MISAGA: An Algorithm for Mining Interesting Sub-Graphs in Attributed Graphs

Tiantian He, and Keith C. C. Chan

***Abstract*—An attributed graph contains vertices that are associated with a set of attribute values. Mining clusters or communities, which are interesting sub-graphs in the attributed graph is one of the most important tasks of graph analytics. Many problems can be defined as the mining of interesting sub-graphs in attributed graphs. Algorithms that discover sub-graphs based on pre-defined topologies cannot be used to tackle these problems. To discover interesting sub-graphs in the attributed graph, we propose an algorithm called MISAGA. MISAGA performs its tasks by first using a probabilistic measure to determine whether the strength of association between a pair of attribute values is strong enough to be interesting. Given the interesting pairs of attribute values, then the degree of association is computed for each pair of vertices using an information theoretic measure. Based on the edge structure and degree of association between each pair of vertices, MISAGA identifies interesting sub-graphs by formulating it as a constrained optimization problem and solves it by identifying the optimal affiliation of sub-graphs for the vertices in the attributed graph. MISAGA has been tested with several large-sized real graphs and is found to be potentially very useful for various applications.**

***Index Terms*—attributed graph, interesting sub-graphs, graph clustering, clustering algorithms, community detection, optimization, information theoretic measure**

I. Introduction

An *attributed graph* contains attributed vertices connected by edges. Each attributed vertex in an attributed graph is associated with a set of attribute values. Many real-world problems, such as the discovering of communities in a social network, or the discovering of protein complexes in a protein-protein interaction (PPI) network can be modeled and solved as mining clusters, or communities in an attributed graph.

For example, if each participant of a social network is represented as a vertex of an attributed graph, then the information such as the age, occupation, interests and hobbies of the participant can be considered as the set of attribute values associated with the vertex. Two participants that interact with each other can be represented as an edge that connects the vertices representing each participant respectively. Detecting communities in a social network can therefore be formulated as the problem of discovering clusters in the attributed graph that represents the social network.

The problem of discovering clusters in a larger graph has drawn much attention in recent years [1] [2]. For example, a number of what are referred to as *graph clustering* algorithms have been developed and used to discover graph clusters. These graph clusters are sub-graphs satisfying some pre-specified topologies or edge structures, such as the vertices being more densely connected within the same sub-graph, etc. Pre-specified topologies, such as the vertex and edge densities, can be captured in different measures of interestingness [52], which can be used to evaluate the quality of discovered clusters. Hence, a graph cluster is a sub-graph satisfying a particular interestingness measure. For example, in [3], an algorithm that detects communities based on specific requirements on edge centrality is presented. In [4], another measure, called *modularity,* which is defined as a function of the differences in density within and between graph clusters and a *null-graph* (in which vertices are connected randomly) is proposed. Based on it, several algorithms, such as CNM [5] and BGLL [6], have been developed to search for graph clusters based on the optimization of modularity. In [7] and [8], the formalism of a random graph is introduced. This formalism shows clusters smaller than a certain size cannot be detected by these algorithms that detect for clusters based on modularity optimization.

There are other algorithms that discover graph clusters based on other properties of network topologies. In [9], for example, an algorithm is proposed to detect graph clusters based on the concept of a clique using a *clique percolation method* (CPM). In [10], a graph clustering method called *affinity propagation* (AP) is proposed to detect clusters based on the similarities between cluster centers and other vertices. In [11], a method is proposed to detect graph clusters by introducing the concept of a *link graph* to facilitate optimization of edge densities. In [12], given that vertices in the same cluster may share similar edge structure, spectral clustering is proposed to consider *normalized cuts* [13] for cluster identification. In [47], another spectral based clustering method is proposed. By making use of deep linear coding, this algorithm has been shown to improve the efficiency of other spectral algorithms for graph clustering. In [14], a semi-supervised algorithm for detecting clusters in graphs is proposed. Besides of making use of spectral techniques, the proposed algorithm also utilizes the prior knowledge determining the cluster affiliation of some vertices

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).

to obtain a better result. In [15] and [16], two approaches based on the use of Mixed Membership Stochastic Block models (MMSB) are proposed, to detect graph clusters by optimizing the posterior probability that a pair of vertices are actually connected. The higher the probability, therefore, the greater the sub-graph density. In [45], another MMSB-based method is proposed. Compared to other similar methods, it makes use of graph statistics, rather than posterior probabilities, to determine cluster membership of a vertex.

There are some algorithms for detecting graph clusters based on other techniques. For example, an algorithm called MEAs-SN [17] has been proposed to detect clusters in signed social graphs. MEAs-SN is a multi-objective evolutionary algorithm which can detect social clusters based on the number of intra-cluster positive edges and negative edges between clusters. In [18], another evolutionary algorithm (DCRO) for detecting graph clusters is proposed. Different from MEAs-SN, DCRO detects graph clusters based on an efficient multi-objective evolutionary algorithm called Chemical reaction optimization (CRO) [19].

Identification of clusters that is based solely on graph topologies does not take into consideration attribute values associated with the vertices. In the case that such attribute values are useful for the discovering and understanding of the sub-graphs identified, many graph clustering algorithms cannot be used.

To consider attributes in graph clustering, some attempts have been made to make use of the *k*-means algorithm [20] to group vertices with higher similarity of attributes into the same clusters. In [21], an algorithm (MAC) that is based on a probabilistic generative model is proposed for clustering vertices that are labeled with Boolean attribute values. In [22], a graph summarization algorithm called *k*-SNAP is proposed to detect graph clusters by grouping vertices into the same cluster according to a similarity measure of the attribute values.

These clustering algorithms are not very well suited for the task to discover interesting sub-graphs in attributed graphs. While they take attribute values into consideration, they do not consider graph topologies during clustering and this is not desirable as the connecting of vertices does provide useful structural information which should be considered.

