2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Evolutionary Community Detection in Social Networks

Tiantian He and Keith C. C. Chan
Department of Computing
The Hong Kong Polytechnic University
Hong Kong, China
{csthe, cskcchan}@comp.polyu.edu.hk

***Abstract*—As people that share common characteristics and interests tend to communicate with each other more frequently, they form communities within social networks. Several methods have been developed to discover such communities based on topological metrics. These methods have been used to successfully discover communities that are relatively large, but for communities characterized by members interacting more frequently with each other rather than interacting with many others, we propose here an effective method which is based on the use of an evolutionary algorithm (EA) called ECDA. Given a social network represented as a graph, unlike existing approaches, ECDA considers both topological metrics of the graph and the attributes of the vertices and edges when detecting for communities in the network. It performs its task by formulating the community detection problem as an optimization problem. By computing a measure of statistical significance for each attribute of the vertices, ECDA looks for communities in a network that have maximal connection significance within a community and minimal significance between any two communities. With such a strategy, ECDA partitions a network into different communities consisting of members with similar attributes within and different attributes without. Unlike other EAs, ECDA adopts a reproduction process consisting of special crossover and mutation operators, called Self-Evolution, to speed up the evolutionary process. ECDA has been tested with several real datasets and its performance is found to be very promising.**

Keywords—evolutionary algorithm; genetic algorithm; community detection; social network

I. INTRODUCTION

Community detection refers to the identification of a goal-oriented partition of a large social network into smaller ones. To do so, such a network is usually first represented as a graph with vertices labeled by an ID and edges that do not take any value except for weights computed according to topological significance.

Existing community detection algorithms consider only topological information in a network and communities are therefore detected based only on the structure of communities. However, in a typical social network, it can be hobbies and political preferences that decide whether or not a user belongs to a particular community. Hence, other than topological properties, a network can be partitioned into very communities if such information is used as well.

One of the most popular properties of a graph that is considered by many algorithms to detect for communities in social networks is *modularity*, Q [12]. A community with high modularity means that its members are connected with each other much more so than that of the others in the other communities. The use of modularity in community detection was first proposed in [12] with a fast algorithm developed to make use of modularity to identify communities hierarchically. The complexity of this fast algorithm was further reduced later by using a sparse-matrix data structure as proposed in [2]. Though a reasonable approach, it is well known that the use of modularity as a criteria for defining a community has some weaknesses – it usually finds one or few very big communities and many other very small ones [17]. To improve such an approach, parameters such as cluster size, edge density penalty are added at each step in the community detection process to reduce community size while compromising modularity score. As a result, the performance of these modified modularity-based approaches is therefore not as good as the original fast algorithms.

Other than modularity, another graph property that is very often used when detecting communities in large networks is known as *edge centrality* [5]. Edge centrality can be considered as a measure computed by assigning higher weights to those edges which connect different communities. Through eliminating connections of the highest weights, community structures can then be unfolded. However, approaches that are based on edge centrality are usually slow. In fact, its computational complexity is approximately $O(mn^2)$, m and n being the numbers of edges and vertices [4].

Recently, there has been some effort to detect community structures using Genetic Algorithms (GAs) as such algorithms can potentially find more feasible partitions in large, complex networks and the resulting partitions can, theoretically, be optimized. In fact, several GA based approaches to community detection have been proven to be effective. The first successful GA application for community detection was proposed in [16]. The approach is to make use of modularity, Q, as the fitness function so that Q can be optimized.

Another GA developed to detect community structures was proposed in [9]. It makes use of a *Silhouette Width* [14] and a *normalized cut* [15] measure in its fitness function.

This GA also introduces a crossover operator to allow community structures to be exchanged. Even though it makes the speed of convergence of the evolutionary process slow, the crossover operator allows more varieties of community structures are considered.

In addition to the above two GAs, several other GAs have also been proposed for community detection and they used different topological metrics when performing their tasks. These GAs have been used successfully for different applications.

The majority of the community detection algorithms, whether or not they are GA based, make use of various topological properties of graphs. The performances of these algorithms are evaluated according to how good they are at finding partitions that are optimal or near-optimal against different graph topological measures.

