Discovering Fuzzy Structural Patterns for Graph Analytics

Tiantian He, and Keith C. C. Chan

Abstract—**Many real-world data can be represented as attributed graphs that contain vertices each of which is associated with a set of attribute values. Discovering clusters, or communities, which are structural patterns in these graphs is one of the most important tasks in graph analysis. To perform the task, a number of algorithms have been proposed. Some of them detect clusters of particular topological properties whereas some others discover them based mainly on attribute information. Also, most algorithms discover disjoint clusters only. As a result, they may not be able to detect more meaningful clusters hidden in the attributed graph. To do so more effectively, we propose an algorithm, called FSPGA, to discover fuzzy structural patterns for graph analytics. FSPGA performs the task of clusters discovery as a fuzzy constrained optimization problem which takes into consideration both graph topology and attribute values. FSPGA has been tested with both synthetic and real-world graph data sets and is found to be efficient and effective at detecting clusters in attributed graphs. FSPGA is a promising fuzzy algorithm for structural pattern detection in attributed graphs.**

Index Terms—**fuzzy clustering, fuzzy structural pattern, fuzzy graph clustering, relational fuzzy c-means clustering, attributed graph, community detection, social network, biological network, complex network, graph analytics**

I. Introduction

An *attributed graph* contains *attributed vertices* connected by edges and each attributed vertex is associated with a set of attribute values. In these attributed graphs, there are a number of sub-graphs in which the vertices are more densely connected and are inter-related, according to their attribute values. Such sub-graphs are deemed as graph clusters, or communities, which are *structural patterns* in the graph. Many real-world problems can be formulated as the discovering of such clusters in the attributed graph. For example, in social network analysis, the identification of social groups is considered as social community detection. Similarly, the identification of functional modules in biological network graphs is also considered as cluster detection in biological graphs.

To solve the problem of discovering clusters in graphs, several so-called graph clustering algorithms have been proposed. And the problem of clustering in graphs has drawn much attention in recent years [1] [2]. Unsurprisingly, most graph clustering algorithms detect clusters based on pre-specified topologies or edge structures. For example, in [3], an algorithm that detects clusters based on *edge centrality* is presented. In [4], another measure, called *modularity*, which is defined as a function of the differences in density within graph clusters and a *null-graph* (in which vertices are connected randomly) is proposed. Based on it, two algorithms presented in [5] and [6] attempt to detect graph clusters through modularity optimization. In [7], the authors present a formalism in which it shows that some clusters smaller than a certain size cannot be detected by those algorithms based on modularity optimization.

Besides these algorithms, there are other algorithms that discover graph clusters taking advantages of other properties of network topologies. For example, in [8], an algorithm is proposed to detect graph clusters based on the *clique percolation method*. In [9], a graph clustering method called *affinity propagation* (AP) is proposed to detect clusters based on the similarities between candidate cluster centers and other vertices. In [10], a method is proposed to detect graph clusters by introducing the concept of a *link graph* to facilitate optimization of edge densities. In [11], spectral clustering for graph data is proposed to consider *normalized cuts* [12] that may reveal the similar edge structure of the vertices in the same cluster. In [13], *Mixed Membership Stochastic Block models* (MMSB) is proposed to detect graph clusters by optimizing the posterior probability that a pair of vertices are connected. In [14], a model based algorithm called CoDa is proposed to detect communities in graphs. Modeling the discovering of communities as identifying the community affiliations of each vertex, the best affiliation can be identified by optimizing the posterior probabilities that are used to represent the possibility that vertices belong to a community in a generative model.

Besides those algorithms based on graph topology, there are several algorithms proposed to discover graph clusters possessing similar attribute values. For example, some attempts have been made to make use of the *k*-means algorithm [15] to group vertices with higher similarity of attributes into the same clusters. In [16], an algorithm (MAC) that is based on a probabilistic generative model is proposed for clustering vertices that are labeled with Boolean attribute values. In [17], a graph summarization algorithm called *k*-SNAP is proposed to

This paragraph of the first footnote will contain the date on which you submitted your paper for review. It will also contain support information, including sponsor and financial support acknowledgment. For example, “This work was supported in part by the U.S. Department of Commerce under Grant BS123456”.

T. He and Keith C. C. Chan are with the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong SAR (e-mail: {csthe, cskcchan}@comp.polyu.edu.hk).

detect graph clusters by grouping vertices into the same cluster according to a similarity measure of the attribute values.

These graph clustering algorithms are not very well suited for the task to discover meaningful communities in attributed graphs because they take more emphasis either on graph topology or attributes associated with the vertices, but overlook the other.

To consider both attributes and structures, several algorithms are proposed. In [18], SA-Cluster is proposed to detect disjoint graph clusters using a *neighborhood random walk model*. The cluster membership of each vertex is obtained when the transition matrix reaches the steady state. In [19], the efficiency of SA-Cluster is improved by computing the transition matrix incrementally. In [20], EDCAR is proposed to mine clusters by grouping together vertices that are densely connected and share similar attribute values. Though these algorithms may detect communities using both edge structure and attributes, the communities discovered are not overlapping.

In addition to the above algorithms, some algorithms detect graph clusters by utilizing generative models. In [21], a general Bayesian model for graph clustering (GBAGC) is proposed to make use of a Bayesian generative model to estimate structural and attribute similarity of pairwise vertices in each cluster. A number of disjoint graph clusters are obtained after the all parameters are estimated. In [22], an algorithm, called CESNA, is proposed to make use of a statistical model to determine the posterior probability that pairwise vertices are connected given edge structures and attributes in a cluster. Cluster membership is determined when the posterior probability is maximized. In [23], an algorithm called Circles is proposed to detect communities in social graphs. Circles determines community membership by estimating the similarity between user attributes and those which are commonly observed in members of each cluster. The cluster membership of a vertex is determined to be those that are predicted to have higher similarities with other vertices in the same cluster. In [24], an evolutionary community detection algorithm, called ECDA, is proposed to detect for communities in social networks by considering network connections and attribute labeled to each pair of vertices.

Inspired by topic modeling [25], several topic-model-based approaches, such as Link-PLSA-LDA [26], Relational Topic Model [27], iTopicModel [28], PL-DC [29] and Block-LDA [30] can also be used to segment document network graphs. With these topic-model-based approaches, cluster membership is determined by maximizing the probability that vertices in the same cluster labeled with the same topics. However, due to rather high demand for computational resources, these Topic-Model-based approaches are not developed to handle large attributed graphs [22].

Recently, fuzzy pattern analysis, such as fuzzy clustering has been drawn much attention because the feature of "soft membership" that is possessed by the algorithms based on fuzzy techniques may lead one to detect more sub-structures in different types of data. Besides of the classical fuzzy c-means algorithm [31], there are several algorithms based on the fuzzy c-means model, such as relational fuzzy c-means [32], fuzzy c-regression models [33], possibilistic fuzzy c-means models [34], and interval-based fuzzy model [35], which have been proposed for data clustering. And there are several fuzzy clustering algorithms proposed to solve specific clustering problems, such as motion detection [36] and linguistic analysis in web documents [37]. Among those proposed algorithms, FCAN [38] is the one that utilizes fuzzy techniques to detect clusters in complex network data. FCAN may detect clusters by segmenting a data matrix in which each element represents the strength of the relationship between pairwise data points. The entries of the data matrix are obtained by adding the binary value and the degree of similarity representing the connection and attribute similarity between pairwise vertices, respectively. Though effective to some extent, FCAN may not truly identify the strengths of topology and attribute values that may determine the cluster arrangement within the clustering process.

Given the prevalent works in graph clustering and fuzzy clustering algorithms, we have the following findings that may motivate us to develop a more suitable algorithm. First, most of the graph clustering algorithms detect clusters based on topological properties only, or the attribute information is not fully utilized, just like the work presented in [38]. Second, most of the approaches cannot detect overlapping clusters, which might be more desirable in some graph data, e.g., some communities in social networks are overlapping. Last but the most, currently, there are no effective fuzzy algorithms for discovering clusters in attributed graphs. To overcome the mentioned challenges, we propose an algorithm for discovering Fuzzy Structural Patterns for Graph Analytics (FSPGA). FSPGA performs its tasks by formulating the identification of clusters in attributed graphs as a fuzzy constrained optimization problem that takes into the consideration edge structure and attributes. FSPGA may identify the optimal membership arrangement that is determined by both edge structure and attribute information between vertices and clusters. By adopting the fuzzy sets theory, FSPGA may detect overlapping clusters in the attributed graph.

For performance evaluation, FSPGA is tested with both synthetic and real datasets including social and biological network graphs. The experimental results are verified against known ground-truth data. It is found that FSPGA obtains a better performance in both efficiency and effectiveness, compared with state-of-the-art graph clustering algorithms and fuzzy clustering algorithms. Given the performance, FSPGA is a very promising fuzzy algorithm for discovering structural patterns in the form of clusters in attributed graph data.

In Section II below, how the problem of discovering clusters in the attributed graph is formulated as a constrained optimization problem is discussed and the details of FSPGA is presented. In Section III, we present the results of experiments performed to evaluate the performance of FSPGA. In Section IV, we discuss the unique features of FSPGA, the differences between FSPGA and other fuzzy or non-fuzzy clustering algorithms. We also compare the computational complexity and memory requirement of FSPGA with some popular clustering algorithms. Finally, in Section V, we present the conclusion that summarizes the contributions of the paper and proposals for the future work.

