Overlapping Community Detection for Multimedia Social Networks

Faliang Huang, Xuelong Li, *Fellow, IEEE*, Shichao Zhang, *Senior Member, IEEE*, Jilian Zhang, Jinhui Chen, and Zhinian Zhai

***Abstract*—Finding overlapping communities from multimedia social networks is an interesting and important problem in data mining and recommender systems. However, extant overlapping community discovery with swarm intelligence often generates overlapping community structures with superfluous small communities. To deal with the problem, in this paper, an efficient algorithm (LEPSO) is proposed for overlapping communities discovery, which is based on line graph theory, ensemble learning, and particle swarm optimization (PSO). Specifically, a discrete PSO, consisting of an encoding scheme with ordered neighbors and a particle updating strategy with ensemble clustering, is devised for improving the optimization ability to search communities hidden in social networks. Then, a postprocessing strategy is presented for merging the finer-grained and suboptimal overlapping communities. Experiments on some real-world and synthetic datasets show that our approach is superior in terms of robustness, effectiveness, and automatically determination of the number of clusters, which can discover overlapping communities that have better quality than those computed by state-of-the-art algorithms for overlapping communities detection.**

***Index Terms*—Ensemble learning, line graph, overlapping-communities detection, particle swarm optimization (PSO), social network.**

Manuscript received October 10, 2016; revised January 7, 2017 and March 3, 2017; accepted March 27, 2017. Date of publication April 12, 2017; date of current version July 15, 2017. This work was supported in part by the China 1000-Plan National Distinguished Professorship, in part by the China 973 Program under Grant 2013CB329404, in part by the China Key Research Program under Grant 2016YFB1000905, in part by the Natural Science Foundation of China under Grant 61672177 and Grant 61363009, in part by the Natural Science Foundation of Fujian Province under Grant 2017J01497, in part by the Guangxi Bagui Teams for Innovation and Research, and in part by the Guangxi Collaborative Innovation Center of MultisSource Information Integration and Intelligent Processing. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Shu-Ching Chen. *(Corresponding authors: Faliang Huang; Jilian Zhang.)*

F. Huang is with the Fujian Engineering Research Center of Public Service Big Data Mining and Application, Faculty of Software, Fujian Normal University, Fuzhou 350007, China (e-mail: faliang.huang@gmail.com).

X. Li is with the Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, China (e-mail: xuelong_li@opt.ac.cn).

S. Zhang is with the College of Computer Science and IT, Guangxi Normal University, Guilin 541000, China (e-mail: zhangsc@mailbox.gxnu.edu.cn).

J. Zhang is with the Department of Computer Science, Jinan University, Guangzhou, Shi 510630, China (e-mail: zhangjilian@yeah.net).

J. Chen is with the Graduate School of System Informatics, Kobe University, Kobe 657-0013, Japan (e-mail: ianchen@me.cs.scitec.kobe-u.ac.jp).

Z. Zhai is with the School of Information and Electronic Engineering, Zhejiang University of Science and Technology, Hangzhou 311122, China (e-mail: zhaizhinian@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2017.2692650

I. Introduction

SOCIAL networks have experienced explosive growth in the last decade. Social media websites, such as Twitter, YouTube and Flickr, have billions of users sharing opinions, photos and videos every day. Usually, users are provided with various features like *reply, comment, subscribe* and *connect to* in order to interact, engage and share information with each other. Such interactions lead to formation of closely knit user groups or densely connected clusters of users around specific topics within the social network; these groups are called communities. Communities discovery is of great importance for understanding the organization and function of social networks, and the extracted communities can be used in various applications such as topic discovery, targeted advertisement, recommendation of multimedia resources such as photos and videos [1]–[3].

Currently, techniques and theories developed for community mining have been successfully applied to multimedia-related applications, such as user modelling, photo tagging, video annotation, recommendation, targeted advertising etc. For example, [22] indicated that involvement of community information shows its potential in more effective targeted advertising, while [23] has successfully utilized community information to achieve more accurate multimedia annotation results. Besides, utilization of community and user connection information can significantly improve results of online recommendation of friends and multimedia resources [2], [3]. In addition, some novel applications in multimedia field, such as online extremism video detection [24] and human collective behavior understanding [25] can also be achieved through effective community discovery.

A plethora of approaches, such as divisive algorithms, dynamic algorithms, spectral algorithms, modularity maximization and statistical mechanics, have been developed for both efficient and effective community detection [4], [5]. However, most existing work focuses on disjoint communities discovery from social networks, i.e., each network node, representing a multimedia resource or a user, belongs to one community only. In reality, social network users are naturally characterized by multiple community memberships, as shown in Fig. 1. For instance, on the popular photo-sharing website Flickr, a user may be active in subscribing to users from a tourism group in order to view landmark photos, and she may also become a fan of other users from a sport group who publish photos related to football and hockey. Similar observations can be obtained on the video-sharing website YouTube. Therefore, for a social network depicted in Fig. 1(a), disjoint communities [Fig. 1(b)]

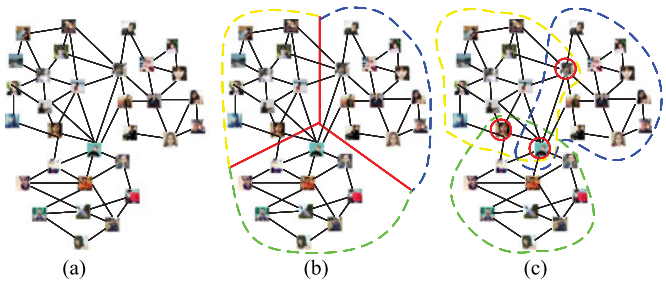

Fig. 1. Comparison of disjoint communities and overlapping communities: (a) the original social network; (b) the detected disjoint communities, where an anomalistic area with yellow/blue/green dashed boundary represents a tourism/movie/sports community; and (c) the detected overlapping communities, where three nodes in red circles are users belonged to more than one communities.

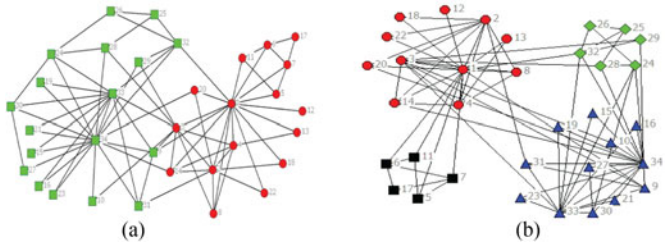

Fig. 2. Drawbacks of traditional approaches based on swarm intelligence optimization. (a) Real community structure, (b) generated community structure ($Q = 0.42$).

resulting from hard-partitioning techniques are less reasonable, compared to overlapping communities shown in Fig. 1(c).

A social network can be modeled as a graph by mapping entities to nodes, and interactions between the entities to edges. Generally, a community can be defined as a subgraph with nodes densely interconnected but sparsely connected to the rest of the graph. Actually, overlapping communities detection problem can be modeled as computing the optimal cover of graph nodes through optimizing some given objective function, such as modularity Q [6], ductance, etc. The NP-hard nature of this optimization problem leads to a class of community detection algorithms based on swarm intelligence techniques [7]–[17]. These swarm intelligence algorithms are indeed useful for overlapping communities detection, among which Particle Swarm Optimization (PSO) is the most representative one. However, PSO may not fully capture community structure information of a network, as shown in the example below.

Example 1: Fig. 2(a) is Zachary's network of karate club members, a well-known graph dataset used as a benchmark to test performance of community detection algorithms. It consists of 2 communities, whose members are depicted as red circles and green squares, respectively. Fig. 2(b) is the resulting communities detected by PSO, through optimizing modularity Q.

Comparing Fig. 2(a) with 2(b), there is a big difference between the real communities of the social network Zachary and the resulting communities generated by conventional PSO-based detection algorithms, i.e., the generated communities consist of superfluous small communities. For example, the community depicted in black squares and the one in green diamonds contain only 4 and 6 members, respectively. Therefore, traditional algorithms based on PSO cannot capture true social relation between members.

From the example we can see that existence of superfluous small communities may lead to unsatisfactory community structure. To alleviate this drawback, we can improve optimization strategy and employ post-processing strategy for merging superfluous small communities.

Compared to conventional optimization algorithms, PSO, which is a collaboration, communication and population based global search algorithm that uses principles of social behavior of swarms, has some advantages such as fast convergence rate, robustness to initial parameter values, tendency to be more accurate and to avoid getting trapped in local optima. Thus, PSO can produce better results in complicated and multi-peak problems, and it has attracted tremendous attention in data mining community recently [18].

Although there are many new variants of PSO, the majority, e.g., PSO-RF [19] and TPSO [20], are applicable to continuous space only, where trajectories are defined as trace of coordinate changes on the dimensions. Essentially, communities detection is a combinatorial optimization problem, where the solution space is discrete. Hence, existing PSO variants are not well suited to communities detection problem. A widely-used PSO variant, discrete PSO (DPSO for short) [21], is specifically designed for handling discrete optimization problems. Comparing to continuous PSO methods, DPSO is simple to implement and converges faster. In particular, for communities detection problems, DPSO needs know neither the number nor the size of communities in advance.