Since a community can also be formed by means of individual attributes about its members rather than topological properties, there is a need for algorithms that can take attributes of individual members that may be shared by other members of a particular community, into considerations. For this purpose, we propose to use an Evolutionary Algorithm (EA) called ECDA (Evolutionary Community Detection Algorithm). ECDA has these characteristics: (i) it represents a community for the evolutionary process using a weighted graph constructed based on the significance of the connections in a network; (ii) it guides the evolutionary process using a fitness function defined in terms of the number of significant connections within and between communities; (iii) it uses a novel crossover operator to allow the structure of two communities to be swapped; (iv) it uses a novel mutation operator to allow Self-Evolution (SE) to take place so as to shorten the time required for convergence. ECDA has been tested with several large datasets and its performance has been found to be very promising.

In the following section, we first describe how meaningful the information for each vertex can be identified and how the significance of connections can be measured with the identified information. In Section III, two different fitness functions are proposed to guide the evolutionary process. In Section IV, the reproduction operators of ECDA are described in details. The performances of ECDA are evaluated with different real datasets and the results are presented in Section V. Finally, in the last section, we give a conclusion.

II. DISCOVERING MEANINGFUL INFORMATION IN NETWORKS

Most of existing approaches to community detection treat the problem as discovering community with special topological properties. However, in the real world, a community may be formed when users share some common interests or characteristics and this may not necessarily be reflected by whether or not they communicate with each other frequently enough.

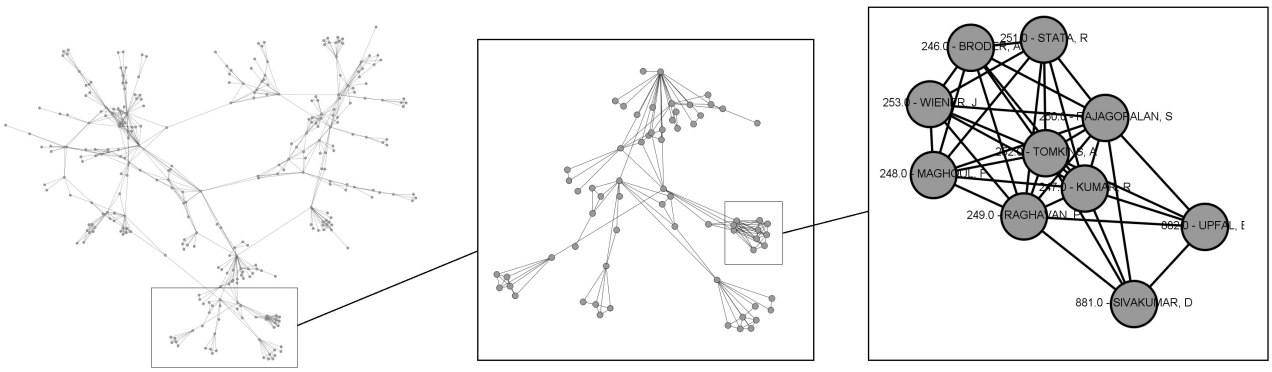
A. Social Network and its Literal Information

In this section, we introduce the notation used to represent a social network. Given a social network, we represent each member in it as a vertex in a graph and we represent whether or not two members communicate with each other as an edge in it. Given a vertex set V, and an edge set E, the graph can be represented as $G(V,E)$. The vertices set, $V=\{ v_i|i=1,2,..., |V|\}$ refers to all the vertices in the network, and $E=\{e(v_i, v_j)=1|i\neq j, v_i, v_j \in V\}$ represents the edge set in G. $|V|$ and $|E|$ are the total numbers of vertices and connections in the graph. An example of such a graph can be represented as Fig. 1 (a) and Fig. 1 (b).

Besides topological properties, ECDA assumes that there are attribute values associated with each member of the network. These attribute values represent "*topics*" or "*labels*". For a vertex v_i in G, these attribute values include a unique identifier, $ID=i$ and a collection of all the topics that are associated with i. This collection of topics can be represented as a set $T_i=(t_{i1}, t_{i2}, t_{i3}...)$, $T_i < T$, where T is the set of all topics that can possibly be assigned to the vertex (Fig. 1 (c)). It is because of this characteristic that the graph of the network is also called topic-labeled graph (*TG*) after each vertex is assigned its corresponding features.

B. The Significance of Connection

The vertices and edges in *TG* indicate not only relationship among members but also how they may relate



(a) (b) (c)

Fig. 1. A social network with topics (*TG*)

to each other based on how the labeled topics relate to each other. As it is usual for people with the same or similar background, ideology or interests to interact more with each other, we can expect a community to be formed among people sharing similar attribute values such as similar discussion topics of common interest or book titles, etc.

To identify such topics of interest to a special community, a statistical measure defined in [3] is adopted here for the determination of significance between each pair of topics. In this paper, this measure is referred to as the measure of Significance of Connection and it is defined as follows.