To consider both attributes and structures, several algorithms can be used. In [23], SA-Cluster is proposed to detect graph clusters using a *neighborhood random walk model*. Based on it, cluster membership of each vertex is determined at the time that the transition matrix reaches steady-state. In [24], inc-Cluster is proposed using the same random walk model as the SA-Cluster except that its efficiency is improved using an incremental method to compute the transition matrix. In [48], SCMAG is proposed to detect network communities by identifying the network sub-spaces in which vertices are densely connected and associated with similar attribute values. In [25], EDCAR is proposed to mine sub-graphs by grouping together vertices that are densely connected and share similar attribute values. In addition to graph structures, these algorithms are able to take into consideration vertex attributes. However, they are more graph partitioning algorithms that are not developed to discover overlapping sub-graphs. When considering attribute values, their focus is mainly on the number of values that are the same without considering relevancy and attribute values that are co-related but not exactly the same. As a result, interesting sub-graphs may sometimes be overlooked.

In addition to the above algorithms, some algorithms detect graph clusters by utilizing generative models. In [26], 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 [27], an algorithm, called CESNA, is proposed to make use of a statistical model to determine the posterior probability that pairwise vertices are connected given particular edge structures and attributes in a graph cluster. Cluster membership is determined when posterior probability is maximized. In [28], an algorithm called Circles is proposed to detect clusters in social network graphs. By treating user profiles as attributes, Circles determines cluster 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 clusters. In [29], an evolutionary community detection algorithm, called ECDA, is proposed to detect for graph clusters in social networks by considering network connections and attribute labeled to each pair of vertices. In [46], another evolutionary algorithm, EGCPI is proposed to detect clusters in PPI network data based on edge structures and attribute similarity. EGCPI is a two-step approach, which firstly identifies densely-connected regions in a network using an evolutionary approach. It then identifies sub-networks of protein complexes by grouping together vertices whose attribute values are sufficiently homogeneous. These mentioned algorithms also face the same disadvantages as SA-Cluster and inc-Cluster and are therefore not ideal for the problem that we propose here.

Inspired by topic modeling [30], several topic-model-based approaches, such as Link-PLSA-LDA [31], Relational Topic Model [32], iTopicModel [33], PL-DC [34] and Block-LDA [35] can also be used to identify graph clusters mainly in document networks with words as attributes of vertices and citations as edges of an attributed graph. 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 [27].

When it comes to discovering interesting sub-graphs in attributed graphs, we need a computationally efficient algorithm that is able to handle large volumes of graph data while at the same time consider both graph topologies and vertex attributes in performing its tasks. The algorithm also needs to be able to distinguish relevant from irrelevant attribute values and determine if these relevant attribute values are associated with each other even though they are not exactly the same. In other words, the measure of interestingness we are looking for is that each sub-graph contains structurally cohesive vertices as well as relevant attribute values associated with the vertices.

For such an algorithm, we propose MISAGA (Mining Interesting Sub-graphs in Attributed Graph Algorithm). MISAGA performs its tasks by using a statistical measure that identifies attribute values that have interesting associations with each other. If the attribute values of a pair of vertices have interesting associations, then a measure called degree of association is computed. With such measures, the problem of discovering interesting sub-graphs is formulated and solved as a constrained optimization problem, MISAGA can find an optimal sub-graph arrangement satisfying the interestingness measure which takes into the consideration both edge density as well as the correlation of attribute values with each vertex.

For performance evaluation, MISAGA is tested with both synthetic and real data sets including real social and PPI network data. The experimental results are verified against known ground-truth data. The findings show that the sub-graphs discovered by MISAGA can be very meaningful.

In Section 2 below, the problem and the mathematical notations used to explain MISAGA are introduced. In Section 3, how the problem of discovering interesting sub-graphs in attributed graph is formulated as a constrained optimization problem is discussed and the details of MISAGA is presented. In Section 4, we present the results of the experiments performed to evaluate the performance of MISAGA. Finally, in Section 5, we present the conclusion with a summary of the contributions of the paper and a proposal for future work.

II. Identifying interesting associations

Given an attributed graph, containing n_V vertices, n_E edges, and n_Λ attribute values that might be associated with each vertex, we can represent the graph as G = (V, E, Λ), where $V = \{v_i \mid 1 \le i \le n_V\}$, $E = \{e_{ij} \mid 1 \le i, j \le n_V, i \ne j, v_i \text{ and } v_j \text{ are connected}\}$, and $\Lambda = \{att_i \mid 1 \le i \le n_\Lambda\}$ denote the set of vertices, edges, and attribute values, respectively.

Utilizing the set E, MISAGA constructs an adjacency matrix **Y** of dimensions, n_V by n_V, to represent the connections between the vertices in G so that an entry, y_{ij}, in **Y** has the value, 1, if v_i and v_j are connected and, 0, if they are not.

A. Identifying interesting associations among attributes

In order to avoid noise and to ensure that the attribute values it considers for mining of interesting sub-graphs are the relevant and interesting ones, MISAGA performs its tasks by making use of a statistical measure to determine whether or not a pair of attributes are significantly associated.

To illustrate how this can be done, let us consider two attribute values, att_i and att_j. Let us use $o(att_i, att_j)$ to denote the number of edges that connect two vertices which are associated with att_i and att_j, respectively. We use $o(+att_i)$ or $o(+att_j)$ to represent the number of edges that connect vertices which are associated with att_i or att_j. We use $e(att_i, att_j)$ 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 $[o(+att_i)o(+att_j)]/n_E$ [36]. Hence, MISAGA considers att_i and att_j as having significant interesting association with each other if $o(att_i, att_j)$ is sufficiently different from $e(att_i, att_j)$. To determine if the difference is significant, we make use of the following test statistics:

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

In [36] and [37], 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, MISAGA filters out attribute values that are not relevant for the identification of interesting sub-graphs. We use an n_Λ-by-n_Λ matrix, **S**, to represent whether two attribute values in Λ are significantly associated. Specifically, say s_{ij}, an element in **S**, it equals to 1 if att_i and att_j are significantly associated with each other, and 0, if they are not.

B. Determining the degree of attribute association

For any pair of vertices in an attributed graph, more than one attribute value between the two vertices may be found to be significantly associated with each other. The total degree of attribute association should therefore vary according to the number and degree of such associations. As such information can provide very useful criteria for interesting sub-graph identification, MISAGA computes a measure of total strength of association. For such purpose, MISAGA adopts an information theoretical measure [20] defined as follows.