In this paper, we propose LEPSO, a meta-heuristic approach that combines together line graph theory, ensemble learning and particle swarm optimization techniques for overlapping communities detection. Specifically, we transform the overlapping communities detection problem into a disjoint communities detection problem on the corresponding line graph, and represent a community in a social network by a particle that is encoded based on ordered-neighbor-list. Then we use ensemble clustering techniques to improve the optimization strategy, so as to effectively optimize modularity of the line graph. After that, we convert the disjoint communities generated by DPSO into overlapping communities. Finally, we obtain the result by merging overlapping communities according to community overlapping rate. Experiments on real-world and synthetic networks indicate that the proposed method is able to find meaningful community structures from networks with satisfactory convergence rate.

Our contributions in the paper mainly include

1) We have proven that discovering overlapping communities from a social network is equivalent to detecting disjoint communities in the corresponding line graph of the social network.
2) To the best of our knowledge, for the first time we propose to incorporate ensemble clustering into discrete particle swarm optimization, for the purpose of boosting search ability to find high-quality and finer-grained overlapping communities from social networks.
3) A novel post-processing strategy is designed to merge finer-grained and suboptimal overlapping communities into better ones.

The rest of the paper is organized as follows. We review related work in Section II, and provide prerequisite definitions, theorems and corollaries in Section III. Section IV describes the proposed algorithm in details. We present experiment results on real-world and synthetic datasets, and conduct detailed discussion in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

Our approach is closely related to multimedia social networks, overlapping communities detection and discrete particle swarm optimization, and we review some of the most relevant work here.

A. Community Detection in Multimedia Social Networks

Widespread use of social multimedia applications, such as Delicious, Digg, Flickr and YouTube, has created multifarious multimedia social networks, arousing keen interests in performing community detection tasks on multimedia social networks, not only as a means of understanding the underlying phenomena taking place in such systems, but also to exploit the results in a wide range of intelligent services and applications. Santos *et al.* presented a characterization of the YouTube video-sharing virtual community and found that YouTube network has a distribution of content drastically influenced by social relationships [26]. Yeung *et al.* proposed to address the problem of tag disambiguation by applying community detection method to extract tag communities hidden in social media networks [27]. Tsatsou *et al.* integrated the results of tag community detection in a personalized ad recommendation system and compared against conventional nearest-neighbor tag expansion schemes [28]. To aid multimedia content discovery, Gargi *et al.* devised a multistage algorithm based on local-clustering and apply it to the YouTube video graph to generate named video communities [1]. However, none of the above work deals with overlapping communities detection problem. Zhao *et al.* [29] proposed to detect overlapped communities in multimedia social networks via hypergraph modelling which is different from our LEPSO technique; their goal is to detect meaningful social communities and uncover their underlying profiles in location-based social networks.

B. Overlapping Communities Detection

Recently, much of the effort in defining efficient and effective methods for community detection focused on finding overlapping communities [30]–[33], among which the link clustering methods have been successfully applied to overlapping communities discovery.

Link clustering methods propose to detect overlapping communities by partitioning links instead of nodes. The main advantage of clustering line graph is that it produces an overlapping subgraph of the original graph, thus allowing nodes to be present in multiple communities. Pereira *et al.* are the first to use line graph to find overlapping modules for protein-protein interaction networks [34]. Since then, series of line graph-based algorithms have been put forward, for example, Ahn *et al.* proposed a hierarchical agglomerative link clustering method to group links into topologically related clusters [35]. The algorithm applies a hierarchical method to line graph by defining two concepts: link similarity and partition density. Edge similarity threshold in the deterministic method [35] plays a key role and must be predetermined by users. In general, an improper threshold can easily mislead the clustering process and result in poor overlapping community structures. However, in real world applications it is often impossible for users to set an appropriate threshold value in advance, because most users have zero knowledge about their networks that are to be analyzed.

Evans and Lambiotte propose three quality functions for partitioning links of a network and explore the structure of the original graph with different dynamic processes such as random walk [36]. GA-NET+ attempts to detect overlapping communities using genetic algorithm(GA) to optimize chromosome fitness value (community score) [37]. Similar to GA-NET+, GaoCD first finds link communities by optimizing partition density and then maps link communities to node communities based on a novel genotype representation method [38]. Both GA-NET+ and GaoCD use genetic algorithm to search optimal partition of line graph and may be trapped in local optima easily, resulting in unsatisfactory overlapping communities. Our proposed LEPSO and the work in [36]–[38] are randomized techniques that are based on link clustering algorithms. On the other hand, LEPSO, GA-NET+, and GaoCD are based on swarm intelligence technique, while the method in [36] is random-walk-driven. It is worth pointing out that different from GA-NET+ and GaoCD, LEPSO introduces ensemble learning technique, post-processing strategies, and swarm-intelligence-driven optimization, to improve quality of the resulting overlapping communities. Compared to the existing work, our proposed LEPSO exploits excellent global optimization ability of PSO and outstanding local search ability of ensemble learning.

C. Discrete Particle Swarm Optimization

Discrete particle swarm optimization (DPSO) technique, inspired by collective social behaviors in nature, tries to solve a discrete optimization problem by generating a population of particles, where each particle represents a candidate solution and can move around in search space according to some update rules that control position and velocity [21].

Consider an unconstrained optimization problem in a d-dimensional space, where $X_i = (X_{i1}, X_{i2}, \ldots, X_{id})$ and $V_i = (V_{i1}, V_{i2}, \ldots, V_{id})$ are d-dimensional vectors denoting position and velocity of the i-th particle, respectively. Let $pbest_i$, $i = 1, 2, \ldots, P_{size}$, be the best solution found by the i-th particle, and $gbest$ the best solution found across the whole particle swarm so far. Self-learning and cooperative learning of particles are achieved by updating $pbest_i$ and $gbest$. In each iteration, PSO updates the velocity and position of a particle according to the following equations:

$$V_i(t+1) = w \times V_i(t) + c_1 \times rand_1 \times (pbest_i - X_i(t)) + c_2 \times rand_2 \times (gbest - X_i(t)) \quad (1)$$

$$X_{ij}(t+1) = \begin{cases} 1, & \text{if } \rho < sig(V_{ij}(t+1)) \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

$$sig(V_{ij}(t+1)) = 1/[1+\exp(V_{ij}(t+1))] \quad (3)$$

where t denotes the t-th iteration, w inertia coefficient, c_1 and c_2 the learning rates, $rand_1$ and $rand_2$ random numbers uniformly generated in [0, 1], and ρ a predefined threshold.

Equation (3) is a transfer function that plays an important role in DPSO, since it defines the probability of changing the elements of position vector from 0 to 1, or 1 to 0, thus forcing the particles to move within a binary space. Transfer functions are roughly classified into two types, i.e., the s-shaped and the v-shaped [39]. In general, particle position update rules based on s-shaped transfer functions usually force particles to take on either 0 or 1, whereas update rules based on the v-shaped do not force particles to do so. Instead, they encourage particles to 1) stay in their current positions when their velocity is low, or 2) switch to their complements when velocity is high. There are some suggested guidelines for selecting a proper transfer function [39].

Recently, many variants of classic DPSO have been proposed, for instance, MDPSO [40] with genotype-phenotype representation and mutation operator, AIS-based DPSP [41] using artificial immune system, S-PSO [42] with a set-based representation scheme, and Catfish BPSO [43]. These variants have improved the search ability of classic DPSO to some degree, but they usually have high computational cost. For example, the mutation operator in MDPSO can lead to very high CPU overhead, thus not suitable for communities detection problems that are NP-hard in nature.

III. Definitions, Theorems, and Corollaries

In this section, we give some definitions, theorems, and corollaries needed in the paper.

A graph is defined as $G = \langle N, E\rangle$, with N and E being the set of vertices and edges, respectively. Each edge is represented by an unordered pair of vertices. The neighbor set $Nb(n)$ of a vertex $n \in N$ is defined as the set of vertices incident to n, and degree $deg(n)$ of n is the cardinality of $Nb(n)$, i.e., $deg(n) = |Nb(n)|$.

Definition 1 (overlapping communities): Suppose $G = \langle N, E\rangle$ is the corresponding graph of a network NET. If set $C = \{C_1, C_2, \ldots, C_m\}$ satisfies (1) $\forall C_i \subseteq N$, (2) $\forall C_i, C_i \neq \emptyset$, (3) $\exists C_i, C_j, (C_i \neq C_j) \wedge (C_i \cap C_j \neq \emptyset)$, and (4) $\bigcup_{i=1}^{m} C_i = N$, then C is a set of overlapping communities of network NET.

Definition 2 (disjoint communities): Suppose $G = \langle N, E\rangle$ is the corresponding graph of a network NET. If set $C = \{C_1, C_2, \ldots, C_m\}$ satisfies (1) $\forall C_i \subseteq N$, (2) $\forall C_i, C_i \neq \emptyset$, (3) $\forall C_i, C_j, (C_i \neq C_j) \wedge (C_i \cap C_j = \emptyset)$, and (4) $\bigcup_{i=1}^{m} C_i = N$, then C is a set of disjoint communities of network NET.

Based on the above definitions, it is clear that disjoint community structure is a partition of the node set of a network, whereas overlapping community structure is a cover of the node set.

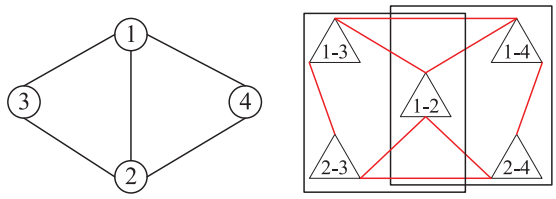

Fig. 3. Example graphs for Theorem 3. (a) Graph G, (b) line graph $L(G)$.