Given two vertices v_i and v_j in *TG*, their associated topics are therefore represented as T_i and T_j, and $T_i \cap T_j$ may or may not be empty. For any one topic from T_i, t_{im} and another one from T_j , t_{jn}, we define $o(t_{im}, t_{jn})$ as the frequency of occurrence of edges connecting a vertex labeled with t_{im} and the other done with t_{jn}. We also define $e(t_{im}, t_{jn})$ to represent the expected frequency of occurrence under the assumed condition that t_{im} and t_{jn} are independent of each other. From a statistical perspective, if t_{im} and t_{jn} are mutually dependent, $o(t_{im}, t_{jn})$ and $e(t_{im}, t_{jn})$ should be significantly different. So the statistical measure to evaluate the significance of differences between $o(t_{im}, t_{jn})$ and $e(t_{im}, t_{jn})$ can be defined by (1), where $o(t_{im}+)$ is the frequency of the occurrence of the edges connecting vertices labeled with t_{im}.

$$r(t_{im}, t_{jn}) = \frac{o(t_{im}, t_{jn}) - e(t_{im}, t_{jn})}{\sqrt{e(t_{im}, t_{jn})\left(1 - \frac{o(t_{im}+)}{|E|}\right)\left(1 - \frac{o(t_{jn}+)}{|E|}\right)}} \quad (1)$$

Based on it, we can determine the significance between vertex *i* and *j* by (2).

$$S_{ij} = \max_{\substack{0 \le m \le |T_i| \\ 0 \le n \le |T_j|}} \left| r(t_{im}, t_{jn}) \right| \quad (2)$$

In other words, the significance between any two vertices in *TG* is the maximum of $r(t_{im}, t_{jn})$. In [6], $r(t_{im}, t_{jn})$ has been shown to follow an approximate Gaussian distribution so that if $r(t_{im}, t_{jn})$ has the value of 1.96, the two topics can be considered as having an association relationship with each other at a 95% confidence level. If S_{ij} is smaller than 1.96, we can say that the connection between the two vertices is insignificant.

After the significant connections are identified and the significance measure is determined, the measure is normalized using an information measurement [10]. Let t_{im} and t_{jn} be the most significant topic pair between vertex *i* and vertex *j*, then an information redundancy between two vertices can be defined as:

$$R(v_i, v_j) = p(t_{im}, t_{jn}) \log \frac{p(t_{im}, t_{jn})}{p(t_{im})\, p(t_{jn})} \quad (3)$$

where $p(t_{im}, t_{jn})$ is the joint probability of t_{im} and t_{jn} which can be obtained by dividing $o(t_{im}, t_{jn})$ by $|E|$, $p(t_{im})$ is the probability that edges connect vertices with t_{im} and it can be obtained by dividing $o(t_{im}+)$ by $|E|$. Hence, $R(v_i, v_j)$ is a mutual information measure representing the average reduction of uncertainty about the connection of the topic pair t_{im} and t_{jn}. A large magnitude means that the relationship between topic pairs is meaningful. However, since the radix of each pair of topics is different, it is better for the joint entropy, $H(v_i, v_j)$ to normalize $R(v_i, v_j)$ and $H(v_i, v_j)$ is defined as:

$$H(v_i, v_j) = -p(t_{im}, t_{jn}) \log p(t_{im}, t_{jn}) \quad (4)$$

Given $R(v_i, v_j)$ and $H(v_i, v_j)$, we can define the Normalized Significance of the connection (NS_{ij}) connecting vertex *i* and vertex *j* as:

$$NS_{ij} = C * \frac{R(v_i, v_j)}{H(v_i, v_j)} \quad (5)$$

where *C* is a constant that is used to magnify or contract the range of NS_{ij}. Apparently NS_{ij} ranges from zero to *C*. If NS_{ij} is equal to *C*, it means the connection between vertex *i* and vertex *j* is completely significant. If it equals to zero, that means the connection is totally insignificant. The value of NS_{ij} is always between 0 and *C*, which means that the connection between two vertices is partially significant. To be added by the binary value of the solid connection, *TG* is represented as a weighted graph (*WTG*) and its connections stand for not only topological information, but also practical significance resulted from the labeled topics. And *WTG* is the graph which will be detected.

III. BENCHMARKING FOR COMMUNITY DETECTION

In this section, we discuss the benchmarks used in evaluating effectiveness of community detection algorithms. Previous community detection algorithms mainly make use of topological properties. For more effective performance evaluation, we define a community as follows.