Let $\mathbf{a}_i$ and $\mathbf{a}_j$ be two n_Λ-by-1 vectors representing whether the attribute values in Λ are associated with two vertices v_i and v_j, respectively. An element, such as, a_{ik}, the *k*th element in $\mathbf{a}_i$, is equal to 1 if att_k in Λ is associated with vertex v_i, and vice versa. If two vertices, say v_i and v_j, are characterized by the attribute values att_k and att_m, respectively, then we can use $\Pr(a_{ik}, a_{jm})$ to denote the probability that two connected vertices are characterized by att_k and att_m, this probability can be computed as $o(att_k, att_m)/n_E$. Similarly, we can use $\Pr(a_{ik})$ or $\Pr(a_{jm})$ to denote the probability that an edge may connect two vertices that are characterize by att_k or att_m, and these two probabilities can be computed as $o(+att_k)/n_E$, and $o(+att_m)/n_E$, respectively. Given these probabilities, we can define a *total degree of association* between v_i and v_j as:

$$r(v_i, v_j) = \sum_k \sum_m (s_{km} \cdot a_{ik} \cdot a_{jm}) \Pr(a_{ik}, a_{jm}) \cdot \log \frac{\Pr(a_{ik}, a_{jm})}{\Pr(a_{ik}) \cdot \Pr(a_{jm})} \quad (2)$$

$r(v_i, v_j)$ can be considered as a measure of the average reduction in uncertainty given the attribute values that v_i and v_j are characterized with [20]. In fact, it should be noted that the higher the value of *r* between two vertices is, the stronger they are associated with each other given the attribute values that they are characterized by. The magnitude of $r(v_i, v_j)$ increases with both the number of attribute values that are found to be associated with each other and the relative magnitude of the probabilities measures that reflect such degree of association. For the purpose of normalization, MISAGA computes an entropy measure as follows:

$$H(v_i, v_j) = -\sum_k \sum_m (s_{km} \cdot a_{ik} \cdot a_{jm}) \cdot \Pr(a_{ik}, a_{jm}) \cdot \log \Pr(a_{ik}, a_{jm}) \quad (3)$$

Given (2) and (3), the *normalized degree of association* between the vertices v_i and v_j can be obtained as:

$$doa(v_i, v_j) = \frac{r(v_i, v_j)}{H(v_i, v_j)} \quad (4)$$

The measure, $doa(v_i, v_j)$, can be interpreted as the information redundancy of the attribute values that are associated with v_i and v_j. The magnitude of $doa(v_i, v_j)$ ranges from 0 to 1. A greater value of $doa(v_i, v_j)$ means that the attribute values of the pair of vertices, v_i and v_j are more strongly associated with each other, and so, their being members in the same group may reflect more interesting patterns.

After obtaining the *doa*'s of all pairwise vertices with interesting and relevant association between attribute values, MISAGA constructs an n_V-by-n_V *attribute-association matrix,* **A** for storing the *doa*'s. Using **Y** and **A**, MISAGA searches for an optimal solution to a constrained optimization problem that considers both edge structure and attribute association for the mining of interesting sub-graphs as clusters in an attributed graph.

III. The optimization algorithm

Having obtained the adjacency matrix **Y** and the attribute-association matrix **A**, MISAGA then attempts to identify interesting sub-graphs in G by utilizing the information in these two matrices. In this section, we describe how this process can be formulated as an optimization problem and how MISAGA solves the problem.

A. The objective function for mining interesting sub-graphs

To formulate the sub-graph identification problem as an optimization problem, let us define k to be the number of interesting sub-graphs in G that is to be identified. Given k, we first introduce two n_V-by-k auxiliary matrices, **D** and **B**, which represent the strength that each vertex belongs each of the k sub-graph, taking into consideration structure and attribute associations, respectively. With **D** and **B,** we can introduce a sub-graph membership matrix **C**, which has the dimension of n_V by k. Each element of **C**, say c_{ij}, indicates the strength of membership for vertex i to belong to sub-graph j so that the larger the value of c_{ij}, the greater the affinity between vertex i and sub-graph j.

Given the adjacency matrix **Y**, the attribute-association matrix, **A**, the auxiliary matrices **D** and **B**, and the sub-graph membership matrix **C**, MISAGA attempts to maximize the following objective function:

$$\begin{aligned} &\text{maximize} \\ &O = \alpha tr(\mathbf{C}^{\mathrm{T}}\mathbf{YD}) + (1-\alpha) tr(\mathbf{C}^{\mathrm{T}}\mathbf{AB}) \\ &\quad -\frac{1}{2}\left[\left|\mathbf{C}\right|_F^2 + \left|\mathbf{D}\right|_F^2 + \left|\mathbf{B}\right|_F^2 + \left|\mathbf{BC}^{\mathrm{T}}\right|_F^2 + \left|\mathbf{DC}^{\mathrm{T}}\right|_F^2 + \lambda\left|\mathbf{Ce_1} - \mathbf{e}_2\right|_F^2\right] \\ &\textit{subject to } \mathbf{C} \geq 0, \mathbf{D} \geq 0, \mathbf{B} \geq 0 \end{aligned} \quad (5)$$

where (i) $|\mathbf{C}|^2_F$, $|\mathbf{D}|^2_F$, and $|\mathbf{B}|^2_F$ are the l_2-norm of matrices **C**, **D**, and **B**, which are used respectively to smooth the variables in these matrices, (ii) $|\mathbf{DC}|^2_F$, and $|\mathbf{BC}|^2_F$ are the l_2-norm of the products of **D** and **C**, and **B** and **C**, respectively, (iii) α is a parameter that is used to adjust the relative weighting between edge density and attribute association to be considered when MISAGA searches for the optimal **C**, (iv) $\mathbf{e_1}$ and $\mathbf{e_2}$ are k-by-1 and n_V-by-1 vectors in which all elements are set to 1, (v) $|\mathbf{Ce_1}\text{-}\mathbf{e_2}|^2_F$ is an l_2-norm which is used to regulate the aggregation of the variables in each row of **C** so that they can be approximated by 1, which can be used conveniently when comparing the strength of sub-graph affiliation between different vertices. (vi) λ is a non-negative factor which is used to control the effects of the regulation term mentioned in (v). By utilizing the proposed objective function, MISAGA possesses the following advantages when used to identify interesting sub-graphs in attributed graphs.