II. FSPGA IN DETAILS

A. Mathematical preliminaries

Given an attributed graph containing n_V vertices and n_E edges, in which each vertex is associated with a set of attribute values, the graph can be represented as G = (V, E, Λ), where the set of vertices, V, can be denoted as, V= $\{v_i \mid 1 \leq i \leq n_V\}$, the set of edges, E, can be denoted as E = $\{e_{ij} \mid 1 \leq i, j \leq n_V, i \neq j\}$, and the set of attributes that is associated with each vertex can be denoted as Λ where $\Lambda = \{att_i \mid 1 \leq i \leq n_\Lambda\}$.

Given the vertices and edges in G, we use an adjacency matrix **M** of dimensions, n_V by n_V, to represent the connections between vertices in G so that an entry, m_{ij}, in **M** has the value, 1, if v_i and v_j are connected and, 0, if they are not.

Besides the topological information, we also use another n_V-by-n_V matrix **A**, to represent the degree of attribute affinity between pairwise vertices in G. Hence, each entry in **A**, say a_{ij}, can be obtained by any measure that may evaluate how similar or related the vertices v_i and v_j are, given the attribute values associated with them. Here we assume that a_{ij} should be nonnegative and a higher magnitude of it means v_i and v_j are more related, given the attribute values associated with the two vertices. It also should be noted that the value of each a_{ij} in **A**, is determined by the attribute inter-relationship between two vertices, v_i and v_j only. In other words, though m_{ij} might be zero, which means that there is no connection between v_i and v_j, a_{ij} might be positive if the attribute values associated with v_i and v_j are considered similar or correlated based on some evaluation measures.

Given adjacency matrix **M** and matrix of attribute affinity **A**, we use the following augmented matrix to represent the mutual information between any pair of vertices in G

$$\mathbf{Y} = \begin{bmatrix} \alpha\mathbf{M} & \mathbf{0} \\ \mathbf{0} & (1-\alpha)\mathbf{A} \end{bmatrix} \tag{1}$$

where the parameter α is used to adjust the bias between edge structure and attribute similarity. The data matrix **Y** has the dimension of $2n_V$ by $2n_V$, the mutual information between pairwise vertices are located in the diagonal blocks of **Y**, while entries in other blocks are all zero-valued. Utilizing **Y**, FSPGA may perform the task of discovering clusters in G.

B. The function based algorithm

FSPGA performs the task of cluster detection using **Y**. To find optimal cluster membership for the vertices in G that takes into the consideration edge structure and attribute, FSPGA is considering to use an objective function to evaluate the overall quality of detected clusters.

To formulate the objective that is adopted by FSPGA, we firstly introduce an auxiliary matrix having the dimension of $2n_V$-by-k, **X**, where k is the number of the clusters to seek. FSPGA uses **X** to represent strength in terms of structure and attributes that a vertex belongs to a cluster. Specifically, the first n_V-by-k entries are used for representing the structural strength that a vertex belongs to a cluster, and the second n_V-by-k entries are used for representing the strength in terms of attributes that a vertex belongs to a cluster. Let x_{ij} be an element in **X**. The value of x_{ij} indicates either the structural strength or that in terms of attributes that vertex i belongs to cluster j, according to the subscripts of the element. Given the properties of **X**, it can be used to measure either the edge structure, or attribute strength that each vertex belongs to each cluster as **X** uses different blocks to consider these two aspects, respectively. The aggregation of the number of connections and the degrees of attribute affinity, weighted by the corresponding variables in **X** can be obtained if an appropriate method can be used. Then, we introduce the membership matrix **C**, which has the dimension of n_V by k. Each element of **C**, say c_{ij}, indicates the strength of membership that vertex i belongs to cluster j. Apparently, a higher value of c_{ij} means vertex i leans to cluster j more.

Given **Y**, auxiliary matrix **X**, and membership matrix **C**, we propose FSPGA to formulate the cluster detection in the attributed graph as the following objective function to be optimized

$$\begin{aligned} &\text{maximize} \\ &O = tr(\mathbf{S}^{\mathrm{T}}\mathbf{Y}\mathbf{X}) - \frac{1}{2}\left[\left|\mathbf{C}\right|_F^2 + \left|\mathbf{X}\right|_F^2 + \left|\mathbf{X}\mathbf{C}^{\mathrm{T}}\right|_F^2\right] \\ &\mathbf{S}^{\mathrm{T}} = \left[\mathbf{C}^{\mathrm{T}}, \mathbf{C}^{\mathrm{T}}\right] \\ &\textit{subject to}\ \mathbf{X} \geq 0,\ \ \mathbf{C} \geq 0, \mathbf{C}\mathbf{e}_1 = \mathbf{e}_2 \end{aligned} \tag{2}$$

where (i) $|\mathbf{C}|^2{}_F$, and $|\mathbf{X}|^2{}_F$ are the matrix Frobenius norms of **C** and **X**, which are used to smooth the variables in these matrices, (ii) $|\mathbf{X}\mathbf{C}^{\mathrm{T}}|^2{}_F$ is the matrix Frobenius norm of the product of **X** and the transpose of **C**. (iii), $\mathbf{e}_1$ and $\mathbf{e}_2$ are k-by-1 and n_V-by-1 vectors, in which all elements are 1's. With the use of the proposed objective function, FSPGA can have the advantage that it can discover graph clusters by taking into consideration both edge structure and attribute information between vertices in the graph.

To better explain how FSPGA determines cluster membership of each vertex in a graph, let us consider the first term of (2). It is used to aggregate the number of connections and the degrees of attribute affinity within each of the k clusters in a graph. This first term, $tr(\mathbf{S}^{\mathbf{T}}\mathbf{Y}\mathbf{X})$ can be rewritten as

$$tr(\mathbf{S}^{\mathrm{T}}\mathbf{Y}\mathbf{X}) = \sum_{i=1}^{k} \mathbf{s}_i^{\mathrm{T}}\mathbf{Y}\mathbf{x}_i \tag{3}$$

From (3), in the other words, the k elements in $tr(\mathbf{S}^{\mathbf{T}}\mathbf{Y}\mathbf{X})$ are summed up and it is this total sum that FSPGA uses to evaluate the overall quality of the k clusters in the optimization process. Based on (3), the roles played by the variables in **C**, and **X** in measuring the overall clustering quality become clear. By evenly dividing the variables in each column of **X** into two parts and substituting **Y** in (3) using (1), (3) can be rewritten as

$$\sum_{i=1}^{k} \mathbf{s}_i^{\mathrm{T}}\mathbf{Y}\mathbf{x}_i = \sum_{i=1}^{k} \left[\mathbf{c}_i^{\mathrm{T}}\ \mathbf{c}_i^{\mathrm{T}}\right] \begin{bmatrix} \alpha\mathbf{M} & \mathbf{0} \\ \mathbf{0} & (1-\alpha)\mathbf{A} \end{bmatrix} \begin{bmatrix} \mathbf{x}_i^a \\ \mathbf{x}_i^b \end{bmatrix} \tag{4}$$

where $\mathbf{x}^a{}_i$ and $\mathbf{x}^b{}_i$ represent the first and second n_V variables in $\mathbf{x}_i$, respectively. From (4), It should be noted that **M** is multiplied by the variables in $\mathbf{x}^a{}_i$, and **A** is multiplied by the variables in $\mathbf{x}^b{}_i$. The products of $\mathbf{M}\mathbf{x}^a{}_i$ and $\mathbf{A}\mathbf{x}^b{}_i$ represent the total number of connections and the total degrees of attribute affinity in cluster i, weighted by $\mathbf{x}^a_i$, and $\mathbf{x}^b_i$, respectively. In other words, the variables in each column of **X**, are used to obtain the weighted sums of connections and weighted degrees of attribute affinity within each cluster. Since **M** is different from **A**, in general $\mathbf{x}^a_i$, and $\mathbf{x}^b_i$ are always different within the optimization process, they contribute to the graph clustering process in different ways.

Unlike the elements in **X**, the elements in **C**, such as $\mathbf{c}_i$, is used to combine the effect of both the number of connections and the degrees of attribute affinity in cluster i, and such aggregation is made possible by FSPGA taking into consideration the variables in each column of **C** as the degrees of the cluster membership for each vertex with respect to each cluster.