Definition 3 (overlapping communities refinement): Suppose $C = \{C_1, C_2, \ldots, C_m\}$ and $C' = \{C'_1, C'_2, \ldots, C'_n\}$ are two sets of overlapping communities of network NET. If for each C_i there always exists a C'_j satisfying $C_i \subseteq C'_j$, then C is called an overlapping communities refinement of C'.

Definition 4 (community overlapping rate): For any two distinct communities $C_i, C_j \in C$, the community overlapping rate between C_i and C_j is defined as $COR(C_i, C_j) = \frac{|C_i \cap C_j|}{\min(|C_i|, |C_j|)}$.

Definition 5 (line graph): Given a graph G, its line graph $L(G)$ is a graph such that each vertex in $L(G)$ corresponds to an edge in G. And two vertices in $L(G)$ are adjacent if and only if their corresponding edges are adjacent in G.

Theorem 1: If G is connected, then $L(G)$ is also connected.

Proof: If G is connected, then it contains a path connecting any two edges in G, meaning that for any two vertices in $L(G)$, there always exists a path connecting the two vertices. Thus, $L(G)$ is also connected.

Theorem 2: A partition of the vertex set of $L(G)$ corresponds to a cover of the vertex set of G.

Proof: Suppose $C = \{C_1, C_2, \ldots, C_m\}$ is a partition of the vertex set of $L(G)$, $N_i = \{n_{i1}, n_{i2}, \ldots, n_{i|N_i|}\}$ is the set of vertices of C_i. From Definition 5, to prove Theorem 2 we need to first prove that there always exist C_i and C_j such that $N_i \cap N_j \neq \emptyset$ holds.

From Definition 1 we know that there exist C_i and C_j, such that there is at least one edge connecting a vertex in C_i and a vertex in C_j. Without loss of generality, let $e = \langle n_{ia}, n_{jb}\rangle$ be an edge between C_i and C_j. By the definition of line graph, we know that a vertex in $L(G)$ corresponds to an edge in G, which means $e = \langle (n_{i1}, n_{i2}), (n_{j1}, n_{j2})\rangle$. Meanwhile, an edge in $L(G)$ corresponds to a vertex in G. Therefore, for the four possible conditions $n_{i1} = n_{j1}, n_{i1} = n_{j2}, n_{i2} = n_{j1}$ and $n_{i2} = n_{j2}$, only one of them can be true. This means that $N_i \cap N_j \neq \emptyset$ must hold. Therefore, we have proven that a partition of the vertex set of $L(G)$ exactly corresponds to a cover of the vertex set of G. ■

Corollary 1: Overlapping communities detection problem in G is equivalent to disjoint communities detection problem in $L(G)$.

Theorem 3: Suppose $C = \{C_1, C_2, \ldots, C_m\}$ is a vertex cover of G. If there exist C_i and C_j, $1 \leq i, j \leq m$, such that vertices in $C_i \cap C_j$ are adjacent, then it is impossible to transform cover C into a partition of $L(G)$.

Proof: We prove the theorem by contradiction. Suppose $C = \{C_1, C_2\}$, where $C_1 = \{1, 2, 3\}$ and $C_2 = \{1, 2, 4\}$ is a vertex cover of graph G, as shown in Fig. 3. Since $C_1 \cap C_2 = \{1, 2\}$

and vertex 1 and 2 are adjacent, we conclude that cover C cannot be transformed into a partition of $L(G)$.

Suppose the theorem is not true. Based on the facts that 1) a vertex in a partition of $L(G)$ corresponds to an edge with end vertices in a cover of G, and 2) vertex 1 and 2 in G are attached to C_1 and C_2 respectively, we find that vertex '1-2' in $L(G)$ belongs to two different parts, which contradicts the definition of partition. Hence, Theorem 3 must be true.

From theorem 3 we can see that if an optimal cover of G satisfies the conditions in Theorem 3, then it is impossible to directly compute the optimal cover through a partition of $L(G)$. Hence, we have the following corollary. ■

Corollary 2: Disjoint communities obtained by partitioning the vertex set of $L(G)$ are finer-grained and suboptimal overlapping communities of G.

IV. The Proposed Algorithm LEPSO

According to Corollary 2, to detect overlapping communities in a network, we just need to detect disjoint communities in the corresponding line graph. In this section, we present an improved DPSO, named LEPSO, to optimize partition result of the line graph.

A. Representation of Communities

Common encoding schemes of DPSO include integer encoding and binary encoding [44]. The locus-based adjacency representation scheme is a popular integer encoding scheme to represent communities in a network, described as follows.

Given line graph $L(G) =< N, E >$, where $N = (n_1, n_2, \ldots, n_k)$, a partition of $L(G)$ can be represented as a particle $X_i = (X_{i1}, X_{i2}, \ldots, X_{id})$, where $k = |N|$. If $X_{ij} = m$, then there exists an edge $e = \langle n_j, n_m \rangle$ in the communities corresponding to particle X_i, that is, vertex n_j and n_m are in the same community in $L(G)$.

This representation scheme, however, has a drawback, i.e., randomness in particle initialization and particle position update procedure makes it difficult to avoid producing illegal particles. In other words, edges represented by some components of a particle may not exist in the network at all. To overcome this shortcoming, we propose a novel representation scheme, which represents a particle based on ordered neighbor list. The key idea of our scheme is to utilize distribution information of the neighbors of each vertex, so as to guarantee legality of newborn particles produced during initialization or moving. We exemplify this below.

Example 2: Suppose G is a network depicted in Fig. 4(a). An example particle P encoded by locus-based adjacency representation scheme is shown in Fig. 4(b), where edges $<3, 6>$, $<5, 1>$ and $<7, 2>$ in particle P do not exist in G. So, P is an illegal particle. In fact, we can create an ordered neighbor list of all vertices as shown in Fig. 4(e). Based on the list we can use our proposed scheme to represent a partition [see Fig. 4(d)] of G as a legal one, as shown in Fig. 4(c).

Compared to traditional locus-based adjacency representation schemes, our representation scheme has several advantages, such as elimination of illegal particles completely, avoidance of generating local optimal communities obtained through iterative bipartition strategy [11], and determining the number of communities automatically.

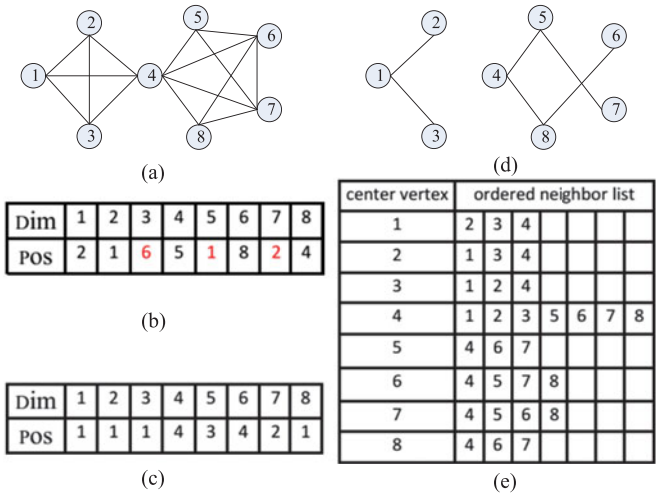

Fig. 4. Encoding particle based on ordered neighbor list. (a) Network G; (b) illegal particle; (c) particle encoded by LEPSO; (d) generated communities; (e) ordered neighbor list."Dim" refers to dimension, and "Pos" refers to position.

B. Particle Fitness

The concept of community in a network is not rigorously defined since its definition depends on the application domain of interest. The subjectivity of community definition encourages researchers to put forward various quality indices to evaluate the goodness of a partition, among which the most famous one is modularity [6]. The idea underlying modularity is that a random graph has no obvious cluster structure, thus edge density of a cluster should be higher than the expected density of a subgraph whose nodes are connected at random. Formally, modularity can be defined as

$$fit\,(P_i) = Q\,(C) = \sum_{c=1}^{m} \left[\frac{l_c}{|E|} - \left(\frac{d_c}{2\,|E|} \right)^2 \right] \tag{4}$$

where $fit(P_i)$ is the fitness value of particle P_i, m the number of communities in partition C of a network $G =< N, E >$, l_c the number of edges connecting vertices in a community $c \in C$, d_c sum of degree of the nodes in c, and $|E|$ the total number of edges in G.

C. Update Particle Velocity and Position

1) Update Particle Velocity: From (1) we can see that $gbest$ has a great impact on search ability of particle swarm, since it acts as the leader of the swarm and every particle learns from it in each iteration. When $gbest$ falls into a local optimum, there is a high possibility that the whole swarm is trapped in the local optimum too. Traditional methods to update particle velocity are generally useful in optimization, however those methods may not fully capture cluster structure information of a network, as demonstrated below.

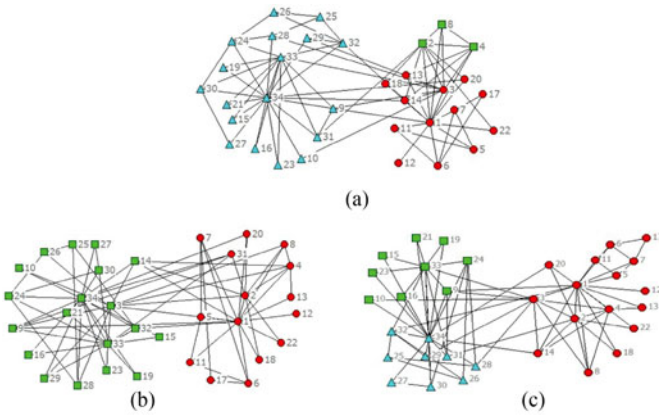

Fig. 5. $gbest$ and suboptimal particle. T denotes the Tth iteration during optimization using DPSO, Q denotes modularity of the Zachary network communities.