First, by introducing the two auxiliary matrices **D** and **B**, MISAGA can identify interesting sub-graphs by taking into consideration both graph topology and attribute association between the pairwise vertices in a graph. To explain how MISAGA determines the sub-graph membership of the vertices, let us consider the first two terms of the objective function: $tr(\mathbf{C}^{\mathrm{T}}\mathbf{YD})$ and $tr(\mathbf{C}^{\mathrm{T}}\mathbf{AB})$.

These terms are used to aggregate the strength of the topology and attribute association of all sub-graphs respectively. Each element in $tr(\mathbf{C}^{\mathrm{T}}\mathbf{YD})$ is used to aggregate the total number of edges within a sub-graph, given the values of the entries in **C** and **D.** $tr(\mathbf{C}^{\mathrm{T}}\mathbf{YD})$ can be optimized only when all the vertices can be assigned to appropriate sub-graphs in which they are connected by more intra-edges. In other words, the corresponding entries in **C** and **D**, say c_{ij} and d_{ij}, which are used to represent sub-graph membership, and the structural strength for vertex i to belong to sub-graph j, should be relatively high when vertex i are connected by relatively more vertices in sub-graph j.

In the case of $tr(\mathbf{C}^{\mathrm{T}}\mathbf{AB})$, it aggregates the total degree of associations between pairwise vertices within each sub-graph. $tr(\mathbf{C}^{\mathrm{T}}\mathbf{AB})$ can be optimized only when all the vertices are assigned to the sub-graphs in which the degrees of attribute association between the pairwise vertices are high.

Given the objective function as shown in (5), MISAGA attempts to find a sub-graph membership matrix **C**, that can be used to assign each vertex into a sub-graph in such a way that it is more connected to and shares a higher degree of attribute association with the other vertices in the same sub-graph. If such an optimal sub-graph assignment can be found, the optimization process adopted by MISAGA can find corresponding optimum values in the entries for this vertex in **D** and **B** and as a result, the corresponding element in **C**, which is used to represent the sub-graph membership between that vertex and that sub-graph is also at its optimum.

However, one may notice that $tr(\mathbf{C}^\mathbf{T}\mathbf{YD})$ and $tr(\mathbf{C}^\mathbf{T}\mathbf{AB})$ also increase when the variables in **D**, **B**, and **C** simply increase. Thus, we use $|\mathbf{DC}|^2_F$, and $|\mathbf{BC}|^2_F$ to penalize improper variations of the variables in **D**, **B**, and **C** so that, only when the elements of **D**, **B**, and **C** are assigned with appropriate values, the objective function O can be maximized. In such case, **C** contains the membership of the most interesting sub-graphs in each of which vertices are densely connected and their attribute values are significantly associated.

The other advantage that the optimization process adopted by MISAGA is that, by using the penalty term $\lambda|\mathbf{Ce_1}\text{-}\mathbf{e_2}|^2_F$, the aggregation of each row in **C** can be controlled to be around 1 and this can make comparison of the strength of sub-graph affiliation between different vertices and the extracting of overlapping sub-graphs to be more convenient. With these advantages, therefore, when an optimal value for the proposed objective function can be determined, the sub-graphs found by MISAGA can take both edge structure and attribute association into consideration at the same time.

B. The iterative updating algorithm

The proposed objective function is a constrained quadratic function. It is non-convex to **C**, **D**, and **B** simultaneously, but it is convex to **C**, **D**, or **B** when keeping the other two matrices unchanged. Based on this property, a series of updating rules for optimizing (5) can be obtained.

1) Updating rule for C

Let β_{ij} be the Lagrange multiplier for the constraint $c_{ij} \geq 0$. The Lagrange function L for **C** is

$$L(\mathbf{C},\boldsymbol{\beta}) = O - tr(\boldsymbol{\beta}^\mathbf{T}\mathbf{C}) \quad (6)$$

where $\boldsymbol{\beta} = [\beta_{ij}]$ is the matrix of Lagrange multipliers for the non-negativity of **C**. Based on the KKT condition for constrained optimization, we have the following

$$\begin{aligned}&\frac{\partial L}{\partial \mathbf{C}} = \alpha\mathbf{YD} + (1-\alpha)\mathbf{AB} + \lambda\mathbf{e}_2\mathbf{e}_1^\mathbf{T}\\ &-\mathbf{CD}^\mathbf{T}\mathbf{D} - \mathbf{CB}^\mathbf{T}\mathbf{B} - \mathbf{C} - \lambda\mathbf{Ce}_1\mathbf{e}_1^\mathbf{T} - \boldsymbol{\beta} = 0\\ &\boldsymbol{\beta}\circ\mathbf{C} = \mathbf{0}\\ &\boldsymbol{\beta} \geq 0\end{aligned} \quad (7)$$

where "∘" means the Hadamard product of two matrices with the same dimension. Based on (7), we have the element wise equation system for each element in **C**

$$\begin{aligned}&[\alpha\mathbf{YD} + (1-\alpha)\mathbf{AB} + \lambda\mathbf{e}_2\mathbf{e}_1^\mathbf{T}]_{ij}\\ &-(\mathbf{CD}^\mathbf{T}\mathbf{D} + \mathbf{CB}^\mathbf{T}\mathbf{B} + \mathbf{C} + \lambda\mathbf{Ce}_1\mathbf{e}_1^\mathbf{T})_{ij} - \beta_{ij} = 0\\ &\beta_{ij}\circ c_{ij} = 0\\ &\beta_{ij} \geq 0\end{aligned} \quad (8)$$

Given the first equation in (8), we have

$$\begin{aligned}&[\alpha\mathbf{YD} + (1-\alpha)\mathbf{AB} + \lambda\mathbf{e}_2\mathbf{e}_1^\mathbf{T}]_{ij}\\ &-(\mathbf{CD}^\mathbf{T}\mathbf{D} + \mathbf{CB}^\mathbf{T}\mathbf{B} + \mathbf{C} + \lambda\mathbf{Ce}_1\mathbf{e}_1^\mathbf{T})_{ij} = \beta_{ij}\end{aligned} \quad (9)$$

Using (9) to replace β_{ij} in the equation of Hadamard product, we have the iterative updating rule for **C**

$$c_{ij} \leftarrow c_{ij}\frac{[\alpha\mathbf{YD} + (1-\alpha)\mathbf{AB} + \lambda\mathbf{e}_2\mathbf{e}_1^\mathbf{T}]_{ij}}{(\mathbf{CD}^\mathbf{T}\mathbf{D} + \mathbf{CB}^\mathbf{T}\mathbf{B} + \mathbf{C} + \lambda\mathbf{Ce}_1\mathbf{e}_1^\mathbf{T})_{ij}} \quad (10)$$

