes + number of edges). As we can see, the running time of algorithm GALS is proportional to the scale of network under the condition that the average community size of actual network community structure is about constant. Therefore, This experiment can not only validate the correctness of Proposition 4 (the time complexity of GALS is $O(cn)$), but also shows that the average community size c of all the LAR chromosomes during the running process of GALS is proportional to the average community size in the actual network community structure. Actually, the sizes of communities in larger scale real-world networks are always much smaller than the scales of networks [30].



(a) (b)

Figure 4. Testing the performance of GALS on random networks. (a) Comparing GALS with GN, FN, CPM and FEC in terms of clustering accuracy; (b) The actual running time of GALS on networks with different scales.

B. *Real-World Networks*

As a further test on algorithm GALS, we applied it to seven widely used real-world networks which not only include small networks containing dozens of nodes, but also include large-scale networks containing tens of thousands of nodes. These actual networks may have different topological properties than those in the computer-generated networks. A simple description for them is given by Table I.

Because our algorithm GALS takes function Q as objective function, two algorithms (GN and FN) which also employ Q as objective function are selected from Sec. III A to be compared. Comparing GALS with these two algorithms on the networks described above, Table II shows the average experimental results over 50 runs. Note that empty cells correspond to computation time over 24 hours. As we can see, the clustering solution of algorithm GALS on real-world networks is also obviously better than that of algorithms GN and FN.

IV. CONCLUSION

A local search based genetic algorithm (GALS) which employs modularity Q as objective function and LAR as genetic presentation is proposed in this paper. GALS firstly adopts Markov random walk based method to produce initial population, and then it detects network community structure by iteratively executing the following three genetic operators: uniform crossover, local search based mutation and $\mu+\lambda$ selection. The proposed GALS is tested on both computer-generated and real-world networks, and compared with some competitive community mining algorithms. Experimental results demonstrate that GALS is highly effective as well as efficient at discovering community structure.

Our future work can be laid out as follows. We intend to further apply GALS in some interesting research areas, such as biological networks analysis, Web community mining, etc., and try to uncover and interpret significative community structure that is expected to be found on them.

ACKNOWLEDGMENT

This work was supported by National Natural Science Foundation of China under Grant Nos. 60873149, 60973088, National High-Tech Research and Development Plan of China

TABLE I. REAL-WORLD NETWORKS USED IN OUR EXPERIMENTS

Networks	V(G)	E(G)	Description
karate	34	78	Zachary's karate club [31]
dolphin	62	160	Dolphin social network [32]
polbooks	105	441	Books about US politics [33]
football	115	613	American College football[3]
jazz	198	5484	Jazz musicians network [34]
world	7,207	31,784	Semantic network [5]
arxiv	56,276	315,921	scientific collaboration networks [35]

TABLE II. COMPARING GALS (OVER 50 RUNS) WITH GN AND FN

Q-value	GN	FN	GALS
karate	0.4013	0.2528	0.4198
dolphin	0.4706	0.3715	0.5294
polbooks	0.5168	0.5020	0.5272
football	0.5996	0.4549	0.6045
jazz	0.4051	0.4030	0.4449
world	-	0.3821	0.4059
arxiv	-	0.5953	0.6126

under Grant No. 2006AA10Z245, Open Project Program of the National Laboratory of Pattern Recognition, and BRIDGING THE GAP Erasmus Mundus project of EU. We would like to thank Mark Newman for providing us with the source code of algorithms FN and GN, and some real-world network data.

REFERENCES

[1] D. J. Watts, and S. H. Strogatz, "Collective Dynamics of Small-World Networks," Nature, vol. 393, Jun. 1998, pp. 440-442, doi:10.1038/30918.

[2] A. L. Barabási, R. Albert, H. Jeong, and G. Bianconi, "Power-Law Distribution of the World Wide Web," Science, vol. 287, Mar. 2000, pp. 2115a, doi:10.1126/science.287.5461.2115a.