Example 3: Let there be a a set of particles generated by DPSO algorithm running on Zachary network, and it consists of the top 10 particles sorted in descending order on fitness value. And three particles (**a**, **b** and **c**) are randomly chosen from the particle set and decoded into three partitions, as shown in Fig. 5(a)–5(c), respectively. The real partition of the network is depicted in Fig. 2(a).

Comparing Fig. 5(a) with Fig. 5(b), we know that although particle b is $gbest$ and has a better fitness value than the suboptimal particle a, the community consisting of the red circles in particle a are more consistent with the real partition than particle b. Similarly, Figs. 5(c) and 2(a) indicate that although fitness value of particle c is smaller than that of particle a and that of particle b, it still fails to find a real community depicted in red circles in the network. From Example 3 we know that although some suboptimal particles in the swarm do not have the best goodness value, there are still some perfect community structures hidden in these suboptimal particles.

To escape from a local optimum, we propose a novel particle velocity update algorithm, named $GbestGenerator$, which adopts voting-based ensemble clustering technique [45] to make full use of the valuable cluster patterns hidden in $gbest$ s and the suboptimal particles. Specifically, if fitness value of $gbest$ does not improve in successive $T_{\max}$ iterations, i.e., the particle swarm has been trapped in prematurity, then we construct a member particle set *MPS* by selecting all the $gbest$ particles in the successive $T_{\max}$ iterations and the particles in the $T_{\max}$ iteration, then combine the particles in *MPS* to generate a new $gbest$ particle.

Equation (1) also suggests that the inertia coefficient w is very important for particle velocity update. Here we adopt a simple self-adaptive strategy to adjust w by using the follow formula:

$$w_t = (w_{\max} - w_{\min}) \frac{t_{\max} - t}{t_{\max}} + w_{\min} \tag{5}$$

where $w_{\max}$ and $w_{\min}$ are the initial and final inertia coefficient respectively, $t_{\max}$ the maximum iteration and t the current iteration.

From (5), we can see that at the initial state $t = 0$, w_t corresponds to $w_{\max}$, and when t approaches $t_{\max}$, w_t gradually decreases to $w_{\min}$. Rationale behind this is that, like traditional PSOs, LEPSO will gradually converge to an optimal point in solution space while the particle swarm is evolving. Hence, in early stages larger coefficient values are required so that particles can have higher speed, whereas in later stages smaller coefficient values are given to particles so as to make them stable gradually.

Algorithm 1 GbestGenerator

Input: member particle set *MPS*
Output: $gbest$
1: Build partition set $PSet = \{P_1, P_2, \ldots, P_{|PSet|}\}$ by decoding particles in MPS;
2: Remove duplicate partitions in $PSet$ by comparing the partitions corresponding to particles with equal fitness;
3: Re-order partitions in $PSet$ in decreasing order on $H(P_i) = \sum_{C_j \in P_i} p(C_j) \log p(C_j)$
4: Represent each partition $P_i \in PSet$ by a matrix MP_i, where a row corresponds to a vertex and a column to a community, and $MP_i \in MPSet$;
5: $MP_0 = MP_1$;
6: **for** i = 2 to $|MPSet|$ **do**
7: $W(i) = \left({MP_i}^T MP_i\right)^{-1} {MP_i}^T MP_0$
8: $V(i) = MP_i W(i)$
9: $MP_0 = \frac{i-1}{i} MP_0 + \frac{1}{i} V(i)$
10: **end for**
11: Encode partition MP_0 as particle $gbest$;
12: Return $gbest$.

2) Update Particle Position: According to (2), we observe that components of particle position vector are assigned either 1 or 0. Obviously, this assignment method is not suitable for our proposed particle representation scheme that is based on ordered neighbor list. In our representation scheme, value X_{ij} of component i is an integer, ranging from 1 to $deg(n_i)$, i.e., $X_{ij} \in \{1, 2, \ldots, \deg(n_i)\}$. To improve the search ability of particle swarm, we propose a new method to update particle position, which can be formulated in the two equations below:

$$X_{ij}(t+1) = \begin{cases} k, & \text{if } (\rho < sig(V_{ij}(t+1))) \wedge (\deg(n_j) > 1) \\ X_{ij}(t), & \text{otherwise} \end{cases} \tag{6}$$

$$sig(V_{ij}) = \left| \frac{1 - \exp(-V_{ij})}{1 + \exp(-V_{ij})} \right| \tag{7}$$

where $k = \lceil rand * \deg(n_j) \rceil$, $k \neq X_{ij}(t)$, $\deg(n_j)$ is the degree of vertex n_j, and ρ a threshold specified by the user.

Comparing (6) with (2) for updating particle position, we find that the major difference between our proposed method and DPSO is how to replace a binary value with a random positive integer. Our method generates a position value according to node degree distribution, i.e., to replace a neighbor of a node

Algorithm 2 HABM

Input: finer-grained overlapping communities OP
Output: optimal overlapping communities OOP
1: For any community pair (C_i, C_j), $C_i, C_j \in OP$, compute $COR(C_i, C_j)$;
2: Select the community pair with maximal COR to merge into a new community C_{new} and delete the others included in C_{new};
3: Repeat step 1-2 until all vertices in the same community and get a final overlapping communities tree OCT;
4: Compute Q_{ov} for each community at each level of OCT;
5: Return community structure with maximal Q_{ov}.

with currently chosen neighbor if a node v_j has more than one neighbor and $sig\left(v_{ij}\left(t+1\right)\right)$ is greater than ρ. Obviously, (6) can greatly enhance the global search ability of the particle swarm, but its local search ability may degenerate. To solve this problem, we modify the sigmoid function $sig()$ in (6) to force the probability of particle changing its position to decrease with the particle velocity, so as to make particle swarm gradually converge to the global optimum.

D. Overlapping Community Structure Optimization

Corollary 2 indicates that we should further refine the finer-grained and suboptimal overlapping communities of G, generated by partitioning vertices in $L(G)$, in order to get more optimal overlapping communities. Recently, some measures have been proposed to evaluate modularity of overlapping communities hidden in a network [31], and a popular modularity function Q_{ov} proposed in [46] are given in (8) and (9) below:

$$Q_{ov} = \frac{1}{2m}\sum_{c\in P}\sum_{i,j}\frac{1}{O_i O_j}\left(A_{ij} - \frac{k_i k_j}{2m}\right)\delta_{ic}\delta_{jc} \tag{8}$$

$$\delta_{ic} = \begin{cases} 1, & \text{if node } i \text{ is contained by community } c \\ 0, & \text{otherwise} \end{cases} \tag{9}$$

where P is a cover of G, m the number of edges in G, k_i the degree of node i, A the adjacent matrix of G, O_i the number of communities containing node i.

To post-process finer-grained and suboptimal overlapping communities, we resort to hierarchical clustering and propose a hierarchical agglomerative and bottom-up merging strategy, namely HABM. Our HABM algorithm, given in Algorithm 2, differs from traditional hierarchical clustering algorithms in that HABM chooses to merge community pair with the maximal community overlapping rate, instead of the one with the maximal similarity.

E. Algorithm Description

Our LEPSO method can be described as follows. First, we initialize parameters needed. Next, we search for the optimal partition of line graph $LG(G)$ with an improved DPSO. Then, we transform the result partition of $LG(G)$ into a cover of graph G. Finally, we perform hierarchical merge to generate the optimal overlapping communities. The procedure of LEPSO is given in Algorithm 3.

Algorithm 3 LEPSO

Input: social network G
Output: overlapping communities of G
1: Transform G into $LG(G)$; build the ordered neighbor list L;
2: $k = 0$; $MPS = \emptyset$;
3: Initialize particle swarm P^{k+1} based on L;
4: $fit\left(gbest^k\right) = -\infty$;
5: $fit\left(pbest_i^k\right) = -\infty,\ i = 1, 2, \ldots, m$;
6: Evaluate particles in P^{k+1} with (4) ;
7: **for** $P_i^k \in P^k$ **do**
8: **if** $fit\left(P_i^{k+1}\right) > fit\left(pbest_i^k\right)$ **then** $pbest_i^k = P_i^k$;
9: $gbest^k = \arg\max_{P_i^k \in P^k} fit\left(P_i^k\right)$
10: **if** $fit\left(gbest^k\right) = fit\left(gbest^{k+1}\right)$ **then** $MPS = MPS \cup gbest^k$;
11: **if** $fit\left(gbest^k\right) < fit\left(gbest^{k+1}\right)$ **then** $MPS = \emptyset$;
12: **if** $fit(gbest)$ not improved in successive $T_{\max}$ iterations **then**
13: $MPS = MPS \cup P^{k+1}$;
14: $gbest^k = GbestGenerator(MPS)$;
15: Update P^{k+1} to P^{k+2} using (1), (5) and (6);
16: $k = k + 1$;
17: **Repeat** from Step 6 to 16 **until** $k > t_{\max}$;
18: Get partition HP of $LG(G)$ that corresponds to $gbest$; transform HP into a cover CP of G;
19: **Return** HABM(CP)

TABLE I
DESCRIPTION OF THE EIGHT DATASETS USED

Network	Nodes	Edges	Data source
Zachary	34	78	http://www-personal.umich.edu/ mejn/netdata/
Dolphins	62	159	http://www-personal.umich.edu/ mejn/netdata/
Football	115	616	http://www-personal.umich.edu/ mejn/netdata/
Email	1133	5452	http://snap.stanford.edu/data/email-Enron.html
SynNet_1	400	2014	LFR benchmark [51]
SynNet_2	400	1782	LFR benchmark [51]
SynNet_3	400	1240	LFR benchmark [51]
SynNet_4	400	1209	LFR benchmark [51]

V. EXPERIMENTAL STUDY

In this section, we conduct extensive experiments to verify the effectiveness of our LEPSO algorithm.