For any vertex in a community, it tends to connect with those vertices which share significant topic pairs. Based on this definition, the following two metrics are used for identifying communities. Let WTG_p $(0 \leqslant p < P)$ be a sub-graph of *WTG*, where *P* is the total number of the sub-graphs and all the figures of the significance of connections have been obtained. For all vertices in WTG_p, the number of significant connections within WTG_p can be summarized as NS_{WTGp}. Therefore, the proportion of significant connection, P_{WTGp} is the quotient of all the significant connections divided by the expected number of edges in WTG_p and it can be described as equations (6) and (7):

$$NS_{WTG_p} = \frac{1}{2} \sum_{0}^{I} NS_{v_i} \quad (6)$$

$$P_{WTG_p} = \frac{NS_{WTG_p}}{|E_p|} \quad (7)$$

where $|E_p|$ refers to the value under the assumption that every vertex has connections to all the other ones in WTG_p and NS_{WTGp} is also computed under this prerequisite. Making this assumption here aims at putting the definition above into practice: although currently, there is no connection between a pair of nodes, they have the potential to interact with each other if they have a significant topic pair, which is the reason why they are located into the same

community. After the proportion of significant connection is acquired, the average proportion of significant connection can be defined as the following:

$$AP_{WTG} = \sum_{P} \frac{|V_p|}{|V|} P_{WTG_p} \tag{8}$$

where $|V_p|$ is the number of nodes in WTG_p and $|V|$ is the total number of vertices in WTG. AP_{WTG} reflects the average number of the significant connection between a vertex and another after dividing WTG into P parts and it ranges from 0 to 1. That AP_{WTG} is maximized means the partition possesses the highest proportion of the significant connection in all communities and any two vertices in the same community have the largest likelihood to interact with each other, although there may not be a solid relationship between them currently. It should be explained that the hypothesis about $|E_p|$ is effective only for the computation of NS_{WTGp} and for other computations of connections and communities, they consider solid significant links only.

Besides of considering interactions within a community, we should also pay attention to the ones connecting different communities. Based on the proposed definition, significant connections between communities should be minimized since too many of them cause a vague boundary between communities. Here we define a measure to weight the vagueness of the community boundary. Given a community WTG_p which contains $|V_p|$ vertices, the vagueness of the community boundary is defined as:

$$VB_{WTG_p} = \frac{\sum_{|V_p|} NS_{ij}\left(v_j \in V_p\right)}{\sum_{|V_p|} NS_{ij}\left(v_j \in V_p\right) + \sum_{|V_p|} NS_{ij}\left(v_j \notin V_p\right)} \tag{9}$$

where NS_{ij} is the aggregated value of all significant connections that start with vertex i, $v_j \in V_p$ represents two vertices are both in WTG_p, and the other one means a vertex is out of WTG_p. Like what has been done in equation (8), the mean vagueness of the community boundary is defined as equation (10):

$$VB_{WTG} = \sum_{P} \frac{|V_p|}{|V|} VB_{WTG_p} \tag{10}$$

VB_{WTG} evaluates to what extent the differentiation is if a community is compared to another and it also ranges from 0 to 1. Apparently, the higher VB_{WTG} is, the more differentiated from other ones a community is. Therefore, VB_{WTG} helps to diminish the similarity and significant interactions between any two communities.

Based on the maximization or the arbitral settlement of AP_{WTG} and VB_{WTG}, in theory a network can be divided into a number of communities which are differentiated from each other, and their vertices have considerable probability to interact with each other as well as possess a number of practical significant connections.

IV. OPTIMIZATION THROUGH EVOLUTIONARY APPROACH

As a part of evolutionary computing which was first proposed by I. Rechenberg in the 1960s, the idea of evolutionary algorithm (such as GA) is invented in [7]. With the growing complexity of recent problems, the evolutionary algorithm (EA) earns considerable attention since its high efficiency on some problems with which traditional approaches are hard to deal, such as NP-Complete problems. In the first ten years of this century, the applications of EA to community detection have been looked into by many researchers who are interested in the field of analyzing complex networks, such as social network analysis and there have been some remarkable publications which are plunged much effort by the academia. Thus EA still plays an important role in modern research and brings us consistent surprise. In this section, the detail of our algorithm (ECDA) will be illustrated.

A. Gene Representation

When the evolutionary algorithm is applied to the task of community detection, two dominant methods for representing the information on the community structure are preferred. One is *locus-based adjacency representation* [13], the other is *straight-forward representation* [8]. In ECDA, the straight-forward representation is used for encoding. The straight-forward representation encodes gene and constructs the chromosome in a more directive way: Given a chromosome of a population which contains M genes, where M stands for the number of the communities, each gene is assigned a value from 1 to M. As a result of that, the ith gene's value j means the ith vertex in the network belongs to the jth community. Fig. 2 illustrates a brief instance of straight-forward representation.