[3] M. Girvan, and M. E. J. Newman. "Community Structure in Social and Biological Networks," Proceedings of National Academy of Science, vol. 99, Jun. 2002, pp. 7821-7826, doi:10.1073/pnas.122653799.

[4] S. Fortunato, "Community Detection in Graphs," Physics Reports, vol. 486, Jun. 2010, pp. 75-174, doi:10.1016/j.physrep.2009.11.002.

[5] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek, "Uncovering the Overlapping Community Structures of Complex Networks in Nature and Society," Nature, vol. 435, Jun. 2005, pp. 814-818, doi:10.1038/nature03607.

[6] B. Yang, W. K. Cheung, and J. Liu, "Community Mining from Signed Social Networks," IEEE Transactions on Knowledge and Data Engineering, vol. 19, Sep. 2007, pp. 1333-1348, doi:10.1109/TKDE.2007.1061.

[7] U. N. Raghavan, R. Albert, and S. Kumara, "Near Linear-Time Algorithm to Detect Community Structures in Large-Scale Networks," Physical Review E, vol. 76, Sep. 2007, pp. 036106, doi:10.1103/PhysRevE.76.036106.

[8] Y. Zhang, J. Wang, Y. Wang, and L. Zhou, "Parallel Community Detection on Large Networks with Propinquity Dynamics," In Proc. the 15th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 09), ACM Press, Jun. 2009, pp. 997-1005, doi:10.1145/1557019.1557127.

[9] C. I. Morărescu, and A. Girard, "Opinion Dynamics with Decaying Confidence: Application to Community Detection in Graphs," arXiv:0911.5239v1, 2009.

[10] M. E. J. Newman, and M. Girvan, "Finding and Evaluating Community Structure in Networks," Physical Review E, vol. 69, Feb. 2004, pp. 026113, doi:10.1103/PhysRevE.69.026113.

[11] M. E. J. Newman, "Fast Algorithm for Detecting Community Structure in Networks," Physical Review E, vol. 69, Jun. 2004, pp. 066133, doi:10.1103/PhysRevE.69.066133.

[12] R. Guimera, and L. A. N. Amaral, "Functional Cartography of Complex Metabolic Networks," Nature, vol. 433, Feb. 2005, pp. 895-900, doi:10.1038/nature03288.

[13] Z. Lü, and W. Huang, "Iterated Tabu Search for Identifying Community Structure in Complex Networks," Physical Review E, vol. 80, Aug. 2009, pp. 026130, doi:10.1103/PhysRevE.80.026130.

[14] M. J. Barber, and J. W. Clark, "Detecting Network Communities by Propagating Labels under Constraints," Physical Review E, vol. 80, Aug. 2009, pp. 026129, doi:10.1103/PhysRevE.80.026129.

[15] X. Liu, and T. Murata, "Advanced Modularity-Specialized Label Propagation Algorithm for Detecting Communities in Networks," Physica A, vol. 389, Apr. 2010, pp. 1493-1500, doi:10.1016/j.physa.2009.12.019.

[16] U. Brandes, D. Delling, M. Gaertler, R. Goerke, M. Hoefer, Z. Nikoloski, and D. Wagner, "Maximizing Modularity is Hard," arXiv:physics/0608255, 2006.

[17] M. Tasgin, A. Herdagdelen, and H. Bingol, "Community Detection in Complex Networks using Genetic Algorithms," arXiv:0711.0491, 2007.

[18] D. He, Z. Wang, B. Yang, and C. Zhou, "Genetic Algorithm with Ensemble Learning for Detecting Community Structure in Complex Networks," In 4th International Conference on Computer Sciences and Convergence Information Technology (ICCIT 09), IEEE Press, Nov. 2009, pp. 702-707, doi:10.1109/ICCIT.2009.189.