To show how (3) is computed by FSPGA in detail, we give a simple example below. It consists of an augmented matrix representing a graph containing 4 vertices which are segmented into 2 clusters. This matrix is shown as

$$\mathbf{Y}=\begin{bmatrix}\begin{bmatrix}0&1&1&1\\1&0&1&0\\1&1&0&0\\1&0&0&0\end{bmatrix} & \mathbf{0}\\ \mathbf{0} & \begin{bmatrix}0&0.6324&0.9575&0.9572\\0.6324&0&0.9649&0.4854\\0.9575&0.9649&0&0.8003\\0.9572&0.4854&0.8003&0\end{bmatrix}\end{bmatrix} \tag{5}$$

where the first and second block at the main diagonal are the matrices **M** and **A**, respectively. For simplicity, we ignore the parameter α here. We assume that the variables in **C** and **X** obtained by FSPGA after some iteration are given as follows:

$$\mathbf{C}=\begin{bmatrix}0.3914&0.6086\\0.9625&0.0375\\0.4827&0.5173\\0.5067&0.4933\end{bmatrix},\ \mathbf{S}=\begin{bmatrix}\mathbf{C}\\\mathbf{C}\end{bmatrix},\ \mathbf{X}=\begin{bmatrix}0.6787&0.2769\\0.7577&0.0462\\0.7431&0.0971\\0.3922&0.8235\\0.6555&0.6948\\0.1712&0.3171\\0.7060&0.9502\\0.0318&0.0344\end{bmatrix} \tag{6}$$

To obtain the total number of connections and the total degrees of attribute affinity in a cluster, say cluster 1, we make use of (4)

$$\mathbf{s}_1^{\mathrm{T}}\mathbf{Y}\mathbf{x}_1=[\mathbf{c}_1^{\mathrm{T}}\ \mathbf{c}_1^{\mathrm{T}}]\begin{bmatrix}\mathbf{M}&\mathbf{0}\\\mathbf{0}&\mathbf{A}\end{bmatrix}\begin{bmatrix}\mathbf{x}_1^a\\\mathbf{x}_1^b\end{bmatrix}=5.5740 \tag{7}$$

where $\mathbf{s}_1$, $\mathbf{c}_1$, and $\mathbf{x}_1$ are the first columns in **S**, **C**, and **X** in (6). Based on (4), FSPGA can evaluate the overall quality of k clusters in the graph. If optimal values in **X** exist, the optimal membership matrix **C**, which assigns each vertex to the cluster that it has more connections with and that contains vertices that have attribute values most related or similar to it, can be found by FSPGA.

In the case that there is no connection between two vertices, it should be noted that they might still be grouped into the same cluster. This is because the attribute values that are associated with each of them can still be similar or correlated and in such case, the corresponding variables in **X** can be positive. However, one may notice that the value of $tr(\mathbf{S}^{\mathrm{T}}\mathbf{Y}\mathbf{X})$, i.e., the sum of the total number of edges in the k clusters and degrees of attribute affinity between the vertex pairs in each of them, may increase when the variables in **S** and **X** become larger. In such case, FSPGA makes use of $|\mathbf{X}\mathbf{C}^{\mathrm{T}}|^2_F$ to penalize the variables in **X** and **C** which are assigned with too large or too small values. In other words, only when the variables in **S** and **X** are assigned with appropriate values that the objective function O can be optimized. The cluster membership matrix **C** obtained in such case is thus determined by both edge structure and attribute information. Moreover, since Equation (2) satisfies the fuzzy clustering constraint that requires the sum of each row in **C** to be 1, it is very convenient for overlapping clusters to be discovered after the optimal cluster membership matrix **C** is obtained.

C. The iterative updating algorithm

The proposed objective function is a constrained quadratic function. Based on the KKT conditions for constrained optimization problems, we may find the corresponding rules to iteratively update the matrices **C**, and **X** to search the local optima.

1) Updating rule for ***C*** *and adoption of fuzzy clustering membership*

Let γ_{ij} and λ_i be the Lagrange multipliers for the constraints of $c_{ij} \geq 0$ and $\Sigma_j c_{ij} = 1$. The Lagrange function L for **C** is

$$L(\mathbf{C},\boldsymbol{\gamma})=O-tr(\boldsymbol{\gamma}^{\mathrm{T}}\mathbf{C})-\boldsymbol{\lambda}^{\mathrm{T}}(\mathbf{C}\mathbf{e}_1-\mathbf{e}_2) \tag{8}$$

where $\boldsymbol{\gamma} = [\gamma_{ij}]$ and $\boldsymbol{\lambda} = [\lambda_i]$ are Lagrange multipliers for the constraints of the non-negativity of **C** and the sum-to-1 of variables in each row of **C**. Based on the KKT conditions, we have

$$\begin{aligned}&\frac{\partial L}{\partial \mathbf{C}}=\alpha\mathbf{M}\mathbf{X}_1+(1-\alpha)\mathbf{A}\mathbf{X}_2-\mathbf{C}-\mathbf{C}\mathbf{X}^{\mathrm{T}}\mathbf{X}-\boldsymbol{\gamma}-\boldsymbol{\lambda}\mathbf{e}_1^{\mathrm{T}}=0\\&\boldsymbol{\gamma}\circ\mathbf{C}=\mathbf{0}\\&\boldsymbol{\gamma}\geq 0\\&\mathbf{C}\mathbf{e}_1=\mathbf{e}_2\end{aligned} \tag{9}$$

where (i) "$\circ$" means the Hadamard product of two matrices with the same dimension, (ii) $\mathbf{X}_1$ and $\mathbf{X}_2$ are two block matrices obtained by dividing **X** between row n_V and n_V+1. Based on (9), we have the following element wise equation system

$$\begin{aligned}&[\alpha\mathbf{M}\mathbf{X}_1+(1-\alpha)\mathbf{A}\mathbf{X}_2]_{ij}-(\mathbf{C}+\mathbf{C}\mathbf{X}^{\mathrm{T}}\mathbf{X})_{ij}-\gamma_{ij}-\lambda_i=0\\&\gamma_{ij}\circ c_{ij}=0\\&\gamma_{ij}\geq 0\\&\sum_j c_{ij}=1\end{aligned} \tag{10}$$

Given the first equation in (10), we have

$$[\alpha\mathbf{M}\mathbf{X}_1+(1-\alpha)\mathbf{A}\mathbf{X}_2]_{ij}-(\mathbf{C}+\mathbf{C}\mathbf{X}^{\mathrm{T}}\mathbf{X})_{ij}-\lambda_i=\gamma_{ij} \tag{11}$$

Using (11) to replace γ_{ij} in the equation of Hadamard product, we have the iterative updating rule for **C**

$$c_{ij} \leftarrow c_{ij} \frac{(\alpha \mathbf{MX}_1 + (1-\alpha)\mathbf{AX}_2)_{ij} - \lambda_i}{(\mathbf{CX}^{\mathrm{T}}\mathbf{X} + \mathbf{C})_{ij}} \tag{12}$$

In the above equation, one more unknown, λ_i, needs to be determined for the updating of the variables in **C**. Given the constraint that the sum of each row of variables is one (see Equation (10)), we have

$$\sum_j c_{ij} \frac{(\alpha \mathbf{MX}_1 + (1-\alpha)\mathbf{AX}_2)_{ij} - \lambda_i}{(\mathbf{CX}^{\mathrm{T}}\mathbf{X} + \mathbf{C})_{ij}} = 1 \tag{13}$$

Given Equation (13), λ_i can, therefore, be solved as

$$\lambda_i = \frac{\left(\sum_j c_{ij} \frac{(\alpha \mathbf{MX}_1 + (1-\alpha)\mathbf{AX}_2)_{ij}}{(\mathbf{CX}^{\mathrm{T}}\mathbf{X} + \mathbf{C})_{ij}} \right) - 1}{\sum_j \frac{c_{ij}}{(\mathbf{CX}^{\mathrm{T}}\mathbf{X} + \mathbf{C})_{ij}}} \tag{14}$$

Using the value of λ_i to replace the corresponding variable in (12), the iterative updating rule, which is under the fuzzy clustering framework for **C**, can be obtained. With such an updating rule, the sum of each row in **C** is constrained to be 1. within the optimization procedure. As a result, a vertex in G may belong to more than one cluster due to the considerations of fuzzy cluster boundaries.

2) Updating rule for ***X***

Let η_{ij} be the Lagrange multipliers for the constraints $x_{ij} \geq 0$, hence the Lagrange function L for **X** is

$$L(\mathbf{X}, \boldsymbol{\eta}) = O - tr(\boldsymbol{\eta}^{\mathrm{T}}\mathbf{X}) \tag{15}$$

where $\boldsymbol{\eta} = [\eta_{ij}]$ is the matrix of Lagrange multipliers for the non-negativity of **X**. Based on the KKT conditions, we have

$$\begin{gathered} \frac{\partial L}{\partial \mathbf{X}} = \mathbf{YS} - \mathbf{XC}^{\mathrm{T}}\mathbf{C} - \mathbf{X} - \boldsymbol{\eta} = \mathbf{0} \\ \boldsymbol{\eta} \circ \mathbf{X} = \mathbf{0} \\ \boldsymbol{\eta} \geq \mathbf{0} \end{gathered} \tag{16}$$

Given (16) we have the following element wise equation system

$$\begin{gathered} [\mathbf{YS} - \mathbf{XC}^{\mathrm{T}}\mathbf{C} - \mathbf{X}]_{ij} = \eta_{ij} \\ \eta_{ij} \circ x_{ij} = 0 \\ \eta_{ij} \geq 0 \end{gathered} \tag{17}$$

Given the equation system (17), the element wise updating rule for **X** can be derived

$$x_{ij} \leftarrow x_{ij} \frac{(\mathbf{YS})_{ij}}{(\mathbf{XC}^{\mathrm{T}}\mathbf{C} + \mathbf{X})_{ij}} \tag{18}$$

By iteratively updating the variables in **λ**, **C** and **X** using the rules shown in (14), (12), and (18), FSPGA may find the local optima for (2) in a finite number of iterations.