2) Updating rule for D

Let γ_{ij} be the Lagrange multiplier for the constraint $d_{ij} \geq 0$, hence the Lagrange function L for **D** is

$$L(\mathbf{D},\boldsymbol{\gamma}) = O - tr(\boldsymbol{\gamma}^\mathbf{T}\mathbf{D}) \quad (11)$$

where $\boldsymbol{\gamma} = [\gamma_{ij}]$ is the matrix of Lagrange multipliers for the non-negativity of **D**. Based on the KKT condition, we have

$$\begin{aligned}&\frac{\partial L}{\partial \mathbf{D}} = \alpha\mathbf{YC} - \mathbf{DC}^\mathbf{T}\mathbf{C} - \mathbf{D} - \boldsymbol{\gamma} = 0\\ &\boldsymbol{\gamma}\circ\mathbf{D} = \mathbf{0}\\ &\boldsymbol{\gamma} \geq 0\end{aligned} \quad (12)$$

Given the equation system (12), the element wise updating rule for **D** can be derived

$$d_{ij} \leftarrow d_{ij}\frac{(\alpha\mathbf{YC})_{ij}}{(\mathbf{DC}^\mathbf{T}\mathbf{C} + \mathbf{D})_{ij}} \quad (13)$$

3) Updating rule for B

Let η_{ij} be the Lagrange multiplier for the constraint $b_{ij} \geq 0$, hence the Lagrange function L for **B** is

$$L(\mathbf{B},\boldsymbol{\eta}) = O - tr(\boldsymbol{\eta}^\mathbf{T}\mathbf{B}) \quad (14)$$

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

$$\begin{aligned}&\frac{\partial L}{\partial \mathbf{B}} = (1-\alpha)\mathbf{AC} - \mathbf{BC}^\mathbf{T}\mathbf{C} - \mathbf{B} - \boldsymbol{\eta} = 0\\ &\boldsymbol{\eta}\circ\mathbf{B} = \mathbf{0}\\ &\boldsymbol{\eta} \geq 0\end{aligned} \quad (15)$$

Given the equation system (15), the element wise updating rule for **B** can be derived

$$b_{ij} \leftarrow b_{ij}\frac{[(1-\alpha)\mathbf{AC}]_{ij}}{(\mathbf{BC}^\mathbf{T}\mathbf{C} + \mathbf{B})_{ij}} \quad (16)$$

While keeping other matrices unchanged, these updating rules will guide **C**, **D** and **B** to identify the local optima in each iteration, respectively.

C. Convergence analysis of the proposed updating rules

To prove the convergence of the algorithm, we may make use of one property of an auxiliary function that is also used in the proof of the Expectation-Maximization algorithm [38]. The property of the auxiliary function is described as the following. If there exists an auxiliary function satisfying the conditions that $Q(x, x') \leq F(x)$ and $Q(x, x) = F(x)$, then F is non-decreasing under the updating rule that

$$x^{t+1} = \arg\max_x Q(x, x^t) \quad (17)$$

The equality $F(x^{t+1}) = F(x^t)$ holds only when x is a local maximum of $Q(x, x')$. By iteratively updating x according to (17), F will converge to the local maximum $x_{max} = \text{argmax}_x F(x)$. By defining an appropriate auxiliary function for O, we may show the convergence of (5).

First, we may prove the convergence of the updating rule (10). Let c_{ij} be any element in **C**, O_{cij} be the partial of (5) that is related to c_{ij}, $O_{cij}(c'_{ij})$ be the partial objective value of (5) that is related to c_{ij} when c_{ij} is equal to some value, say c'_{ij}. Since the updating rule for **C** is element wise, it is sufficient to show O_{cij} is non-decreasing according to the updating rule (10). To prove this, we define the following auxiliary function for O_{cij}:

$$Q(c,c_{ij}^t)=O_{c_{ij}}(c_{ij}^t)+O'_{c_{ij}}(c-c_{ij}^t) -\frac{(\mathbf{CD}^{\mathrm{T}}\mathbf{D}+\mathbf{CB}^{\mathrm{T}}\mathbf{B}+\mathbf{C}+\lambda\mathbf{Ce}_1\mathbf{e}_1^{\mathrm{T}})_{ij}}{2c_{ij}^t}(c-c_{ij}^t)^2 \tag{18}$$

where O'_{cij} is the first order partial derivative relevant to c_{ij}. Although the auxiliary function is defined in (18), we need to prove it satisfies the aforementioned conditions. Apparently, $Q(c, c) = O_{cij}(c)$. Hence, the left we need to prove is $Q(c, c^t_{ij}) \le O_{cij}(c)$. To prove this, we compared $Q(c, c^t_{ij})$ shown in (18) with the Taylor expansion of O_{cij} near to c^t_{ij}

$$O_{c_{ij}}(c)=O_{c_{ij}}(c_{ij}^t)+O'_{c_{ij}}(c-c_{ij}^t)+\frac{1}{2}O''_{c_{ij}}(c-c_{ij}^t)^2 \tag{19}$$

where O'_{cij} and O''_{cij} are the first and second order partial derivatives relevant to c_{ij}. Note that

$$O'_{c_{ij}}=\frac{\partial O}{\partial c_{ij}}=\begin{bmatrix}\alpha\mathbf{YD}+(1-\alpha)\mathbf{AB}\\-\mathbf{CD}^{\mathrm{T}}\mathbf{D}-\mathbf{CB}^{\mathrm{T}}\mathbf{B}-\mathbf{C}-\lambda\mathbf{Ce}_1\mathbf{e}_1^{\mathrm{T}}+\lambda\mathbf{e}_2\mathbf{e}_1^{\mathrm{T}}\end{bmatrix}_{ij}$$
$$O''_{c_{ij}}=\frac{\partial^2 O}{\partial(c_{ij})^2}=-\left[\mathbf{D}^{\mathrm{T}}\mathbf{D}+\mathbf{B}^{\mathrm{T}}\mathbf{B}+1+\lambda\mathbf{e}_1\mathbf{e}_1^{\mathrm{T}}\right]_{jj} \tag{20}$$