B. The Revised Initialization

Since the straight-forward representation is used in ECDA, the parameter M must be determined before the initialization. Here M is a randomized parameter decided by ECDA before each chromosome is initialized. And in our approach, the maximum value of M is set to $|V|/2$. It is usual that the ordinary way of initializing a chromosome is to assign a randomized community ID to each gene. However, the randomness of this method might not ensure the quality of the initialized chromosome and might lead to a time-consuming phase before obtaining the best solution. Therefore, the process of initialization in ECDA is revised to generate more qualified chromosomes: First, M nodes is selected randomly as the initial communities; Second, vertices possessing at least one significant connection are

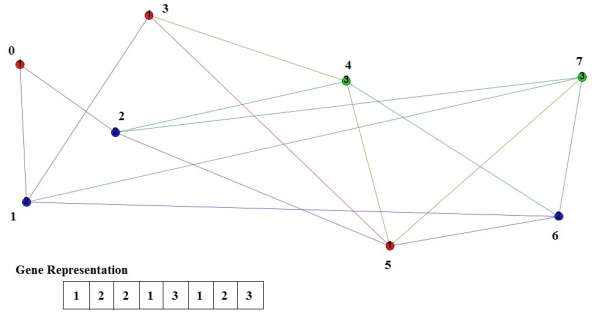


Fig. 2. Straight-forward representation

randomly assigned to a community with which they share the significant connection; Third, all the vertices without any significant connection are assigned to a community randomly. The above three-step method are iterated P times, where P is the size of the population. After that, ideally, there will be P different solutions dividing the network into different numbers of communities.

C. The Process of Reproduction

This stage contains two sub-steps, which are crossover and mutation. Rather than the standard crossover and mutation, the operator of the uniform crossover is modified and then we use an extraordinary phase which is named as *Self-Evolution* (*SE*) to improve the fitness of the new-birthed chromosome and obtain satisfactory chromosomes in less time.

1) Crossover

In this phase, ECDA reproduces one descendant after each selection. Aiming at letting more adaptive chromosomes have more opportunities for crossover, ECDA allows only a particular proportion of more adaptive chromosomes to involve into reproduction and the proportion is an arbitral value of 0.30. Having selected the parents randomly in the population of candidates, ECDA will crossover them in the following way: First, one parent is selected as a template; Second, for each community in the other parent, their members replace the allele genes in the template if the rate of crossover is satisfying and each member of a selected community is possible to mutate according to the mutation rate. Having completed the above steps, ECDA will reproduce a descendant.

2) Self-Evolution

Although the fitness of the population is improved as generations grow, it would be an extremely time-consuming stage to obtain the satisfactory fitness if no factitious assistance is involved, especially when the data size is very large. Therefore, a new phase helping the descendant to be more adaptive will be executed after each crossover. And it is named as *Self-Evolution* (*SE*). The core of SE is to search a more appropriate community for each vertex and move the vertex into it, based on the current vertices distribution. To achieve the expected goal, we use a two-dimension matrix to complete the stage of SE: Given all nodes in a network and all communities, the matrix $VC_{[|V|][M]}$ is defined to represent how significant the relationship between a vertex and a community is. For an element $VC_{[i][j]}$, it equals to zero if there is no connection between node i and community j. Otherwise, it means several solid connections or significant connections bridge node i and community j. By using $VC_{[|V|][M]}$, Self-Evolution is completed like the follows: For each vertex in some community j, if $VC_{[i][max]} \neq VC_{[i][j]}$, it is moved from community j to community max; If node k connects node i, $VC_{[k][j]}$ is decreased by NS_{ik}, while $VC_{[k][max]}$ is added by NS_{ik}. After the completion of SE, those communities which possess fewer significant connections might be eliminated and the community structure produced by crossover can be improved. ECDA has to spend much time on SE at the beginning, but after a number of generations, the time consumed by SE will be shortened as the proportion of more adaptive communities from the parents is higher. As a result of that, the efficiency of ECDA is enhanced.

D. The Fitness Function

As what has been illustrated in last section, we use equations (8) and (10) as fitness functions together and they take different but revisable weights, respectively. Equation (8) optimizes the proportion of significant connection both tangible and intangible within the community. And equation (10) diminishes the solid significant connections bridging two communities. The fitness of each chromosome is computed after initialization and reproduction.