D. Summary of the algorithm

Given the description from A to C in Section II, FSPGA can be summarized as the pseudo codes shown in Fig. 1. Once the number of clusters k, the adjust parameter α, maximum number of iteration and the minimum tolerance, τ are determined, FSPGA will automatically search for the optimal matrix of membership, **C** in a finite number of iterations. After FSPGA is stopped according to the terminal condition, the obtained **C** can be seen as the approximately optimal cluster arrangement.

E. Determining the cluster affliation

Having obtained the fuzzy membership in **C** for each vertex to belong to each cluster, FSPGA can determine, for each cluster, all its members. As vertices may belong to more than one cluster, FSPGA may determine whether v_i belongs to cluster j according to the following inequality

$$c_{ij} \geq \frac{\beta\sqrt{k-1}+1}{k} \tag{19}$$

where k is the number of clusters and β is a positive real number that is used to determine the extent of overlapping between identified clusters in the attributed graph. Here, β is a global parameter which is used to determine if each vertex, say v_i, belongs to cluster j after the optimization process. In addition, it should be noted that β is used only for the case of vertices whose degrees of cluster membership are not the highest for that vertex and FSPGA can discover disjoint clusters in an attributed graph when β is set to zero. Given this setting, it should be noted that it becomes more possible for more vertices to be assigned only to those clusters with the highest cluster membership and the extent of overlapping between detected clusters becomes smaller when β is set to a relatively high value. Hence, β can be adjusted according to the demand of overlapping in different attributed graph data and the variations of β won't change the number of clusters.

III. Experiments and analysis

In this section, we describe the details of the data sets that we used. We also explain how experiments and what criteria we used to evaluate the performance of FSPGA.

A. Experimental set up and evaluation metrics

1) Baselines for comparison

To show the desirable features of FSPGA, we selected a number of graph clustering algorithms to compare with FSPGA. These algorithms include Affinity Propagation clustering (AP), Spectral clustering (SC), CoDa, Fuzzy *c*-means clustering (FCM), improved Relational Fuzzy *c*-means clustering

```
Algorithm FSPGA
Input:      Y, $\alpha$, max_iteration, $\tau$, k
Output:     C, X
randomly initialize C, X;
normalize C using C= C./($\mathbf{C}\mathbf{e}_1\mathbf{e}_1^{\mathrm{T}}$)
for count=1: max_iteration
    fixing X
    update $\boldsymbol{\lambda}$ and C using (14) and (12);
    fixing C
    update X using (18);
     if ($|\mathbf{C}^{i} - \mathbf{C}^{i-1}|_F < \tau$)
        compute objective value using (2);
        break;
     end if
end for
return C, X;
```

Fig. 1. Pseudo codes of FSPGA

(iRFCM), CESNA, Relational topic model (RTM) and ECDA. Selecting these algorithms as baselines is because they are either the latest algorithms or classical ones and have all been used effectively to detect clusters in various network graphs. Specifically, AP, SC, and CoDa may detect graph clusters that take different topological properties of network graph data. For our experiments, we used the SC that makes use of the *normalized cut* in graph clustering. FCM may detect graph clusters making use of information of similarity between pairwise vertices in G. Therefore, we used the information in Λ as the input that is used to compute the similarity between pairwise vertices for FCM. As iRFCM is a version of FCM that can be used to discover graph clusters, we tested it using the same data as FSPGA uses. Algorithms like CESNA, RTM, and ECDA are ones taking into consideration both graph topologies and attribute values. RTM has been shown to be a very effective topic-model based approach to segment relational data. CESNA performs graph clustering using a generative process that determines cluster membership of a vertex by computing an estimate of the joint probability based on structure and vertex attributes. ECDA performs its tasks using an evolutionary graph clustering algorithm.

For performance benchmarking, we used the source code or executables made available by the authors. All the experiments were conducted under the same environment which included a workstation with 4-core 3.4GHz CPU and 16GB RAM.

2) Experimental set-up

To ensure that the algorithms we used in the experiment may obtain a robust performance, we tested them using the parameters in such a way that either the default settings as recommended by the authors are used or that they are tuned by trials to find the best settings.

Specifically, the AP, Coda, and ECDA algorithms do not require input parameters to be set by the users. For these algorithms, the default settings as recommended and implemented by the authors were used. For algorithms, including SC, FCM, iRFCM, and RTM, which require parameters to manually input into the system, we tried as many different settings as we can, to obtain the best results for performance benchmarking. For example, SC requires that the parameter of *sigma* to be set by the users before it can run. To find a better set of parameters, we tried SC using different sigma from 1 to 10. The settings that give the best performance of SC are recorded and presented in our performance analysis report below. As for the number of clusters, *k*, we set it for those algorithms that need *k* as a predefined parameter, including, SC, FCM, iRFCM, CESNA, and RTM, to be equal to the number of ground truth clusters that are used for benchmarking.

For FSPGA, we set β to 0 when FSPGA discovers structural patterns in those datasets whose ground-truth clusters are disjoint. We set β to 3 for all those datasets whose ground-truth clusters overlap with each other. As for the other parameters, we set α to 0.5, the maximum number of iterations to 300. As for *k*, it is set to be the same as the other algorithms, which is equal to the number of ground-truth clusters in each of the datasets. All the algorithms, including FSPGA, were executed 10 times to obtain statistical averages for the performance measures.

3) Data description

For performance evaluations, we used both synthetic and real datasets with known ground truth. We used synthetic data to test the effectiveness and efficiency of different algorithms and we used the real-world data sets to test the robustness of the different algorithms regarding to different applications. The real data sets that we used are mainly categorized into two classes, including social network graph data and biological network graph data.

The data sets *Twitter*, *Ego-facebook*, and *Googleplus* [23] are obtained from real social networking sites. The vertices, edges, and attributes in these data sets represent users of the social networks, the friendship between users and user profiles, respectively. The *Twitter* dataset is constructed based on a number of social circles extracted from twitter.com. For this dataset, we have 2511 vertices, 37154 edges, and 9067 attribute values. The *Ego-facebook* data set is constructed based on a number of sub-networks extracted from facebook.com. In this data set, there are 4039 vertices, 88234 edges and 1283 attribute values. *Googleplus* is another set of online social network data which was constructed based on the sub-networks from plus.google.com. There are 7856 vertices, 321268 edges, and 2024 attribute values in the dataset. The ground truth social communities for this data set have been identified. There are 132, 191, and 91 ground truth clusters which are used for benchmarking the identified clusters from datasets *Twitter*, *Ego-facebook*, and *Googleplus*, respectively.

Krogan [40], *DIP* [41], and *BioGrid* [39] are three sets of biological data that are constructed based on known interactions between proteins related to *Saccharomyces cerevisiae*. In these three data sets, the vertices, edges, and attribute values represent the proteins, protein-protein interactions and GO terms [42], respectively. In *Krogan*, there are in total 2674 vertices, 7075 edges, and 3064 attribute values. In *DIP*, there are 4579 vertices, 20845 edges and 4237 attributes. In *BioGrid*, there are 5640 vertices, 59748 edges, and 4286 attribute values. These three data sets have the ground-truth data stored in the CYC2008 database [43] and there are 200 ground-truth clusters. Compared with those social network graph data used, *Krogan*, *DIP*, and *BioGrid*, are sparser. Using these two types of data allows us to find out how robust the algorithms are when used with different types of graphs.

Syn1k is a set of synthetic data which is generated based on the rule that the probability of intra-cluster edges is higher than that of inter-cluster edges and that vertices in the same cluster are more related to each other than those that are not. For this dataset, we used 1000 vertices that are divided into 4 disjoint ground truth communities, 9900 edges and 50 attribute values that are made to associate with each vertex. It should be noted that, the ground truth clusters of all the real data sets overlap with each other to some extent. Specifically, the overlapping rates between pairwise ground truth clusters in datasets *Twitter*, *Ego-facebook*, and *Googleplus* are 0.00193, 0.00113, and

0.01913, respectively. And that in *Krogan*, *DIP*, and *BioGrid*, it is 0.0004.

The above data sets are used to test the effectiveness of FSPGA and other algorithms. In addition, to test the scalability of FSPGA, we have generated several additional synthetic datasets ranging in size from 5,000 to 100,000 for our experiments.

4) Determining the degree of attribute affinity between pairwise vertices

To determine the degrees of attribute affinity between all pairs of vertices that are used by FSPGA, we use the following method. First, we use a statistical measure to determine whether a pair of attribute values, say att_i and att_j are significantly associated. This measure is defined as

$$diff(att_i, att_j) = \frac{o(att_i, att_j) - e(att_i, att_j)}{\sqrt{e(att_i, att_j)(1 - \frac{o(att_i +)}{n_E})((1 - \frac{o(att_j +)}{n_E}))}} \tag{20}$$

where $o(att_i, att_j)$ represents the number of edges that connect two vertices which are associated with att_i and att_j, $o(att_i+)$ represents the number of edges that connect vertices which are associated with att_i, $e(att_i, att_j)$ to represent the expected number of such edges in the case that the attributes of the connected vertices are independent and unassociated, and in such case $e(att_i, att_j)$ can be computed as $[o(att_i+)o(att_j+)]/n_E$. In [44] and [45], this measure is shown to approximately follow the *Standard Normal* distribution. One may, therefore, decide that att_i and att_j are significantly associated with each other at a 95% confidence level if $diff(att_i, att_j)$ is greater than 1.96. Otherwise, they can be considered not significantly associated with each other. With this measure, attribute values that are not relevant can be filtered out.