Using (20) to replace the relevant terms in (19), we can see that if $Q(c, c^t_{ij}) \le O_{cij}(c)$, the following inequality must hold

$$-\frac{(\mathbf{CD}^{\mathrm{T}}\mathbf{D}+\mathbf{CB}^{\mathrm{T}}\mathbf{B}+\mathbf{C}+\lambda\mathbf{Ce}_1\mathbf{e}_1^{\mathrm{T}})_{ij}}{2c_{ij}^t}\le\frac{1}{2}O''_{c_{ij}} =-\frac{1}{2}\left[\mathbf{D}^{\mathrm{T}}\mathbf{D}+\mathbf{B}^{\mathrm{T}}\mathbf{B}+1+\lambda\mathbf{e}_1\mathbf{e}_1^{\mathrm{T}}\right]_{jj} \tag{21}$$

Therefore, to show $Q(c, c^t_{ij}) \le O_{cij}(c)$, it is equivalent to show

$$(\mathbf{CD}^{\mathrm{T}}\mathbf{D}+\mathbf{CB}^{\mathrm{T}}\mathbf{B}+\lambda\mathbf{Ce}_1\mathbf{e}_1^{\mathrm{T}})_{ij}\ge c_{ij}^t\left[\mathbf{D}^{\mathrm{T}}\mathbf{D}+\mathbf{B}^{\mathrm{T}}\mathbf{B}+\lambda\mathbf{e}_1\mathbf{e}_1^{\mathrm{T}}\right]_{jj} \tag{22}$$

Since the elements in **C**, **D**, and **B** are non-negative, we have

$$(\mathbf{CD}^{\mathrm{T}}\mathbf{D}+\mathbf{CB}^{\mathrm{T}}\mathbf{B}+\lambda\mathbf{Ce}_1\mathbf{e}_1^{\mathrm{T}})_{ij}= \sum_l c_{il}^t(\mathbf{D}^{\mathrm{T}}\mathbf{D})_{lj}+\sum_l c_{il}^t(\mathbf{B}^{\mathrm{T}}\mathbf{B})_{lj}+\lambda\sum_l c_{il}^t(\mathbf{e}_1\mathbf{e}_1^{\mathrm{T}})_{lj} \ge c_{ij}^t(\mathbf{D}^{\mathrm{T}}\mathbf{D})_{jj}+c_{ij}^t(\mathbf{B}^{\mathrm{T}}\mathbf{B})_{jj}+\lambda c_{ij}^t(\mathbf{e}_1\mathbf{e}_1^{\mathrm{T}})_{jj} \tag{23}$$

Up to here, $Q(c, c^t_{ij}) \le O_{cij}(c)$ has been proved thus (18) is an auxiliary function for O_{cij}.

Next, we will define the auxiliary functions regarding to the updating rules (13) and (16). Similarly, let O_{dij} and O_{bij} be the partial of (5) relevant to d_{ij} and b_{ij} and $O_{dij}(d'_{ij})$ and $O_{bij}(b'_{ij})$ be the partial objective values when d_{ij} and b_{ij} equal to d'_{ij} and b'_{ij}, respectively. Since the updating rules for **D** and **B** are also element wise, it is sufficient to show that O_{dij} and O_{bij} are non-decreasing according to the updating rules (13) and (16). Let the following be the auxiliary functions regarding to O_{dij} and O_{bij}:

$$Q(d,d_{ij}^t)= O_{d_{ij}}(d_{ij}^t)+O'_{d_{ij}}(d-d_{ij}^t)-\frac{(\mathbf{DC}^{\mathrm{T}}\mathbf{C}+\mathbf{D})_{ij}}{2d_{ij}^t}(d-d_{ij}^t)^2$$
$$Q(b,b_{ij}^t)= O_{b_{ij}}(b_{ij}^t)+O'_{b_{ij}}(b-b_{ij}^t)-\frac{(\mathbf{BC}^{\mathrm{T}}\mathbf{C}+\mathbf{B})_{ij}}{2b_{ij}^t}(b-b_{ij}^t)^2 \tag{24}$$

Since the proof for the above functions to be auxiliary functions for O_{dij} and O_{bij} is similar to that for O_{cij}, we don't show the proof in detail due to the space limitation.

Having obtained the auxiliary functions for O_{cij}, O_{dij}, and O_{bij}, now we can show the convergence of (5) using the updating rules (10), (13) and (16). Since (18) is an auxiliary for O_{cij}, according to (17), we have

$$c_{ij}^{t+1}=\arg\max_c Q(c,c_{ij}^t)=c_{ij}^t\frac{(\alpha\mathbf{YD}+(1-\alpha)\mathbf{AB}+\lambda\mathbf{e}_2\mathbf{e}_1^{\mathrm{T}})_{ij}}{(\mathbf{CD}^{\mathrm{T}}\mathbf{D}+\mathbf{CB}^{\mathrm{T}}\mathbf{B}+\lambda\mathbf{Ce}_1\mathbf{e}_1^{\mathrm{T}})_{ij}} \tag{25}$$

The above result is same to the updating rule (10). Since (18) is an auxiliary function, O_{cij} is non-decreasing when c_{ij} is updated according to (25) or (10). This is equivalent to say that O is non-decreasing when c_{ij} is updated according to (10) for c_{ij} is any element of **C**.