E. Parameters of ECDA

In ECDA, the following parameters need to be determined before it is executed, including population size P, crossover rate cr, mutation rate mr and maximal number of generations mg. In order to reduce the time of initialization, we prefer to use a smaller population size. Although there are fewer potential solutions, it has limited negative-effect to the experimental results under the assistance of SE. For the configurations of crossover rate and mutation rate, ECDA uses common settings. The detail information on these parameters is shown in Table 1.

F. Summary of the algorithm

Based on the description above, the main body of ECDA is summarized as the pseudo codes in Fig. 3.

V. EXPERIMENTAL RESULTS

The performance of ECDA for community detection has been evaluated by using three sets of real data: the Political Blogs dataset [1], the High energy theory collaboration network dataset [11] and the Astrophysics collaborations dataset [11]. These datasets are abbreviated as PB, HC and AC, respectively. These datasets have the following characteristics: (i) PB is a network dataset collected around the 2004 US presidential election. The network contains 1,222 vertices and 16,174 edges. (ii) HC is a network of collaborating scientists posting preprints on the High-Energy Theory Archive at www.arxiv.org. The dataset contains 13,815 edges and 5,835 vertices representing authors and their research fields. (iii) AC is a network dataset of scientists posting preprints on the Astrophysics Archive at www.arxiv.org from 1995 to 1999. It contains 119,652 edges and 14,845 vertices representing scientists with their research interests. A summary of the above three datasets and some examples of the data used in our approach are shown in Tables 2 and 3. The performance of ECDA has been compared to several different algorithms which include improved fast algorithm CNM [2], and fast algorithms with parameters HE, HN and HEN [17].

A. Computational efficiency

To determine the efficiency of different algorithms, we

TABLE 1. PARAMETERS IN ECDA

P	cr	mr	mg
50, 100, 200	0.5, 0.6	0.01	100, 200

```
initialization;
done = false; count = 0;
if(fitness){ done = true;
  found satisfying solution; break;}
else{ while(done == false){ count += 1;
      for( i = 0; i < maxdescendant; i++){
        crossover;
        self-evolution;
        insert descendant into population;}
       elimination;
      if(fitness){
        done = true;
        found satisfying solution; break;}
      if(count == mg){
        cannot find satisfying solution; break;}
      if(count%10==0){
        initialization;
        insert best chromosomes;
        elimination;
        if(fitness){
          done = true;
          found satisfying solution; break;}}}}
```

Fig. 3. Pseudo codes of ECDA

recorded the average execution time when the algorithms were used with the three datasets. The results are shown in Table 4. For the dataset PB, ECDA completes its tasks at an average time of 1.56 seconds (ranked 4th place). When data size becomes larger, the results are quite different. For HC, ECDA uses approximately 6.7 seconds to identify all communities in the network, but other algorithms uses about 10 seconds more to complete the task. For the largest dataset, the average execution time of ECDA is 59.3 seconds, but that of other algorithms are between 147 seconds and 211 seconds.

For the fast algorithms, it should be noted that their complexity is of the order of $O(n\log^2 n)$, where n is the total number of vertices in the network. As a result, the running time increases quickly as the size of the dataset becomes large. Compared to these fast algorithms, the complexity of ECDA can be considered separately for initialization and reproduction. When ECDA initializes population whose size is P for a network containing n vertices and m edges (significant and insignificant ones), for each chromosome which can be randomly initialized into c communities, it performs its tasks requiring $O(2(m+c)+cn'^2)$, where n'^2 is the average size of each community, to construct the chromosomes and compute their fitness values. Hence, the total time for initialization is of the order of $O(P(2(m+c)+cn'^2))$. For reproduction, if d descendants are produced for each generation, and if the rate of crossover is r, ECDA works under the complexity of $O(nr)$ and it takes $O(c(n'^2+1)+n(c+e)+2m)$, where e is the average degree of each vertex, for mutation and the computation of fitness. For the whole reproduction process, therefore, the complexity is $O(d(c(n'^2+1)+n(c+e+r)+2m))$.