Datasets: We use eight datasets, four real and four synthetic, to evaluate our LEPSO algorithm. Table I shows statistics of the datasets used. The four real-world datasets are well-known benchmark networks in communities detection research. Specifically, Zachary is a social network of friendships between 34 members of a karate club at a US university in the 1970s. Dolphins is an undirected social network of frequent associations between 62 dolphins in a community living off Doubtful Sound, New Zealand. Football is a network of American football games

TABLE II
PARAMETERS SETTING

Parameter	Description of parameter	Value
P_{size}	population size	50
$t_{\max}$	iteration number	1000
$T_{\max}$	number of successive generations in which fitness of gbest has not been improved	20
ρ	predefined threshold	0.75
$w_{\min}$	final inertia coefficient	0.6
$w_{\max}$	initial inertia coefficient	1.5

between Division IA colleges during regular season of Fall 2000. And Email is a network established by receiving and sending emails, in which each node represents an email address and two nodes are connected when they have email exchanges in history.

Platform: All the experiments are conducted on a PC with a 3.4 GHz Intel(R) Core(TM) i7-2600 CPU and 8 GB RAM, running Windows 7. The result is averaged over 50 trials. The parameter settings for LEPSO are given in Table II.

Parameter initialization: In swarm intelligence computation [53], it is well-known that how to determine and adjust algorithm parameters, including population size, final inertia coefficient, initial inertia coefficient etc, is an open problem. Although parameter tuning is out of the scope of this paper, we give the rationale of why choosing the initial values as shown in Table II. Basically, we conducted extensive preliminary experiments on synthetic networks with different size, modularity values and overlappingness, and we observed that although for different networks there are different parameter settings producing high-quality clusters, the parameter setting in Table II can produce better overlapping community structures in most cases.

Metrics: To evaluate performance of algorithms for overlapping communities detection, we adopt the commonly used normalized mutual information(NMI) [30], given in (10)

$$\mathrm{NMI}\left(P^{res}, P^{true}\right) = \frac{-2\sum_{N_m \in P^{res}}\sum_{N_n \in P^{true}} \frac{|N_m \cap N_n|}{|N|}\log\left(\frac{|N||N_m \cap N_n|}{|N_m||N_n|}\right)}{\sum_{N_m \in P^{res}} \frac{|N_m|}{|N|}\log\left(\frac{|N_m|}{|N|}\right) + \sum_{N_n \in P^{true}} \frac{|N_n|}{|N|}\log\left(\frac{|N_n|}{|N|}\right)} \tag{10}$$

where P^{res} is the overlapping communities generated, P^{true} the true community structure of the network, $|N|$ the number of nodes in N, N_m the m-th community in P^{res}, N_n the n-th community in P^{true}. It is worth noting that when computing NMI using (10), if node set N_m in the generated community structure and node set N_n in the true community structure are disjoint, we discard the result.

A. Quality of Generated Overlapping Communities

In this section, we compare LEPSO with four representative competitors, including two non-randomized algorithms CPM [52] and ABL [35], and four randomized algorithms BMLPA [47], BNMTF [48], EPM [49] and MCMOEA [50], with respect to NMI score. To the best of our knowledge, CPM is the first algorithm for detecting overlapping communities, and the other five are the state-of-the-art. CPM, ABL and EPM can be downloaded from *angel.elte.hu/clustering*, *barabasilab.neu.edu/projects/linkcommunities/* and *github.com/mingyuanzhou/EPM*, respectively. The other three competitors MCMOEA, BMLPA and BNMTF, and our LEPSO are implemented in java 1.7 and Eclipse 4.5.

Table III shows the quality of detected overlapping communities in the eight datasets. For the sake of faireness, each result is averaged over 50 trials for the randomized algorithms BMLPA, BNMTF, EPM, MCMOEA and LEPSO. It is worth pointing out that parameter k in CPM is the size of the rolling clique, parameter t in ABL is the edge similarity threshold, parameter p in BMLPA is coefficient threshold, and parameter k in BNMTF is the maximum value for the rank. In Table III the best result is achieved with the parameter value given in brackets.

From Table III we can see that comparing to non-randomized algorithms CPM and ABL, LEPSO can produce much higher mean NMIs on both real and synthetic datasets. On the other hand, comparing to randomized algorithms BMLPA, BNMTF, EPM and MCMOEA, all the max mean NMIs are achieved by LEPSO on the eight datasets, meaning that our approach performs better than other randomized ones consistently. Moreover, among the four randomized algorithms, LEPSO generates nearly all the minimal standard deviations of NMI (sdNMIs), except for datasets Email and Football, which means that our approach exhibits higher stability in general than the other randomized ones. When comparing to two state-of-the-art algorithms EPM and MCMOEA, LEPSO performs better in terms of meanNMI and sdNMI, although the performance gain is not as significant as LEPSO outperforming the other two random algorithms. Last but not least, although the quality, measured by NMI score, of overlapping communities detected by LEPSO has small fluctuations, it does not show significant deviations with network size. Based on the above observations, our LEPSO significantly outperforms both non-randomized and randomized algorithms for overlapping communities detection. Meanwhile, LEPSO shows better stability than the two randomized algorithms, and scales up well with network size. In the following two subsections, we investigate the reason why LEPSO consistently outperforms its competitors.

1) Impact of Merging Strategy on Community Quality: In this section, we verify effectiveness of the proposed merging strategy on network Zachary and Dolphins. From Fig. 6(b) we can see that there are 3 overlapping communities in Zachary before applying the merging strategy, where nodes 3,9,10 and 31 are sharable nodes, and the corresponding Q_{ov} score and NMI score are 0.222 and 0.823, respectively. After applying the merging strategy, we have the final overlapping community structure depicted in Fig. 6(c), where there is only one sharable node 3. The number of overlapping communities decreases to 2, and the corresponding Q_{ov} score and NMI score increase to 0.2313 and 0.9028, respectively. We observe a similar trend on Dophins, by examining the results in Fig. 7.

From Figs. 6(b) and 7(b), we can see that some overlapping community structures with superfluous small communities

TABLE III
COMPARISON IN TERMS OF NMI

Type	Datasets	CPM	ABL	EPM	BMLPA	BNMTF	MCMOEA	LEPSO
Real-life	Zachary	0.335 (k = 3)	0.409 (t = 0.15)	0.857 ± 0.015	0.347 ± 0.014 (p = 0.7)	0.346 ± 0.015 (k = 2)	0.885 ± 0.007	0.906 ± 0.005
	Dolphins	0.461 (k = 3)	0.209 (t = 0.25)	0.782 ± 0.021	0.649 ± 0.013 (p = 0.75)	0.602 ± 0.017 (k = 3)	0.816 ± 0.011	0.823 ± 0.008
	Football	0.516 (k = 3)	0.527 (t = 0.25)	0.765 ± 0.014	0.765 ± 0.034 (p = 0.75)	0.684 ± 0.025 (k = 5)	0.831 ± 0.023	0.852 ± 0.016
	Email	0.316 (k = 3)	0.345 (t = 0.25)	0.751 ± 0.026	0.645 ± 0.031 (p = 0.75)	0.591 ± 0.028 (k = 5)	0.807 ± 0.025	0.815 ± 0.027
Artificial	SynNet_1	0.381 (k = 4)	0.353 (t = 0.2)	0.687 ± 0.033	0.697 ± 0.047 (p = 0.75)	0.251 ± 0.042 (k=8)	0.694 ± 0.044	0.713 ± 0.029
	SynNet_2	0.348 (k = 4)	0.338 (t = 0.2)	0.698 ± 0.044	0.684 ± 0.043 (p = 0.75)	0.257 ± 0.048 (k = 8)	0.701 ± 0.036	0.717 ± 0.034
	SynNet_3	0.288 (k = 4)	0.382 (t = 0.18)	0.691 ± 0.057	0.678 ± 0.048 (p = 0.75)	0.284 ± 0.047 (k = 8)	0.708 ± 0.051	0.718 ± 0.031
	SynNet_4	0.275 (k = 4)	0.329 (t = 0.18)	0.712 ± 0.051	0.693 ± 0.052 (p = 0.75)	0.312 ± 0.058 (k = 8)	0.715 ± 0.049	0.724 ± 0.032

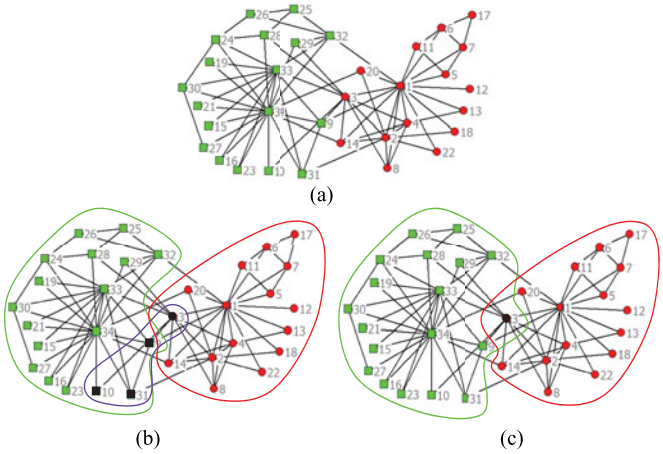

Fig. 6. Impact of merging strategy on quality of communities resulted from Zachary. (a) Real community structure. (b) Initial result overlapping communities without merging (Q_{ov} = 0.222, NMI = 0.823). (c) Final overlapping communities (Q_{ov} = 0.2313, NMI = 0.9028).