[19] S. Li, Y. Chen, H. Du, and M. W. Feldman,"A Genetic Algorithm with Local Search Strategy for Improved Detection of Community Structure," Complexity, vol. 15, Mar. 2010, pp. 53-60, doi:10.1002/cplx.v15:4.

[20] C. Pizzuti, "Community Detection in Social Networks with Genetic Algorithms," In Genetic and Evolutionary Computation Conference (GECCO 08), ACM Press, Jul. 2008, pp. 1137-1138, doi:10.1145/1389095.1389316.

[21] C. Pizzuti, "A Multi-Objective Genetic Algorithm for Community Detection in Networks," In 21st IEEE Int'l Conference on Tools with Artificial Intelligence (ICTAI 09), IEEE Press, Nov. 2009, pp. 379-386, doi:10.1109/ICTAI.2009.58.

[22] S. Fortunato, and M. Barthélemy, "Resolution Limit in Community Detection," Proceedings of the National Academy of Sciences, vol. 104, Jan. 2007, pp. 36-41, doi:10.1073/pnas.0605965104.

[23] Y. J. Park, and M. S. Song, "A Genetic Algorithm for Clustering Problems," In Proceedings of the 3rd Annual Conference on Genetic Programming, Morgan Kaufmann, Jul. 1998, pp. 568-575.

[24] J. Handle and J. Knowles, "An Evolutionary Approach to Multiobjective Clustering," IEEE Transactions on Evolutionary Computation, vol. 11, Feb. 2007, pp. 56-76, doi:10.1109/TEVC.2006.877146.

[25] N. Makate, M. Miki, T. Hiroyasu, and T. Senda, "Multiobjective Clustering with Automatic K-Determination for Large-Scale Data," In Proc. of the Int. Genetic and Evolutionary Computation Conference (GECCO 07), ACM Press, Jul. 2007, pp. 861-868, doi:10.1145/1276958.1277126.

[26] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms. Cambridge, MA: MIT Press, 2001.

[27] P. Erdos, and A. Renyi, "On the Evolution of Random Graphs," Publ. Math. Inst. Hung. Acad. Sci., vol. 5, 1960, pp. 17-61.

[28] E. M. Montes, and C. A. C. Coello, "A Simple Multi-Membered Evolution Strategy to Solve Constrained Optimization Problems," IEEE Transactions on Evolutionary Computation, vol. 9, Feb. 2005, pp. 1-17, doi:10.1109/TEVC.2004.836819.

[29] G. Syswerda, "Uniform Crossover in Genetic Algorithms," In Proc. 3rd Int. Conf. Genetic Algorithms, Morgan Kaufmann, Jun. 1989, pp. 2-9.

[30] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Statistical Properties of Community Structure in Large Social and Information Networks," In Proceedings of the 17th International Conference on World Wide Web (WWW 08), ACM Press, Apr. 2008, pp. 695-704, doi:10.1145/1367497.1367591.

[31] W. W. Zachary, "An Information Flow Model for Conict and Fission in Small Groups," J. Anthropological Research, vol. 33, 1977, pp. 452-473.

[32] D. Lusseau, "The Emergent Properties of a Dolphin Social Network," Proc Biol Sci, vol. 270 Suppl 2, Jul. 2003, pp. S186-8, doi:10.1098/rsbl.2003.0057.

[33] M. E. J. Newman, "Modularity and Community Structure in Networks," Proceedings of the National Academy of Sciences, vol. 103, Jun. 2006, pp. 8577-8582, doi:10.1073/pnas.0601602103.

[34] P. M. Gleiser, and L. Danon, "Community Structure in Jazz," Advances in Complex Systems, vol. 6, Jul. 2003, pp. 565-573 doi:10.1142/S0219525903001067.

[35] M. E. J. Newman, "The Structure of Scientific Collaboration Networks," Proceedings of National Academy of Science, vol. 98, Jan. 2001, pp. 404-409, doi:10.1073/pnas.021544898.