In summary, if it takes G generations to converge to optimal community structures, the complexity of ECDA is of $O(P(2(m+c)+cn'^2)+dG(c(n'^2+1)+n(c+e+r)+2m))$. Since P and dG are much smaller than m and n, we can assume that the two are equal to a constant con and $m\sim n$ and the complexity becomes $O(con(n(c+e+r+4)+c(3+2n'^2)))$. But as the reproduction goes on, the complexity should be much lower than this estimate since the operations for SE (i.e., $O(n(c+e))$) decrease tremendously. Based on the complexity of different algorithms and the average execution time shown in the table, we can conclude that the complexity of our approach is less than $O(n\log^2 n)$. But like other EAs, ECDA needs time to initialize and to compute fitness values. Due to connatural flaws of the evolutionary algorithm, it is hard to tremendously reduce the complexity of ECDA. However, ECDA is still faster than those approaches (e.g. algorithms that are based on the use of edge centrality) whose complexity is equal to or larger than $O(n\log^2 n)$.

B. The Effectiveness of the approach

In order for us to compare the effectiveness of ECDA with the other algorithms, we need to evaluate the quality of the community detected. To do so, we used the *Modularity* (Q) measure as it is one of the most prevalent benchmarks for evaluating the effectiveness of community detection algorithms. As described in the first section, Q is defined as:

$$Q = \sum_i \left(e_{ii} - a_i^{2`} \right) \tag{11}$$

where e_{ii} represents the fraction of solid links within community i and a^2 stands for the links in this community under the expected situation. The experimental results of Q obtained by different algorithms are shown in Table 5.

From the table, ECDA performs well in datasets PB and AC. For these two datasets, it performs better than other fast algorithms based on modularity optimization. For the dataset HC, the modularity obtained by ECDA is 0.03 less than the best. As ECDA does not make use of modularity as a criterion for optimization, it is understandable that it may not always perform better than approaches that adopt modularity as a criterion for performance optimization. Despite this being the case, it shows that weighing the significance of the connection for community detection can allow very satisfactory results to be obtained even when a performance measure based on topological properties is adopted.

TABLE 2. DETAIL INFORMATION ON THE DATASETS

Dataset	PB	HC	AC
Vertices	1222	5835	14845
Edges	16174	13815	119652
Topics	7	5293	11340

TABLE 3. VERTICES IN PB

ID	Label (Source)
1306	BlogPulse, LabeledManually, eTalkingHead
1307	Blogarama
1308	Blogarama, BlogCatalog

TABLE 4. EXECUTION TIME OF DIFFERENT APPROACHES

	ECDA	CNM	HE	HN	HEN
PB	1.56s	1.3s	1.3s	2s	1.2s
HC	6.67s	16s	17s	17s	18s
AC	59.3	147s	178s	211s	179s

(a) Astrophysics collaborations (b) High energy theory collaboration (c) Political Blogs

Fig. 4. Results obtained by ECDA in dataset AC, HC and PB, different communities are with different colors

Besides comparing modularity of different algorithms, we also obtained the number of detected communities by different methods and the data of AP_{WTG} and VB_{WTG} from other algorithms. The results of the comparisons are summarized in Tables 6 and 7. From Table 6, it is seen that ECDA tends to identify more communities in a given network especially in larger ones such as HC and AC. As shown in Table 7, most results of AP_{WTG} and VB_{WTG} obtained by ECDA are better than those obtained by the other four approaches, except for the figures of VB_{WTG} for dataset HC and AC. According to Tables 5, 6 and 7, it is found that the proposed evolutionary algorithm can detect communities with higher proportion of significant connections. Moreover, since ECDA finds communities with optimally significant maximal intra-community connections and significant minimal inter-community connections, it ensures that it performs well with AP_{WTG}, VB_{WTG} and modularity.

Based on the experimental results, it can be seen that modularity optimization can improve the performance of VB_{WTG} to some extent like that of the performance obtained by other algorithms. However, maximizing modularity may sometimes result in the pitfall that the detected communities might be meaningless. For example, for the case of HC, the other four algorithms obtained very high VB_{WTG} which are approximately 0.1 higher than the VB_{WTG} of ECDA. However, the difference in AP_{WTG} is evident as it ranges from 0.35 to 0.38. As a result, the vertices in the detected communities obtained by those modularity optimization algorithms have meaningful relationship only with those sharing solid connections. For those nodes which are disconnected, they will probably never interact with each other because they share insufficient similarity causing interactions potentially. Besides considering topological information in a network, our EA also emphasizes on the practical meaning of communities just like the benchmarks proposed in this paper and it still can obtain decent experimental results evaluated by the metrics which are based on topological aspects, such as modularity. So, ECDA is a feasible approach to community detection and can identify meaningful communities in social networks.