After the significantly associated attribute values are obtained, we may determine the degree of attribute affinity given all significantly associated attribute values of pairwise vertices (a_{ij}) using an information theoretical measure [15]

$$a_{ij} = \frac{r(v_i, v_j)}{H(v_i, v_j)}$$

$$r(v_i, v_j) = \sum_k \sum_m \Pr(att_{ik}, att_{jm}) \cdot \log \frac{\Pr(att_{ik}, att_{jm})}{\Pr(att_{ik}) \cdot \Pr(att_{jm})} \tag{21}$$

$$H(v_i, v_j) = -\sum_k \sum_m \Pr(att_{ik}, att_{jm}) \cdot \log \Pr(att_{ik}, att_{jm})$$

where $\Pr(att_{ik}, att_{jm})$ denotes the probability that two connected vertices are characterized by att_k and att_m, this probability can be computed as $o(att_k, att_m)/n_E$, $\Pr(att_{ik})$ denotes the probability that an edge may connect two vertices that are characterized by att_k, and these two probabilities can be computed as $o(+att_k)/n_E$. The magnitude of a_{ij}, can be interpreted as the information redundancy of the attribute values that are associated with v_i and v_j in the attributed graph. After normalization, it ranges from 0 to 1. A greater value of it means that the attribute values of the pair of vertices, v_i and v_j are more strongly associated with each other. Having obtained the degrees of attribute affinity, we use them to construct **A** that is used by FSPGA.

5) Evaluation metrics

For performance evaluation, we are considering different evaluation measures which are widely used for evaluating graph clustering algorithms and fuzzy clustering algorithms. For measures used for validating graph clusters, we used the Normalized Mutual Information (*NMI*), and the Average Accuracy (*Acc*) [46]. There are a number of measures for fuzzy clustering validity, such as Beni Index [47], Earth Mover's Distance [48], and several fuzzy Rand-Index-based measures [49]. In our experiments, we selected Fuzzy Adjusted Rand Index (*FARI*) [49] for evaluating the graph clusters discovered by different algorithms.

The *NMI* measures the overall accuracy of the matches between detected clusters and those that are considered "ground truth". It is defined as

$$NMI = \frac{\sum_{C,C^*} \Pr(C_i, C_j^*) \log \frac{\Pr(C_i, C_j^*)}{\Pr(C_i)\Pr(C_j^*)}}{\max(-\sum_i \Pr(C_i)\log\Pr(C_i), -\sum_j \Pr(C_j^*)\log\Pr(C_j^*))} \tag{22}$$

where $\Pr(C_i, C_j^*)$ denotes the probability that vertices are in both the detected cluster i and the true cluster j, and $\Pr(C_i)$ denotes the probability that a vertex is found to exist in cluster i. Based on this definition, if the *NMI* measure is high, it means that the clusters detected match well with the ground-truth clusters.

Contrary to the *NMI*, the *Acc* measure evaluates individually detected cluster. It is defined as

$$Acc = \sum_c \frac{|C_i|}{|C|} f\left(C_i, C^*\right) \tag{23}$$

where $|C|$ means the size of the detected clusters, and $f(.)$ stands for a mapping function between cluster i and the ground truth. For our purpose, we define $f(.)$ to be the maximum overlap between detected cluster i and a ground-truth cluster. Thus, *Acc* evaluates the best matching of each cluster. A higher value of *Acc*, therefore means that each detected cluster has a better match with the ground truth. The higher the *Acc* of all clusters detected by an algorithm, therefore means that the algorithm is more effective.

The Fuzzy Adjusted Rand Index (*FARI*) measures the overall adjusted agreement between the discovered and ground truth clusters and it is defined as

$$a = \frac{1}{2}\sum_i \sum_j \phi n_{ij}\left(\phi n_{ij} - 1\right), b = \frac{1}{2}\left(\sum_j\left(\sum_i \phi n_{ij}\right)^2 - \sum_i \sum_j \phi^2 n_{ij}^2\right)$$

$$c = \frac{1}{2}\left(\sum_i\left(\sum_j \phi n_{ij}\right)^2 - \sum_i \sum_j \phi^2 n_{ij}^2\right)$$

$$d = \frac{1}{2}\left(\left(\sum_i \sum_j \phi n_{ij}\right)^2 - \sum_i \sum_j \phi^2 n_{ij}^2 - 2(b+c)\right), \phi = \frac{N_g}{\sum_i \sum_j n_{ij}}$$

$$FARI = 2(ad - bc)/\left[b^2 + c^2 + 2ad + (a+d)(b+c)\right] \tag{24}$$

TABLE I
NMI, *ACC* AND *FARI* IN SYN1K

	Syn1k		
Approach	*NMI*	*Acc*	*FARI*
AP	0.152	0.747	0.01
CoDa	0.116	0.43	0.097
SC	0.232	0.528	0.277
FCM	0.732	0.871	0.674
iRFCM	0.718	0.739	0.677
CESNA	0.792	0.845	0.813
RTM	0.797	0.797	0.683
ECDA	0.272	0.466	0.203
FSPGA	**0.992**	**0.998**	**0.995**

where n_{ij} represents the number of vertices in both discovered cluster i and ground truth cluster j, N_g is the number of vertices in the ground truth database. *NMI*, *Acc*, and *FARI* evaluate the quality of detected clusters against the ground truth from different aspects. With these three measures, we can better evaluate the robustness of different algorithms.

B. Experimental results using synthetic data

1) Evaluation on clustering quality

For performance evaluation, we used a set of synthetic graph data containing 1000 vertices to test the effectiveness of all different algorithms. There are four disjoint ground truth clusters in the synthetic dataset. As mentioned above, the synthetic data are generated by assuming that the probability of vertices within the same cluster to be connected with other vertices to be higher than that of the probability between clusters. For our experiment, the data set *Syn1k* was generated by setting the probability of intra-cluster connections to be 0.05 and the probability of inter-cluster connections to be 0.01.

The performance of FSPGA and other algorithms on the synthetic dataset *Syn1k* with respect to *NMI*, *Acc*, and *FARI* is given in Table I. As the table shows, FSPGA performs better than other algorithms. No matter which of *NMI*, *Acc*, or *FARI* is considered, FSPGA may outperform all the compared baselines in dataset *Syn1k*. These experimental results show that FSPGA can be very effective with the discovering of clusters in the synthetic attributed graph.

2) Scalability test

To find out how FSPGA can scale up when dataset size increases, a series of synthetic data of sizes ranging from 5000 to 100,000 were generated using the same probabilities of 0.05 and 0.01 for intra- and inter-cluster vertex connections as is with *Syn1k*. Given these generated data, the scalability of FSPGA was studied in a number of experiments involving different data sets. The results obtained were compared with those obtained with CESNA, RTM, iRFCM, and AP. As FSPGA and these algorithms are all iterative in nature, a comparison is made based on the average execution time of each iteration. The results are shown in Fig. 2.

The results show that FSPGA scales up well when compared with RTM and iRFCM and AP. Even with the data sets containing as many as 100,000 vertices, FSPGA could complete each iteration in the optimization process in around 1 second and this is slightly faster than CESNA. However, when comparing the number of iterations that are required for the two algorithms to complete the cluster discovery tasks, it should be noted that CESNA needed at least 300 iterations whereas FSPGA converges much below 300. Given this to be the case, FSPGA is more computationally efficient.

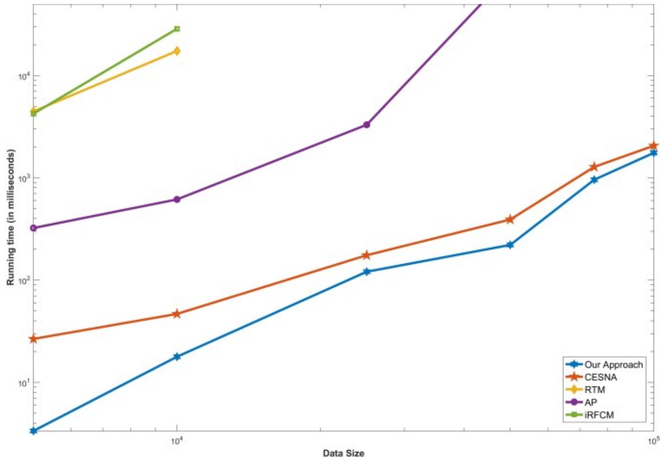


Fig. 2. Scalability test between different algorithms

When compared with AP, RTM and iRFCM, the computational time used by them is much more than FSPGA did. It should be noted that we did not obtain the results of scalability test of RTM or iRFCM when the size of synthetic data is larger than 10,000 as they were crushed under that situation. And the computational time of AP is also intolerable when the data size is larger than 25,000.

3) Sensitivity test of α

As described in Section 2, for FSPGA to performs its tasks, it requires the setting of a parameter α. The parameter is used to adjust the bias between the edge density and the degree of attribute affinity during the process of cluster identification. How the parameter may affect the performance of FSPGA can be investigated in several sensitivity tests using the data set *Syn1k*.

In our experiment, α was set to different values from 0 to 1, with an increment of 0.2, and FSPGA was used under these different settings to detect clusters. The performance was measured with *NMI*, *Acc*, and *FARI* and the results are shown in Fig. 3.