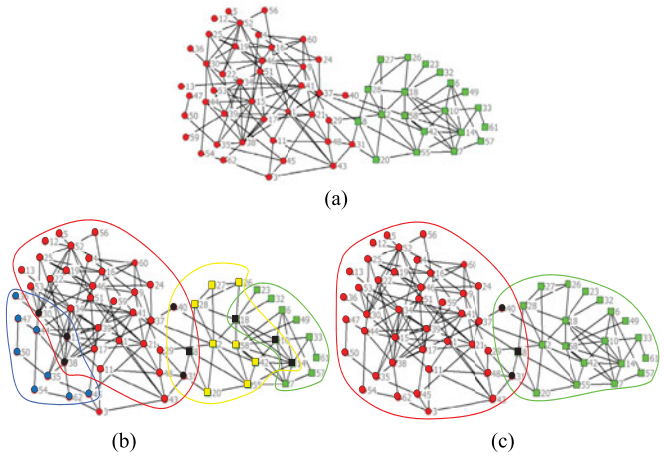

Fig. 7. Impact of merging strategy on quality of communities resulted from Dolphins. (a) Real community structure. (b) Initial result overlapping communities without merging (Q_{ov} = 0.156, NMI = 0.47). (c) Final overlapping communities (Q_{ov} = 0.318, NMI = 0.824).

may be produced during searching optimal overlapping communities, which may degrade community quality if not handling properly. Through our proposed post-processing strategy, finer-grained overlapping communities in Zachary [Fig. 6(c)] and Dolphins [Fig. 7(c)] are transformed into larger overlapping communities, as shown in Figs. 6(b) and 7(b) respectively. Therefore, post-processing finer-grained overlapping communities is beneficial to enhance community quality. This is also an experimental verification for Corollary 2.

There are two crucial questions about the merging strategy. First, is the proposed merging strategy HABM an universal and effective post-processing procedure? Second, can HABM boost quality of the overlapping community structures generated by any other approaches? To answer these questions, we use HABM to post-process overlapping communities generated by the other six competitors CPM, ABL, EPM, MCMOEA,BMLPA and BNMTF. The results are summarized in Table IV.

From Table IV we can see that almost all entries in column Delta are non-zero and the majority of maximal values in Delta achieved by randomized algorithms. This means that our proposed merging strategy can enhance quality of overlapping communities generated by the competitors to some extent. And HABM is more beneficial to randomized algorithms than to non-randomized algorithms CPM and ABL. Moreover, results in column BeforeMerging show that LEPSO outperforms the other six counterparts in terms of NMI, even without post-processing. This also verifies that it is advantageous to use line graph presentation and ensemble learning technique $GbestGenerator$. Hence, we conclude that merging strategy HAMB is beneficial to a wide range of approaches, especially to randomized ones, for detecting high-quality overlapping communities.

2) Impact of Ensemble Learning Strategy on Community Quality: In this section we evaluate the impact of ensemble learning strategy on quality of the generated overlapping communities. Since the influence of post-processing strategy HABM varies for different algorithms, for the sake of fairness we first remove HABM and ensemble learning component from LEPSO, resulting in a simplified LEPSO called LEPSO-1. Meanwhile, we also build another simplified LEPSO named LEPSO-2, by removing HABM from LEPSO only. The experiment results of LEPSO-1 and LEPSO-2 are depicted in Fig. 8, from which we can see that LEPSO-2 outperforms LEPSO-1 on both real and synthetic datasets. This indicates that integrating ensemble learning strategy into discrete particle swarm optimization can effectively boost the search ability for high-quality overlapping communities.

Based on observations in Section V-A.1 and V-A.2, we can find that 1) adoption of ensemble learning technique can help LEPSO to avoid trapping in local optimal partition of the line

TABLE IV
IMPACT OF MERGING STRATEGY ON COMMUNITY QUALITY IN TERMS OF NMI

Datasets	Algorithms	BeforeMerging	AfterMerging	Delta
Zachary	CPM	0.335	0.335	0
	ABL	0.409	0.894	**0.485**
	BMLPA	0.347	0.393	0.046
	BNMTF	0.346	0.38	0.034
	EPM	0.857	0.865	0.008
	MCMOEA	0.885	0.901	0.016
	LEPSO	0.825	0.906	0.081
Dolphins	CPM	0.461	0.461	0
	ABL	0.209	0.269	0.06
	BMLPA	0.649	0.73	0.081
	BNMTF	0.602	0.655	0.053
	EPM	0.782	0.814	0.032
	MCMOEA	0.816	0.821	0.005
	LEPSO	0.712	0.823	**0.112**
Football	CPM	0.516	0.535	0.019
	ABL	0.527	0.582	0.055
	BMLPA	0.765	0.793	0.028
	BNMTF	0.684	0.741	0.057
	EPM	0.765	0.836	0.071
	MCMOEA	0.831	0.844	0.011
	LEPSO	0.795	0.852	**0.093**
Email	CPM	0.316	0.408	0.092
	ABL	0.345	0.427	0.082
	BMLPA	0.645	0.783	0.138
	BNMTF	0.591	0.735	**0.144**
	EPM	0.751	0.797	0.046
	MCMOEA	0.807	0.813	0.006
	LEPSO	0.706	0.815	0.109
SynNet_1	CPM	0.381	0.405	0.024
	ABL	0.353	0.397	0.044
	BMLPA	0.697	0.708	0.011
	BNMTF	0.251	0.387	**0.136**
	EPM	0.687	0.691	0.004
	MCMOEA	0.694	0.708	0.006
	LEPSO	0.698	0.713	0.015
SynNet_2	CPM	0.348	0.405	0.057
	ABL	0.338	0.431	0.093
	BMLPA	0.684	0.702	0.018
	BNMTF	0.257	0.438	**0.181**
	EPM	0.698	0.701	0.003
	MCMOEA	0.701	0.715	0.014
	LEPSO	0.701	0.717	0.016
SynNet_3	CPM	0.288	0.363	0.075
	ABL	0.382	0.544	**0.162**
	BMLPA	0.678	0.691	0.013
	BNMTF	0.284	0.406	0.122
	EPM	0.691	0.694	0.003
	MCMOEA	0.708	0.72	0.012
	LEPSO	0.703	0.718	0.015
SynNet_4	CPM	0.275	0.352	0.077
	ABL	0.329	0.538	0.209
	BMLPA	0.693	0.715	0.022
	BNMTF	0.312	0.583	**0.271**
	EPM	0.712	0.717	0.005
	MCMOEA	0.715	0.725	0.01
	LEPSO	0.707	0.724	0.017

graph, and 2) introduction of merging strategy is also beneficial to boosting the quality of overlapping communities in the original graph. Hence, with ensemble learning technique and merging strategy combined, LEPSO can overwhelmingly outperform its competitors.

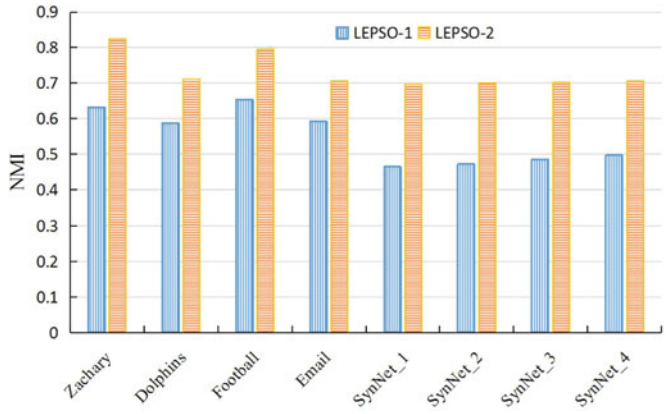

Fig. 8. Impact of ensemble learning strategy on communities quality.

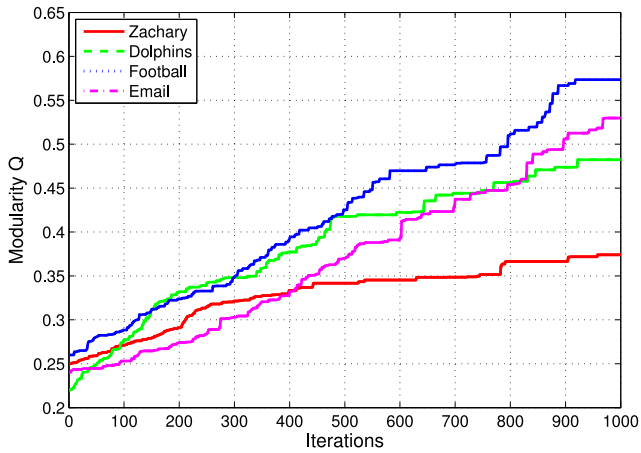

Fig. 9. Convergence rate of LEPSO.