Since (24) are auxiliary functions for O_{dij} and O_{bij}, according to (17), we have

$$d_{ij}^{t+1}=\arg\max_d Q(d,d_{ij}^t)=d_{ij}^t\frac{(\alpha\mathbf{YC})_{ij}}{(\mathbf{DC}^{\mathrm{T}}\mathbf{C}+\mathbf{D})_{ij}}$$
$$b_{ij}^{t+1}=\arg\max_b Q(b,b_{ij}^t)=b_{ij}^t\frac{[(1-\alpha)\mathbf{AC}]_{ij}}{(\mathbf{BC}^{\mathrm{T}}\mathbf{C}+\mathbf{B})_{ij}} \tag{26}$$

The above results are same to the updating rules (13) and (16). Since (24) are auxiliary functions, O_{dij} and O_{bij} are non-decreasing when d_{ij} and b_{ij} are updated according to (13) and (16). This is equivalent to say that O is non-decreasing when d_{ij} and b_{ij} are updated according to (13) and (16), respectively. The above proof shows that O is non-decreasing when **C**, **D**, and **B** are iteratively updated according to (10), (13) and (16). Thus, we have

$$O(\mathbf{C}^0,\mathbf{D}^0,\mathbf{B}^0)\le O(\mathbf{C}^1,\mathbf{D}^0,\mathbf{B}^0)\le O(\mathbf{C}^1,\mathbf{D}^1,\mathbf{B}^0) \le O(\mathbf{C}^1,\mathbf{D}^1,\mathbf{B}^1)\le\cdots\le O(\mathbf{C}^{opt},\mathbf{D}^{opt},\mathbf{B}^{opt}) \tag{27}$$

where O shows a non-decreasing trend in each iteration of updating and it may finally achieve the local optima. It should be noted that updating **D** ahead of **B** or the other way around may not have any impact on the convergence of O since the iterative updating rules for **D** and **B** are mutually independent.

D. The stopping criterion

As **C**, **D** and **B** are iteratively updated, the objective value converges to the local optima asymptotically. Simultaneously, the variation of the three matrices, **C**, **D** and **B** becomes less evident as the elements in each matrix are approximate to the magnitudes which lead the objective value to local optima. Thus, we may use the following stopping criterion to terminate the optimization process and MISAGA may obtain matrices **C**, **D** and **B** that lead O to converge approximately

$$\left|\mathbf{C}^i-\mathbf{C}^{i-1}\right|_F<\tau \tag{28}$$

where $\mathbf{C}^i$ stands for the sub-graph membership matrix after the ith iteration of updating, τ represents the predefined tolerance which the Frobenius norm of the difference of **C** between two iterations should satisfy. When τ is set to be a relatively small value, MISAGA may obtain a sub-graph membership matrix **C** which is very approximate to the optimal.

E. *Summary of the algorithm*

Having obtained the updating rules for **C**, **D**, and **B** and the stopping criterion for the optimization process, now we may describe the details of MISAGA. Based on the aforementioned description, the proposed algorithm can be summarized as the pseudo codes shown in Fig. 1. As it is seen in the figure, there are not many parameters that need to be input. After the parameter of weight justification α, maximum number of iteration *max_iteration*, tolerance for improvement τ, penalty factor λ and the number of sub-graphs k are determined, MISAGA will iteratively update the strength matrices **C**, **D**, and **B** till the variation of **C** between each two iterations is less than τ or the objective function converges to the local maxima. After the optimization process is terminated, MISAGA obtains the sub-graph membership matrix **C**, which contains the optimal or approximately optimal membership between each vertex and k sub-graphs. Given **C**, MISAGA can directly identify the best sub-graph membership for each vertex in the attributed graph and in this regard, it is more efficient than other approaches, such as EGCPI [46] which requires an additional step to further search for interesting sub-graphs.

IV. Experiment and analysis

To evaluate the effectiveness of MISAGA, we performed a number of experiments using both synthetic and real world data sets. 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 performance.

A. *Experimental set-up and performance metrics*

1) *Baselines for comparison*

Compared with other approaches for sub-graph detection, MISAGA has many desirable features and it would be interesting to find out how much these features make MISAGA better. For such purpose, we have selected a number of popular algorithms for performance benchmarking. They include Affinity Propagation clustering (AP) [11], Spectral clustering (SC) [12], *k*-means clustering [20], Multi-Assignment Clustering (MAC) [21], CESNA [27], Relational topic model (RTM) [32] and ECDA [29]. One of the reasons why they are selected is that they are relatively more popular algorithms and have all been used effectively to discover sub-graphs in various network graphs. Also, they are representatives of the three categories of topology-based, attribute-based and topology-and-attribute based algorithms respectively.

```
Algorithm MISAGA
Input:          Y, A, $\alpha$, max_iteration, $\tau$, $\lambda$, k
Output:         C, D, B
Randomly initialize C, D, B;
$\mathbf{C}=\mathbf{C}./[\mathbf{C}\mathbf{e}_1\mathbf{e}_1^T]$;
for count=1: max_iteration
    Fixing D and B
    update C according to (10);
    Fixing C
    update D according to (13);
    update B according to (16);
    if ($|\mathbf{C}^i - \mathbf{C}^{i-1}|_F<\tau$)
        compute objective value according to (5);
        break;
    end if
end for
return C, D, B;
```

Fig. 1. Pseudo codes of MISAGA

For example, AP and SC are sub-graph identification algorithms that mainly consider topological structures of a graph when performing graph clustering. These two algorithms can discover interesting sub-graphs with sizes that can be very different from those methods that perform graph clustering based on modularity optimization, e.g., CNM and BGLL. For our experiments, we used the SC that makes use of the normalized cut in graph clustering. For sub-graph identification algorithm that are based on attribute values, *k*-means clustering and MAC are used as to cluster vertices based on the similarity of the attribute values associated with them.

For graph clustering algorithms that consider both graph topologies and attribute values, we used CESNA, RTM and ECDA. As most effective algorithms taking into consideration both graph topology and attribute values are model-based, we selected CESNA and RTM, which utilize generative models and topic models, respectively. While, ECDA performs its tasks using an evolutionary graph clustering algorithm. The three approaches are therefore very different even though they all take both topologies and attribute values into consideration.

For performance benchmarking, the algorithms above were not re-implemented. The source code or executables made available by the authors were used in our experiments. All experiments were conducted under the same environment which included a workstation with 4-core 3.4GHz CPU and 16GB RAM.

2) *Experimental set-up*

Other than ensuring that the algorithms we used for benchmarking purposes are tested with the original code written by the authors in the same computing environment, the parameters that are required for these algorithms to run are set 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, CESNA 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, *k*-means, MAC, and RTM, which require parameters to be 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* and the number of clusters (k) be set by the users before it can run. To find a better set of parameters, we tried SC using different sigma and k settings from 1 to 10 and from 10 to 200 respectively. The settings that give the best performance of SC are recorded and presented in our performance analysis report below.