It is seen that when α was set to 0, which means that only the attribute values are considered, and when it is set to 1, which

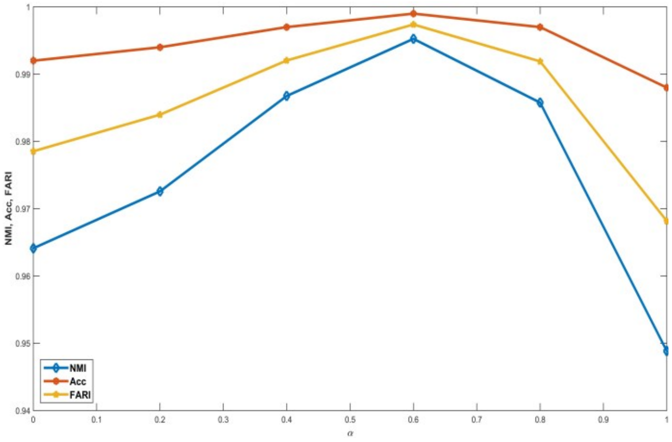


Fig. 3. Sensitivity test of α

TABLE II
NMI, *ACC* AND *FARI* OBTAINED FROM SOCIAL NETWORK DATA

	Twitter			***Ego-facebook***			***Googleplus***		
Approach	*NMI*	*Acc*	*FARI*	*NMI*	*Acc*	*FARI*	*NMI*	*Acc*	*FARI*
AP	0.598^{2nd}	0.479^{3rd}	0.123	0.528^{2nd}	0.416	0.194^{1st}	0.355	0.273	0.095
CoDa	0.584^{3rd}	0.471	0.182^{3rd}	0.524^{3rd}	0.502^{3rd}	0.13^{3rd}	0.373	0.375^{3rd}	0.079
SC	0.493	0.305	0.094	0.52	0.447	0.126	0.33	0.296	0.081
FCM	0.08	0.09	0.016	0.28	0.208	0.056	0.128	0.181	0.031
iRFCM	0.535	0.37	0.172	0.315	0.282	0.074	0.266	0.318	0.054
CESNA	0.572	0.528^{1st}	0.169	0.483	0.623^{1st}	0.118	0.42^{2nd}	0.47^{2nd}	0.105^{3rd}
RTM	0.028	0.099	0.014	0.227	0.167	0.061	0.023	0.151	0.019
ECDA	0.529	0.385	0.184^{2nd}	0.322	0.234	0.099	0.395^{3rd}	0.341	0.122^{2nd}
FSPGA	$\mathbf{0.641^{\text{1st}}}$	$\mathbf{0.513^{\text{2nd}}}$	$\mathbf{0.241^{\text{1st}}}$	$\mathbf{0.579^{\text{1st}}}$	$\mathbf{0.588^{\text{2nd}}}$	$\mathbf{0.17^{\text{2nd}}}$	$\mathbf{0.489^{\text{1st}}}$	$\mathbf{0.519^{\text{1st}}}$	$\mathbf{0.149^{\text{1st}}}$

means that only the edge structure is considered, the performance of FSPGA is affected negatively. When setting α to the value between 0.4 and 0.6, FSPGA obtains very good results. Given these results, we set α to be 0.5 in all our experiments so that both attribute values and edge structures are considered equally important by FSPGA.

C. Experimental results in real data

1) Application in social community detection

Social communities are important structural patterns in social graphs. The identification of such communities is important to social network analysis. For performance evaluation of FSPGA, we used three sets of social network data, including *Twitter*, *Ego-facebook,* and *Googleplus*. All these data sets have known ground-truth communities that have been verified in previous work. Given the fact that the number of ground truth clusters is known, for those algorithms which need to set the number of clusters (k), we set it to be the number of known ground truth clusters in each dataset.

The experimental results of *NMI*, *Acc*, and *FARI* obtained with these datasets are summarized in Table II. As the table shows, FSPGA performs more robustly than other algorithms. When the identified clusters are evaluated by *NMI*, FSPGA outperforms all the other algorithms in all the three social network datasets. When evaluated by *Acc*, FSPGA ranks the best in *Googleplus*, and second best in *Twitter*, and *Ego-facebook*, respectively. When the identified clusters are evaluated by *FARI*, FSPGA outperforms the other algorithms in the case of *Twitter*, and *Googleplus*, and ranks second best in *Ego-facebook*. In total, the above results obtained from social network data show that the social communities detected by FSPGA better match with the ground-truth when compared with the others.

2) Functional modules detection in biological graph data

Functional modules in biological networks, such as protein complexes in protein-protein interaction (PPI) network graphs also can be considered as structural patterns in the form of graph clusters.

To further test the effectiveness of FSPGA, we used three sets of PPI network data in our experiments. They included the data sets *Krogan*, *DIP*, and *BioGrid*. These datasets were chosen as the ground-truth, which correspond to known protein complexes, could be found and some of the known protein complexes are overlapping. Performance data based on *NMI*, *Acc* and *FARI* were obtained from the experiments. The results obtained with these two data sets are shown in Table III.

As shown in the table, FSPGA obtains better performance than all the other algorithms regardless of performance measures used. When the evaluation measure, *Acc* is considered, FSPGA outperforms all the baselines in all three datasets. When *NMI* is considered, FSPGA ranks the best in the case of *DIP* and *BioGrid*, and third with *Krogan*. When the discovered clusters are evaluated by *FARI*, FSPGA outperforms all other algorithms with *DIP* and *BioGrid*, and ranks third with *Krogan*.

As the objective function used by FSPGA considers the pairwise relationship between any pair of vertices in terms of edge structure and attribute information, the relative weighting between how much each of these two factors should be considered can be adjusted dynamically during the optimization process. The fuzzy cluster membership matrix **C** obtained by FSPGA can find k clusters in which vertices share

TABLE III
NMI, *ACC* AND *FARI* OBTAINED FROM BIOLOGICAL NETWORK DATA

	Krogan			***DIP***			***BioGrid***		
Approach	*NMI*	*Acc*	*FARI*	*NMI*	*Acc*	*FARI*	*NMI*	*Acc*	*FARI*
AP	0.692^{1st}	0.187^{3rd}	0.11	0.688^{2nd}	0.117^{2nd}	0.098	0.109	0.016	0.003
CoDa	0.688^{2nd}	0.199^{2nd}	0.298^{1st}	0.463	0.068^{3rd}	0.045	0.299	0.035	0.017
SC	0.609	0.079	0.026	0.588	0.047	0.009	0.545^{3rd}	0.032	0.087^{3rd}
FCM	0.454	0.078	0.115	0.49	0.06	0.138^{3rd}	0.444	0.048^{2nd}	0.073
iRFCM	0.342	0.055	0.058	0.444	0.049	0.091	0.355	0.046^{3rd}	0.045
CESNA	0.484	0.055	0.027	0.425	0.026	0.063	0.449	0.026	0.049
RTM	0.578	0.037	0.169	0.614^{3rd}	0.025	0.184^{2nd}	0.622^{2nd}	0.021	0.194^{2nd}
ECDA	0.631	0.142	0.229^{2nd}	0.299	0.058	0.016	0.145	0.026	0.043
FSPGA	$\mathbf{0.677^{\text{3rd}}}$	$\mathbf{0.202^{\text{1st}}}$	$\mathbf{0.191^{\text{3rd}}}$	$\mathbf{0.712^{\text{1st}}}$	$\mathbf{0.129^{\text{1st}}}$	$\mathbf{0.267^{\text{1st}}}$	$\mathbf{0.755^{\text{1st}}}$	$\mathbf{0.125^{\text{1st}}}$	$\mathbf{0.372^{\text{1st}}}$

similar weighted structure with each other. This represents, in other words, the optimized weighted aggregation of intra-cluster connections and attribute relativity. This feature allows FSPGA to group unconnected but related vertices to be taken into consideration based on attribute relativity. Moreover, as FSPGA allows fuzzy cluster membership to be considered within the optimization process, thereby making it possible for FSPGA to find overlapping clusters as fuzzy structural patterns in an attributed graph. These features are the reasons why FSPGA can obtain a more robust performance with both overlapping and non-overlapping clustering.

D. Case study-overlapping rate versus cluster quality

To find out what impact the parameter β can have on the quality of clusters, FSPGA is tested with all real datasets, using β from 0.1 to 4, with a 0.1 increment. The clusters obtained under different settings of β are evaluated using *NMI*, *Acc*, and *FARI*. The results are shown in Fig. 4 (a) - (c). Together with the overlapping rate of ground truth clusters of each real dataset, the impact of β on the quality of overlapping clustering were studied in detail.

The variations of *NMI*, *Acc*, and *FARI* are shown in Fig. 4 (a)-(c). As it is shown in the figures, the magnitudes of different clustering validity measures share similar variations. As the value of β becomes larger than a particular value, the clustering quality does not improve by much. In Fig. 4 (d), it should be noted that the extent of overlapping in all the datasets decreases and approximates to zero as β becomes larger and larger. Given these results, it should be noted that β needs to be adjusted for FSPGA to discover clusters with different extent of overlapping.