B. Convergence of LEPSO

Essentially, LEPSO is a swarm-intelligence-based optimization method, and convergence rate is an important performance index for PSOs. So, it is meaningful to investigate convergence rate of LEPSO when searching for optimal community structures. To this end, we conduct experiments on the real datasets, and modularity Q is averaged over 50 trails. The results are presented in Fig. 9, from which we can see that for small networks like Zachary or large-scale networks like Email, LEPSO can effectively avoid trapping in local optimum. Also, by referring to Fig. 9 we find that LEPSO has a good convergence rate when optimizing modularity.

C. Sensitivity to Network Topology

Real-world complex networks are enormously diverse in terms of topological structure, and it is interesting to see how our approach performs on networks with various structures, e.g., different modularity, varied degrees of overlapping etc. In this section, we experimentally evaluate sensitivity of LEPSO to network topology.

To get networks with required properties for experiments, we use LFR generator to produce two types of networks, where for the first type we gradually decrease the modularity. Specifically, 5 groups of networks with modularity parameter [see (11)]

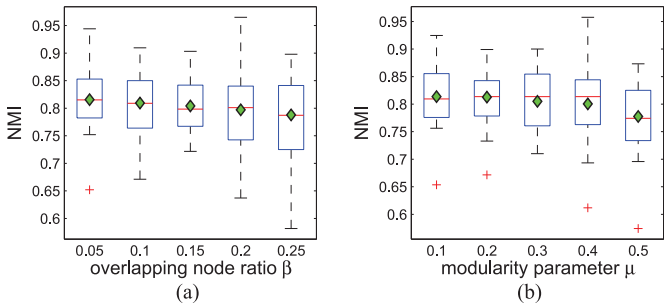

Fig. 10. Robustness of LEPSO. (a) W.r.t. degree of overlapping of networks; (b) w.r.t. network modularity.

$\mu = 0.1, 0.2, 0.3, 0.4$, and 0.5 are generated, respectively. For the second type of networks we gradually increase the number of overlapping nodes, where 5 groups of networks with overlapping node ratio [see (12)] $\beta = 0.05, 0.1, 0.15, 0.2$, and 0.25 are constructed respectively. Each of the 10 groups contains 50 networks with size $|N| = 200$. The results are shown in Fig. 10. Through (11) and (12), it is obvious that larger μ or β value will result in networks with weaker community structures.

$$\mu = \frac{|\{\, e =< n_i, n_j > | \, n_i \in C_{i_1} \wedge n_j \in C_{j_1} \wedge i_1 \neq j_1 \}|}{|E|} \tag{11}$$

$$\beta = \frac{|\{\, n_i | \, (\exists i_1)\,(\exists i_2)\,(n_i \in C_{i_1} \wedge n_i \in C_{i_2} \wedge i_1 \neq i_2) \}|}{|N|} \tag{12}$$

From (11) we can see that a larger modularity can make it more challenging to detect overlapping communities from networks. This is verified by Fig. 10(b), where $Hspread$, defined as $Hspread = UpperHinge - LowerHinge$, of each box increases gradually with μ, meaning that variances of modularity have some influences on stability of LEPSO. However, it is interesting that mean NMI scores (green diamonds in the boxes) of overlapping communities generated by LEPSO do not significantly decrease with μ. Instead, they fluctuate between 0.783 and 0.81. This indicates that in terms of quality of generated overlapping communities, LEPSO is robust against variances in modularity. As for overlapping node ratio, the experiment results are similar. As shown in Fig. 10(a), we can also see that a larger overlapping node ratio will give rise to higher degree of divergence in the distribution of NMI scores of the generated overlapping communities. On the other hand, larger overlapping node ratio will not produce significant differences, for instance, the mean NMI scores remain the same in [0.791, 0.817].

Therefore, based on the above observations we claim that network topology affects the performance of LEPSO to some extent, but does not degenerate LEPSO's ability to discover high-quality overlapping communities.

D. Time Efficiency and Robustness Analysis

Based on the previous experiment results, we can see that LEPSO performs much better than its competitors, except for two state-of-the-art algorithms EPM and MCMOEA, where EPM and MCMOEA can generate overlapping communities that are nearly as good as those generated by LEPSO. Therefore, in this section we further investigate performance of LEPSO, EPM, and MCMOEA with respect to CPU time and robustness.

For the sake of fairness, we first need to introduce a time measure. Obviously, the number of iterations cannot be used for this purpose, since the algorithms perform different amount of work in their inner loops. Hence, we choose the elapsed CPU time as a measure instead of number of iterations. Moreover, we use a user-specified maximum number of iterations as termination condition for the three algorithms. We record the running time of the three algorithms according to a rule like this: *if the termination condition of an algorithm has been satisfied but its NMI has not achieved the corresponding value in Table IV, then we do not record the running time. In other words, if its NMI has achieved the corresponding value in Table IV before the termination condition is met, then we break the loop and record the time used.*

TABLE V
COMPARISON IN TERMS OF RUNNING TIME (SEC.)

Dataset	EPM	MCMOEA	LEPSO
Zachary	0.97	1.05	1.03
Dolphins	3.16	3.25	3.07
Football	9.93	11.33	9.35
Email	105.47	113.42	92.08

The experiment results are listed in Table V, from which we can see that LEPSO spends similar CPU time as EPM and MCMOEA on networks with easily discernable community structures, such as Zachary and Dolphins. On the other hand, LEPSO is superior on networks with indistinguishable community structures such as Email. The reason is that for dataset with easily discernable community structures, any algorithm with fair optimization ability can efficiently discover community structures given a NMI value. Hence, the performance gain of LEPSO is not significant in general for this kind of dataset. However, for datasets with complex and ambiguous community structures, LEPSO performs much better. The rationale behind this is that for networks with ambiguous community structures, MCMOEA spends too much time on evaluating multiple objective functions and time cost of Gibbs sampling to estimate EPM parameters is also too expensive. From the above observations, we conclude that for datasets with either clear community structures or ambiguous community structures, LEPSO spends less CPU time than EPM and MCMOEA on complex networks.

We now turn to robustness of EPM, MCMOEA, and LEPSO, where we investigate their NMI values with varying network modularity. The experiment results are summarized in Table VI, from which we can see that compared to LEPSO, NMI values of EPM and MCMOEA drop significantly when network modularity increases. In other words, LEPSO is more robust than EPM and MCMOEA with respect to network modularity, meaning that LEPSO is more suitable for community detection tasks on networks with complex community structure.

TABLE VI
ROBUSTNESS (IN TERMS OF NMI) WITH RESPECT TO NETWORK MODULARITY

Algorithm	$\mu = 0.1$	$\mu = 0.2$	$\mu = 0.3$	$\mu = 0.4$	$\mu = 0.5$
EPM	0.796	0.782	0.744	0.727	0.705
MCMOEA	0.807	0.785	0.756	0.732	0.713
LEPSO	0.814	0.814	0.805	0.801	0.798

VI. CONCLUSION

In this paper, we propose a meta-heuristic algorithm, LEPSO, for overlapping communities discovery from social networks. Specifically, a particle representation scheme based on ordered neighbor list and a particle update strategy are proposed. Also, a hierarchical agglomerative and bottom-up merging strategy is designed to post-process the generated fine-grained overlapping communities. We conducted extensive experiments and the results show that 1) compared with the non-randomized and randomized algorithms, our LEPSO is superior in terms of validity and robustness, and 2) the proposed hierarchical agglomerative and bottom-up merging strategy can improve quality of the generated overlapping communities.

Our future studies can be divided into two directions. First, we will integrate DPSO with other optimization methods such as K-Means and spectral clustering to achieve better performance. Second, we will optimize population initialization strategy in LEPSO, so as to further improve efficiency in communities detection when handling with large-scale networks.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees and the associate editor for their valuable comments and suggestions, which have improved the paper vastly.