VI. CONCLUSION

In this paper, a novel evolutionary algorithm for community detection, ECDA is proposed. Compared to many existing approaches, which mainly consider topological properties when detecting communities in networks, ECDA considers both topological properties and attribute information as criteria for defining a community during the evolutionary process. The results of performance evaluation show that ECDA can be efficient and effective for the detection of meaningful communities. The fact that ECDA can perform relatively well means that, considering attribute information for each vertex in defining a community can allow very satisfactory results to be obtained even if the evaluation metrics are based on topological properties. ECDA has demonstrated that it can detect meaningful communities when there is more information on practical meaning. For future work, the efficiency of ECDA and the balance between AP_{WTG} and VB_{WTG} will be further improved.

TABLE 5. MODULARITY OBTAINED BY DIFFERENT ALGORITHMS

	PB	HC	AC
ECDA	0.4247	0.7331	0.6574
CNM	0.4269	0.7687	0.6262
HE	0.4002	0.7629	0.5679
HN	0.3952	0.7758	0.5366
HEN	0.3875	0.7386	0.5308

TABLE 6. NUMBER OF COMMUNITIES OBTAINED BY DIFFERENT APPROACHES

	ECDA	CNM	HE	HN	HEN
PB	3	11	12	11	5
HC	235	94	43	40	44
AC	725	161	56	43	49

TABLE 7. AP_{WTG} AND VB_{WTG} OF DIFFERENT ALGORITHMS

		ECDA	CNM	HE	HN	HEN
PB	AP_{WTG}	0.43	0.429	0.427	0.426	0.426
	VB_{WTG}	0.942	0.911	0.890	0.910	0.899
HC	AP_{WTG}	0.438	0.082	0.058	0.061	0.052
	VB_{WTG}	0.752	0.891	0.843	0.852	0.835
AC	AP_{WTG}	0.40	0.229	0.193	0.197	0.188
	VB_{WTG}	0.812	0.84	0.706	0.657	0.676

REFERENCES

[1] L.A. Adamic, "The political blogosphere and the 2004 U.S. election," in *Proc. 3rd Int. workshop. Link Discovery*, New York, 2005, pp. 36-43.

[2] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E*, vol. 70, pp. 066111-066116, December 2004.

[3] K. C. C. Chan, and A. K. C. Wong, "A Statistical Technique for Extracting Classificatory Knowledge from Databases," in *Knowledge Discovery in Databases*, G. Piatetsky-Shapiro and W.J. Frawley, eds., Cambridge, Mass.: AAAI/MIT Press, 1991, pp. 107-123.

[4] S. Fortunato, "Community Detection in Graphs," *Physics Reports*, vol. 486, no. 3-5, pp. 75-174, February 2010.

[5] M. Girvan, and M. E. J. Newman, "Community Structure in Social and Biological Networks," *PNAS*, vol. 99, pp. 7821-7826, 2002.

[6] S. J. Haber, "The analysis of Residuals in Cross-Classified Tables," *Biometrics*, vol. 29, pp. 205-220, March 1973.

[7] J. H. Holland, *Adaptation in Natural and Artificial Systems*. Oxford, U. K.: Univ. Michigan Press, 1975.

[8] Y. Hong, S. Kwong, H. Xiong, and Q. Ren, "Genetic-Guided Semi-Supervised Clustering Algorithm with Instance-Level Constraints," in *Proc. 10th Annu. Conf. Genetic and Evolutionary Computation*, Atlanta, 2008, pp. 1381–1388.

[9] M. Lipczak, and E. Milios, "Agglomerative Genetic Algorithm for Clustering in Social Networks," in *Proc. 11th Annu. Conf. Genetic and Evolutionary Computation*, New York, 2009, pp. 1243-1250.

[10] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge, U. K.: Cambridge Univ. Press, 2003.

[11] M. E. J. Newman, "The structure of scientific collaboration networks," *PNAS*, vol. 98, no. 2, pp. 404-409, 2001.

[12] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Phys. Rev. E*, vol. 69, pp. 066133-066137, June 2004.

[13] Y. J. Park, and M. S. Song, "A genetic algorithm for clustering problems," in *Proc. 3rd Annu. Conf. Genetic algorithms*, 1989, pp. 2-9.

[14] P. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53-65, November 1987.

[15] J. Shi, and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888-905, August 2000.

[16] M. Tasgin, and H. Bingol, "Community detection in complex networks using genetic algorithm," in *Proc. European Conf. Complex Systems*, Oxford, 2006, pp. 57.

[17] K. Wakita, and T. Tsurumi, "Finding community structure in mega-scale social networks," in *Proc. 16th Int. Conf. World Wide Web*, New York, 2007, pp. 1275-1276.