It is mentioned in Section II.E that β is used to constrain the number of vertices that can belong to more than one cluster. As a result, each vertex in an attributed graph is probably assigned to the cluster with the greatest degree of cluster membership when β is set to be high enough. As a result, the clustering quality of FSPGA would be approximately the same as with crisp clustering. Given the fact that the overlapping rate of most datasets is relatively low, e.g., 0.00193 in *Twitter*, 0.00113 in *Ego-facebook*, and 0.0004 in biological datasets, the clustering quality is better when β is set higher, e.g., 2.5 to 3.5 in *Twitter*, *Ego-facebook*, and biological datasets. Using relatively large β, discovered clusters are mostly disjoint and the overall overlapping rate is therefore similar to that of the ground truth clusters (see Fig. 4 (d)).

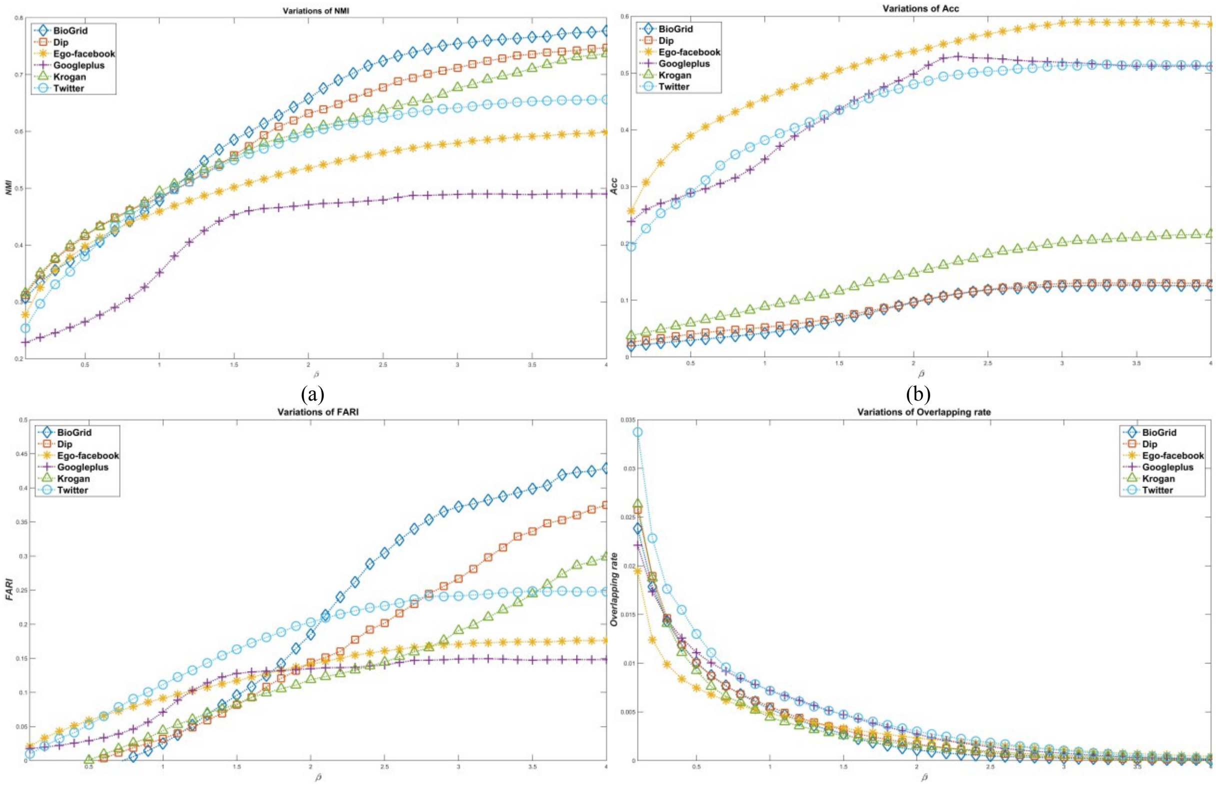
However, when β is set between 2.5 and 3.5, it may degrade the quality of clusters discovered in *Googleplus*. This is because this data set has a relatively high overlapping rate. E.g., the *Acc* decreases when β is set larger than 2.5.

Given these characteristics of β, it is necessary for one to adjust the setting. However, FSPGA performs robustly when β is set between 2 and 3.5 and this is why β is set to 3 in our experiments.

IV. Discussion

A. Comparisons between FSPGA and formal fuzzy clustering algorithms

With the above features, FSPGA can be considered different from such popular fuzzy clustering algorithms as the fuzzy *c*-means algorithm (FCM), and the relational fuzzy c-means algorithm (iRFCM). With the objective function that it uses,



(a)

(b)

(c)

(d)

Fig. 4. *NMI*, *Acc*, *FARI*, and overlapping rate in social and biological datasets

FSPGA can determine, dynamically, the degrees of cluster membership based on both the edge structure and attribute information of vertices. Compared with algorithms such as iRFCM which adjust global bias between **M** and **A**, FSPGA may identify meaningful clusters which other algorithms may miss.

Also, while existing fuzzy clustering algorithms minimize dissimilarity between data entities and cluster centers, FSPGA takes into consideration the edge structure and attribute relativity within each cluster. By so doing, FSPGA can identify clusters in which the weighted aggregation of intra-cluster connections and degrees of attribute relativity between any pair of vertices is optimized. These features are some of the reasons why FSPGA may perform better than other algorithms we used for comparison.

B. Computational complexity and space requirements of FSPGA

As the computational complexity of FSPGA is dependent mainly on the iterative updating of **C** and **X**, the complexity of the process of determining these matrices is considered here.

Let n_V and k be the number of vertices in the graph and the number of clusters in an attributed graph. It should be noted that $k << n_V$ in practice. According to Equation (8), for updating each element in **C**, say c_{ij}, it approximates the order of $O((2k+2)n_V)$. Hence updating all elements in **C** approximates the order of $O(k(2k+2)n_V^2)$. According to Equation (10), updating each element in $\boldsymbol{\lambda}$ follows the order of $O(k)$. This is because the computational components in the numerator and denominator are the same as those in (8). As a result, updating $\boldsymbol{\lambda}$ follows the order of $O(kn_V)$. According to Equation (14), the updating of each element in **X** approximates the order of $O((k+2)n_V)$. Hence updating all the elements in **X** follows the order of $O(2k(k+2)n_V^2)$.

Given the complexity of updating variables in **C** and **X**, FSPGA is an algorithm with complexity of $O(n^2)$. In other words, FSPGA is more efficient than the spectral-based clustering algorithms since their computational complexity follows the order of $O(n^3)$. In fact, as the augmented matrix **Y** is always very sparse, theoretically, FSPGA should run faster than those algorithms, such as Affinity Propagation (AP), that also have the complexity of $O(n^2)$. The scalability test shown in Fig. 2 also supports the analysis here.

Regarding the space requirement of FSPGA, it should be noted that FSPGA does not require much memory space when performing the task of discovering fuzzy structural patterns in the attributed graph. This is because FSPGA only stores those non-zero elements in the augmented matrix **Y**. For example, one synthetic dataset used in our experiment contains 100,000 vertices, but there are about 30,000,000 elements in **Y** which are larger than zero. Compared with the full memory space for $100{,}000^2$, 30,000,000 is only 0.3% of the full space. Given the analysis here and the scalability test shown in Fig. 2, FSPGA can be used for discovering fuzzy structural patterns in large attributed graphs.

V. Conclusion

In this paper, FSPGA, which is an algorithm for discovering fuzzy structural patterns in the form of clusters in the attributed graph, is proposed. Compared with prevalent algorithms that take different properties of an attributed graph, including topology, attribute, and both of the aforementioned, FSPGA may find an optimal arrangement of clusters for vertices in an attributed graph by formulating the task as a fuzzy constrained optimization problem. As the adoption of fuzzy set theory when determining the cluster membership, FSPGA can detect overlapping clusters, while most of the prevalent algorithms cannot. The experimental results presented in this paper show that FSPGA may perform robustly and efficiently in different types graph data, compared with the classical, latest graph clustering algorithms, and fuzzy clustering algorithms. In future, we will intend to further improve the efficiency of FSPGA and develop a version of FSPGA that may discover hierarchical structural patterns in attributed graphs.

References

[1] S. Fortunato, “Community detection in graphs,” *Phys. Rep.*, vol. 486, no.3-5, pp. 75-174, Feb. 2010.

[2] J. Leskovec, K. J. Lang, and M. Mahoney, “Empirical comparison of algorithms for network community detection,” in *Proc 19th Int. conf. World Wide Web,* 2010, pp. 631-640.

[3] M. Girvan, and M. E. J. Newman, “Community structure in social and biological networks,” *Proc. Nat. Acad. Sci. U. S. A.*, vol. 99, no. 12, pp. 7821-7826, Jun. 2002.

[4] M. E. J. Newman, “Modularity and community structure in networks,” *Proc. Nat. Acad. Sci. U. S. A.*, vol. 103, no. 23, pp. 8577-8582, Jun. 2006.

[5] A. Clauset, M. E. J. Newman, and C. Moore, “Finding community structure in very large networks,” *Phys. Rev. E*, vol. 70, no. 6, pp. 066111, Dec. 2004.

[6] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *J. Stat. Mech.*, vol. 2008, no. 10, pp. P10008, Oct. 2008.

[7] S. Fortunato, and M. Barthelemy, “Resolution limit in community detection,” *Proc. Nat. Acad. Sci. U. S. A.*, vol. 104, no. 1, pp. 36-41, Jan. 2007.

[8] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek, “Uncovering overlapping community structure of complex networks in nature and society,” *Nature*, vol. 435, pp. 814-818, Jun. 2005.