REFERENCES

[1] U. Gargi *et al.*, "Large-scale community detection on YouTube for topic discovery and exploration," in *Proc. Int. Conf. Weblogs Social Media*, 2011, pp. 486–489.
[2] M. Cheung, J. She, and Z. Jie, "Connection discovery using big data of user shared images in social media," *IEEE Trans. Multimedia*, vol. 17, no. 9 pp. 1417–1428, Sep. 2015.
[3] S. Huang, J. Zhang, L. Wang, and X.-S. Hua, "Social friend recommendation based on multiple network correlation," *IEEE Trans. Multimedia*, vol. 18, no. 2, pp. 287–299, Feb. 2016.
[4] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, pp. 75–174, 2010.
[5] M. Planti and M. Crampes, "Survey on social community detection," in *Social Media Retrieval*. London. U.K.: Springer, 2013, pp. 65–85.
[6] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Phys. Rev. E*, vol. 69, 2004, Art. no. 026113.
[7] F. L. Huang, S. C. Zhang, and X. F. Zhu, "Discovering network community based on multi-objective optimization," *Ruan Jian Xue Bao/J. Softw.*, vol. 24, no. 9, pp. 2062–2077, 2013.
[8] Q. Cai *et al.*, "A survey on network community detection based on evolutionary computation," *Int. J. Bio-Inspired Comput.*, vol. 8, no. 2, pp. 84–98, 2016.
[9] M. Tasgin and H. Bingol, "Community detection in complex networks using genetic algorithm," *CoRR*, 2007. [Online]. Available: http://arxiv.org/abs/0711.0491
[10] C. Pizzuti, "GA-NET: A genetic algorithm for community detection in social networks," *Parallel Problem Solving Nature*, vol. 5199, pp. 1081–1090, 2008.
[11] X. D. Duan, C. R. Wang, X. D. Liu, and Y. P. Lin, "Web community detection model using particle swarm optimization," in *Proc. Congr. Evol. Comput.*, 2008, pp. 1074–1079.
[12] M. Lipczak and E. Milios, "Agglomerative genetic algorithm for clustering in social networks," in *Proc. Genetic Evol. Comput.*, 2009, pp. 1243–1250.
[13] G. Bello, H. Menendez, and D. Camacho, "Using the clustering coefficient to guide a genetic-based communities finding algorithm," in *Proc. Intell. Data Eng. Automated Learn.*, 2011, pp. 160–169.
[14] M. Gong *et al.*, "A non-dominated neighbor immune algorithm for community detection in networks," in *Proc. Conf. Genetic Evol. Comput.*, 2011, pp. 1627–1634.
[15] C. Pizzuti, "Boosting the detection of modular community structure with genetic algorithms and local search," in *Proc. Symp. Appl. Comput.* 2012, pp. 226–231.
[16] R. Shang *et al.*, "Community detection based on modularity and an improved genetic algorithm," *Physica A, Statist. Mech. Appl.*, vol. 392, no. 5, pp. 1215–1231, 2013.
[17] J. Li and Y. Song, "Community detection in complex networks using extended compact genetic algorithm," *Soft Comput.*, vol. 17, no. 6, pp. 925–937, 2013.
[18] A. A. A. Esmin, R. A. Coelho, and S. Matwin, "A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data," *Artif. Intell. Rev.*, vol. 44, no. 1, pp. 23–45, 2013.
[19] M. Broilo and F. G. B. De Natale, "A stochastic approach to image retrieval using relevance feedback and particle swarm optimization," *IEEE Trans. Multimedia*, vol. 12, no. 4, pp. 267–277, Jun. 2010.
[20] H. H. Chou, L. Y. Hsu, and H. T. Hu, "Turbulent-PSO-based fuzzy image filter with no-reference measures for high-density impulse noise," *IEEE Trans. Cybern.*, vol. 43, no. 1, pp. 296–307, Feb. 2013.
[21] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm algorithm," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Oct. 1997, vol. 5, pp. 4104–4108.
[22] T. Mei *et al.*, "ImageSense: Towards contextual image advertising," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 8, no. 1, pp. 159–170, 2012.
[23] M. Wang *et al.*, "Assistive tagging: A survey of multimedia tagging with human-computer joint exploration," *ACM Comput. Surv.*, vol. 44, no. 4, 2012, Art. no. 25.
[24] A. Sureka *et al.*, "Mining YouTube to discover extremist videos, users and hidden communities," in *Proc. Asia Inf. Retrieval Symp.*, 2010, pp. 13–24.
[25] Z. Wang, D. Zhang, and X. Zhou, "Discovering and profiling overlapping communities in location-based social networks," *IEEE Trans. Syst. Man, Cybern., Syst.*, vol. 44, no. 4, pp. 499–509, Apr. 2014.
[26] R. L. Santos *et al.*, "Characterizing the YouTube video-sharing community," Federal Univ. Minas Gerais (UFMG), Belo Horizonte, Brazil, 2007. [Online]. Available: http://homepages.dcc.ufmg.br/~rodrygo/wp-content/papercite-data/pdf/santos2007report.pdf
[27] C. M. A. Yeung, N. Gibbins, and N. Shadbolt, "Contextualising tags in collaborative tagging systems," in *Proc. ACM Conf. Hypertext Hypermedia*, 2009, pp. 251–260.
[28] D. Tsatsou *et al.*, "Distributed technologies for personalized advertisement delivery," in *Online Multimedia Advertising: Techniques and Technologies*, X. C. S. Hua, T. Mei, and A. Hanjalic, Eds. Hershey, PA, USA: IGI Global, 2010, pp. 233–261.
[29] Y. L. Zhao *et al.*, "Detecting profilable and overlapping communities with user-generated multimedia contents in LBSNs," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 10, no. 1, 2013, Art. no. 3.
[30] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: The state-of-the-art and comparative study," *ACM Comput. Surv.*, vol. 45, no. 4, 2013, Art. no. 43.
[31] A. Amelio and C. Pizzuti, "Overlapping community discovery methods: A survey," in *Social Networks: Analysis and Case Studies*. Vienna, Austria: Springer, 2014, pp. 105–125.
[32] J. J. Whang, D. F. Gleich, and I. S. Dhillon, "Overlapping community detection using neighborhood-inflated seed expansion," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 5, pp. 1272–1284, May 2016.
[33] M. Shahriari *et al.*, "Predictive analysis of temporal and overlapping community structures in social media," in *Proc. 5th Int. Conf. Companion World Wide Web*, 2016, pp. 855–860.
[34] J. B. Pereira, A. J. Enright, and C. A. Ouzounis, "Detection of functional modules from protein interaction networks," *Proteins, Struct. Funct. Bioinformat.*, vol. 54, no. 1, pp. 49–57, 2004.
[35] Y. Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 466, pp. 761–764, 2010.

[36] T. S. Evans and R. Lambiotte, "Line graphs, link partitions, and overlapping communities," *Phys. Rev. E*, vol. 80, no. 1, pp. 1–8, 2009.
[37] C. Pizzuti, "Overlapped community detection in complex networks," in *Proc. Genetic Evol. Comput. Conf.*, 2009, pp. 859–866.
[38] C. Shi *et al.*, "A link clustering based overlapping community detection algorithm," *Data Knowl. Eng.*, vol. 87, pp. 394–404, 2013.
[39] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary particle swarm optimization," *Swarm Evol. Comput.*, vol. 9, pp. 1–14, 2013.
[40] S. Lee *et al.*, "Modified binary particle swarm optimization," *Progress Natural Sci.*, vol. 18, no. 9, pp. 1161–1166, 2008.
[41] F. Afshinmanesh, A. Marandi, and A. Rahimi-Kian, "A novel binary particle swarm optimization method using artificial immune system," in *Proc. Int. Conf. Comput. Tool*, 2005, vol. 1, pp. 217–220.
[42] W. N. Chen *et al.*, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Trans. Evol. Comput.*, vol. 14, no. 2, pp. 278–300, Apr. 2010.
[43] L. Y. Chuang, S. W. Tsai, and C. H. Yang, "Improved binary particle swarm optimization using catfish effect for feature selection," *Expert Syst. Appl.*, vol. 38, no. 10, pp. 12699–12707, 2011.
[44] E. R. Hruschka, R. J. G. B. Campello, A. A. Freitas, and A. C. P. L. F. de Carvalho, "A survey of evolutionary algorithms for clustering," *IEEE Trans. Syst., Man, Cybern.*, vol. 39, no. 2, pp. 133–155, Mar. 2009.
[45] H. G. Ayad and M. S. Kamel, "On voting-based consensus of cluster ensembles," *Pattern Recog.*, vol. 43, pp. 1943–1953, 2010.
[46] V. Nicosia *et al.*, "Extending the definition of modularity to directed graphs with overlapping communities," *J. Statist. Mech.*, vol. 2009, 2009, Art. no. P03024.
[47] Z. H. Wu, Y. F. Lin, S. Gregory, H. Y. Wan, and S. F. Tian, "Balanced multi-label propagation for overlapping community detection in social networks," *J. Comput. Sci. Technol.*, vol. 27, pp. 468–479, 2012.
[48] Y. Zhang and D. Y. Yeung, "Overlapping community detection via bounded nonnegative matrix tri-factorization," in *Proc. Knowl. Discovery Data*, 2012, pp. 606–614.
[49] M. Zhou, "Infinite edge partition models for overlapping community detection and link prediction," *Comput. Sci.*, pp. 1135–1143, 2015.
[50] X. Wen *et al.*, "A maximal clique based multiobjective evolutionary algorithm for overlapping community detection," *IEEE Trans. Evol. Comput.*, to be published.
[51] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E*, vol. 78, no. 4, 2008, Art. no. 046110.
[52] G. Palla *et al.*, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, pp. 814–818, 2005.
[53] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317–325, 2003.

Faliang Huang received the Ph.D. degree in data mining from the South China University of Technology, Guangdong Sheng, China, in 2011.

He is now an Associate Professor in the Department of Software Engineering, Fujian Normal University, Fuzhou Shi, China. His research interests include data mining and natural computing.

Xuelong Li (M'02–SM'07–F'12) is a full professor with the Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, China.

Shichao Zhang (M'04–SM'04) received the Ph.D. degree in computer science from Deakin University, Burwood, VIC, Australia.

He is currently a China 1000-Plan Distinguished Professor with the Department of Computer Science, Guangxi Normal University, Guangxi, China. He has authored or coauthored about 60 international journal papers and more than 60 international conference papers. His research interests include information quality and pattern discovery.

Prof. Zhang served/is serving as an Associate Editor for the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, *Knowledge and Information Systems*, and the IEEE INTELLIGENT INFORMATICS BULLETIN.

Jilian Zhang received the Ph.D. degree in information systems from the Singapore Management University, Singapore, in 2014.

He has worked on query authentication for outsourced databases, database privacy, and spatial databases.

Jinhui Chen received the Ph.D degree in computer science from Kobe University, Kobe, Japan, in 2016.

He is currently an Assistant Professor with Kobe University. He is mainly engaged in pattern recognition and machine learning. He has authored or coauthored more than 10 publications in major international conferences and journals, such as ACM MM, ACM ICMR, ACCV, ICIP, ACII, and EURASIP JIVP.

Zhinian Zhai received the Ph.D. degree in data mining from the South China University of Technology, Guangdong Sheng, China, in 2012.

His research interests include information security and artificial intelligence.