[9] B. J. Frey, and D. Dueck, “Clustering by passing messages between data points,” *Science*, vol. 16, pp. 972-976, Feb. 2007.

[10] Y. Ahn, J. P. Bagrow, and S. Lehmann, “Link communities reveal multiscale complexity in networks,” *Nature*, vol. 466, pp. 761-764, Aug. 2010.

[11] U. Luxburg, “A tutorial on spectral clustering,” *Statistics and Computing.*, vol. 17, no. 4, pp. 395-426, Dec. 2007.

[12] J. Shi, and J. Malik, “Normalized cuts and image segmentation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888-905, Aug. 2000.

[13] E. M. Airoldi, D. Blei, S. E. Fienberg, and E. P. Xing, “Mixed Membership Stochastic Blockmodels,” *J. Mach. Learn. Res.*, vol. 9, pp. 1981-2014, Sep. 2008.

[14] J. Yang, J. McAuley, and J. Leskovec, “Detecting Cohesive and 2-mode Communities in Directed and Undirected Networks,” in *Proc. 7th ACM Int. Conf. Web Search and Data Mining*, 2014, pp. 323-332.

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

[16] M. Frank, A. P. Streich, D. Basin, and J. M. Buhmann, “Multi-assignment clustering for boolean data,” *J. Mach. Learn. Res.*, vol. 13, pp. 459-489, Feb. 2012.

[17] Y. Tian, R. A. Hankins, and J. M. Patel, "Efficient aggregation for graph summarization," in *Proc. of the 2008 ACM Int. Conf. Management of Data,* pp. 567-580, 2008.

[18] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," in *Proc. VLDB* 2009, pp. 718-729.

[19] Y. Zhou, H. Cheng, and J. X. Yu, "Clustering large attributed graphs: an efficient incremental approach," in *Proc. IEEE 10th Int. Conf. Data Mining,* 2010, pp. 689-698.

[20] S. Gunnermann, B. Boden, I. Farber, and T. Seidl, "Efficient mining of combined subspace and subgraph clusters in graphs with feature vectors," in *Proc. PAKDD*, 2013, pp.261-275.

[21] Z. Xu, Y. Ke, Y. Wang, H. Cheng and J. Cheng, "Gbagc: a general bayesian framework for attributed graph clustering," *ACM Trans. Knowl. Discov. Data*, vol. 9, no. 1, article 5, 2014.

[22] J. Yang, J. McAuley, and J. Leskovec, "Community detection in networks with node attributes," in *Proc. IEEE 13th Int. Conf. Data Mining*, 2013, pp. 1151-1156.

[23] J. McAuley, and J. Leskovec, "Discovering social circles in ego networks," *ACM Trans. Knowl. Discov. Data*, vol. 8, no. 1, article 4, 2014.

[24] T. He, and K. C. C. Chan, "Evolutionary community detection in social networks," in *Proc. CEC*, 2014, pp. 1496-1503.

[25] D. Blei, "Probabilistic topic models," *Commun. ACM*, vol. 55, no. 4, pp. 77-84, 2012.

[26] R. Nallapati, A. Ahmed, E. P. Xing, and W. W. Cohen, "Joint latent topic models for text and citations," in *Proc. 14th ACM Int. Conf. Kowl. Discov. Data Mining, 2008*, pp. 542-550.

[27] J. Chan, and D. Blei, "Relational topic models for document networks," in *Proc. 12th Int. Conf. Artificial Intel. Stat.*, 2009, pp. 81-88.

[28] Y. Sun, J. Han, J. Gao, and Y. Yu, "itopicmodel: information network-integrated topic modeling," in *Proc. IEEE 9th Int. Conf. Data Mining*, 2009, pp. 493-502.

[29] T. Yang, R. Jin, Y. Chi, and S. Zhu, "Combining link and content for community detection: a discriminative approach," in *Proc. 15th ACM Int. Conf. Kowl. Discov. Data Mining*, 2009, pp. 927-936.

[30] R. Balasubramanyan, and W. W. Cohen, "Block-LDA: Jointly modeling entity-annotated text and entity-entity links," in *Proc. SIAM Int. Conf. Data Mining*, 2011, pp. 450-461.

[31] J. C. Bezdek, R. Ehrlich, W. Full, "FCM: the fuzzy c-means clustering algorithm," *Computers and Geosciences*, vol.10, no. 2-3, pp. 191-203, 1984.

[32] M. A. Khalilia, J. Bezdek, M. Popescu, and J. M. Keller, "Improvements to the relational fuzzy c-means clustering algorithm," *Pattern Recog.*, vol. 47, no. 12, pp.3920-3930, 2014.

[33] R. J. Hathaway, J. C., Bezdek, "Switching regression models and fuzzy clustering," *IEEE Trans. Fuzzy Systems*, vol. 1, no. 3, pp. 195-204, 1993.

[34] N. R. Pal, K. Pal, and J. M. Keller, "A possibilistic fuzzy c-means clustering algorithm," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 4, pp. 517-530, 2005.

[35] L. Silva, R. Moura, A. M. P. Canuto, R. H. N. Santiago, and B. Bedregal, "An interval based frame work for fuzzy clustering applications," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 6, pp. 2174-2187, Mar. 2015.

[36] T. M. Nguyen, and Q. M. J. Wu, "Dynamic fuzzy clustering and its application in motion segmentation," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 6, pp. 1019-1031, Dec. 2013.

[37] I. Chiang, C. C. Liu, Y. Tsai, and A. Kumar, "Discovering latent semantics in web documents using fuzzy clustering," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 6, pp. 2122-2134, Dec. 2015.

[38] L. Hu and K. C. C. Chan, "Fuzzy Clustering in a Complex Network Based on Content Relevance and Link Structures," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 2, pp. 456-470, Apr. 2016.

[39] C. Stark, B. Breitkreutz, T. Reguly, L. Boucher, A. Breitkreutz, and M. Tyers, "BioGRID: A General Repository for Interaction Datasets," *Nucleic Acids Research*, vol. 34, no. Suppl. 1, pp. D535-D539, 2006.

[40] N. J. Krogan, *et al.*, "Global Landscape of Protein Complexes in the Yeast Saccharomyces cerevisiae," *Nature*, vol. 440, no. 7084, pp. 637-643, 2006.

[41] I. Xenarios, L. Salwinski, X. J. Duan, P. Higney, S. Kim, and D. Eisenberg, "DIP, the Database of Interacting Proteins: A Research Tool for Studying Cellular Networks of Protein Interactions," *Nucleic Acids Research*, vol. 30, no. 1, pp. 303-305, 2002.

[42] M. Ashburner, *et al.*, "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics*, vol. 25, no. 1, pp. 25-29, 2000.

[43] S. Pu, J. Wong, B. Turner, E. Cho, and S. J. Wodak, "Up-to-date catalogues of yeast protein complexes," *Nucleic Acids Res.*, vol. 37, no. 3, pp. 825-831, Feb. 2009.

[44] K. C. C. Chan, A. K. C. Wong, and D. K. Y. Chiu, "Learning sequential patterns for probabilistic inductive prediction," *IEEE Trans. Systems, man and cybernetics,* vol. 24, no. 10, pp. 1532-1547, Oct. 1994.

[45] J. Y. Ching, A. K. C. Wong, and K. C. C. Chan, "Class-dependent discretization for inductive learning from continuous and mixed-mode data," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 17, no. 7, pp. 641-651, Jul. 1995.

[46] G. Qi, C. C. Aggarwal, and T. Huang, "Community detection with edge content in social media networks," in *Proc. IEEE 28th Int. Conf. Data Engineering*, 2012, pp. 534-545.

[47] L. Xie, and G. Beni, "A validity measure for fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 8, pp. 841-847, Aug. 1991.

[48] D. T. Anderson, A. Zare, and S. Price, "Comparing Fuzzy, Probabilistic, and Possibilistic Partitions Using the Earth Mover's Distance," *IEEE Trans. Fuzzy Syst.*, vol. 21, no. 4, pp. 766-775, Aug. 2013.

[49] D. T. Anderson, J. C. Bezdek, M. Popescu, and J. M. Keller, "Comparing Fuzzy, Probabilistic, and Possibilistic Partitions," *IEEE Trans. Fuzzy Syst.*, vol. 18, no. 5, pp. 906-918, Oct. 2013.

Tiantian He received the BEng degree in computer science from North China University of Technology, Beijing, China in 2008, the MSc degree in information systems, and the PhD degree in computer science from the Hong Kong Polytechnic University in 2012 and 2017, respectively. Currently he is a postdoctoral research fellow in Department of Computing, the Hong Kong Polytechnic University. His research interests include machine learning, data mining, and bioinformatics.

Keith C. C. Chan received the BMath (Hons.) degree in computer science and statistics in 1984 and the MASc and PhD degrees in systems design engineering in 1985 and 1989, respectively, from the University of Waterloo, Ontario, Canada. Soon after graduation, he worked as a software analyst for the development of multimedia and software engineering tools at the IBM Canada Laboratory in Toronto, Canada. He joined the Hong Kong Polytechnic University in 1994, where he is currently a professor in the Department of Computing. His research interests include bioinformatics, data mining, and software engineering. He has over 250 publications in these areas, and his research is supported both by government research funding agencies and the industry.