For MISAGA, we tried different settings of the parameter, λ, that is required for the algorithm to work. λ was set in our experiments to 10^{-1}, 10^0, 10 and 10^2 and we found that

MISAGA performed well when λ was set to 10^{-1} or 10^{0}. Thus, all the experimental results of MISAGA shown in this manuscript were obtained when λ was set to these values. As for the other parameters, we set α to 0.5, maximum iterations to 300, and τ to 1e-9. As for k, it is set to be the same as the other algorithms, k-means, MAC and RTM. All the algorithms, including MISAGA, were executed 10 times in each set of data to obtain average results.

3) Data sets descriptions

For performance evaluations, we used both synthetic and real data sets 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. The data sets that we used are described below.

Ego-facebook. The *Ego-facebook* [28] data set is social network data that is constructed based on a number of sub-networks extracted from facebook.com. As such, interesting ground truth sub-graphs that represent social circles are known and can be used for evaluation. In this data set, there are 4039 vertices each of which represents a facebook user. These vertices are connected by 88234 edges that represent friendship between users and for each vertex, a total of 1283 attribute values, which represent the profile of the user are associated with the vertex that represent the user.

Caltech. This is a set of social network data which is constructed based on the friendship relationship extracted from the California Institute of Technology. The social network users in *Caltech* can be segmented into different classes based on the house affiliation according to the college's dorm system [39]. There are 769 vertices representing 769 social network users, and 16656 edges representing friendship between users. A total of 53 attribute values that represent the user profile are associated with the vertex that represent the user.

Rice. This is a set of online social network data which is constructed based on the friendship relationship among students studying at the Rice University. There are altogether 4087 vertices representing students and 184828 edges representing friendship between students represented as vertices. The student profile, which is made up of 74 attributes, is considered the set of attribute values that is associated with each vertex. The ground-truth communities of this data set have been identified based on dormitory residence [39].

Villa. This is another set of online social network data which was constructed based on the friendship relationship obtained from the University of Villanova. There are altogether 7772 vertices representing different users and 314989 edges representing friendship between users. For each vertex, a set of 140 attribute values that represent the profile of the user is made associated with the vertex. The ground truth communities for this data set have been identified.

Krogan [40]. This is a set of protein-protein interaction (PPI) network data related to Saccharomyces cerevisiae. It is constructed based on known interactions between proteins. In *Krogan*, there are 2674 vertices representing proteins, 7075 edges representing interactions between them and a set of 3064 attribute values representing GO terms [41], associated with each protein, that represents the properties and functions of each protein. Interesting sub-graphs in this set of data represent known protein complexes that can be found in a CYC2008 database [42]. In other words, the ground-truth sub-graphs of this set of PPI network is known.

DIP [43]. This is another set of PPI network data which are constructed based on the interactions between proteins. In this data set, there are 4579 vertices representing proteins, 20845 edges representing interactions between proteins and 4237 attribute values associated with each protein that represent biological properties and functions. Like the *Krogan* data set, the ground-truth sub-graphs for this *DIP* data set are also known.

Syn1k. This is a set of synthetic data which is generated based on the rule that the probability of intra-sub-graph edges is higher than that of inter-sub-graph edges and that vertices in the same interesting sub-graph are more related to each other than those that are not. For this data set, we used 1000 vertices that are divided into 4 ground truth communities, 9900 edges and 50 attribute values are made to associate with each vertex.

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

4) Performance metrics

For the purpose of performance evaluation, we used three evaluation measures, including the Normalized Mutual Information (*NMI*), the Average Accuracy (*Acc*) [44] and the Mean Jaccard Similarity (*JS*) [27] measures. All of them are widely used for evaluating the validity of detected sub-graphs or graph clusters.

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

$$NMI = \frac{\sum_{C,C^*} \Pr(C_i, C_j^*) \log \frac{\Pr(C_i, C_j^*)}{\Pr(C_i)\Pr(C_j^*)}}{\max(-\sum_i \Pr(C_i)\log\Pr(C_i), -\sum_j \Pr(C_j^*)\log\Pr(C_j^*))} \quad (29)$$

where $\Pr(C_i, C_j^*)$ denotes the probability that vertices are in both the detected sub-graph i and the ground truth sub-graph j, and $\Pr(C_i)$ denotes the probability that a vertex is found to exist in detected sub-graph, i. The *NMI* considers both the size of each discovered sub-graphs and the ground-truth sub-graphs by computing a fraction ratio between the sub-graphs that are identified and those that are available in the ground-truth database. If the *NMI* measure is high, it means that the sub-graphs detected match well with the ground-truth sub-graphs.

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

$$Acc = \sum_i \frac{|C_i|}{n_V} f\left(C_i, C^*\right) \quad (30)$$

where $|C_i|$ means the size of a detected sub-graph, and $f(.)$ stands for a particular mapping function between a detected sub-graph i and the ground truth. For our purpose, we define $f(.)$

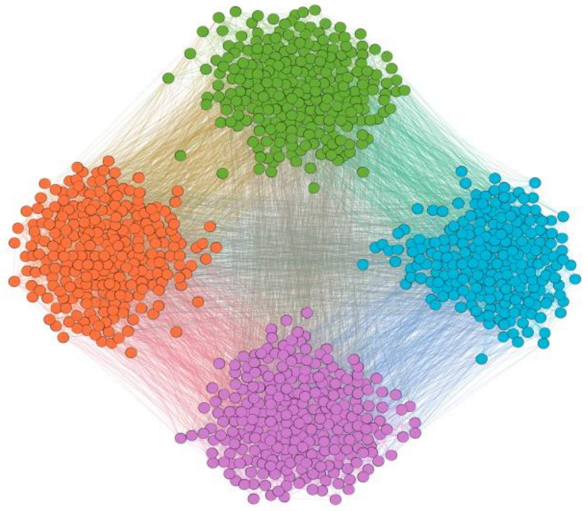

Fig. 2. Four ground truth sub-graphs of Syn1k. Vertices of the four sub-graphs are painted with different colors.

to be the maximum overlap between detected sub-graph *i* and a ground-truth sub-graph. As a result, *Acc* evaluates the weighted average of maximum overlapping between each discovered sub-graph and a ground-truth cluster. A higher value of *Acc* therefore means that each detected sub-graph has a better match with the ground truth. The higher the *Acc* is, the more effective the algorithm can be considered to be. Given its definition, *Acc* emphasizes more on the discovered sub-graphs when evaluating.

The Mean Jaccard Similarity (*JS*) measures the average degree of agreement between the detected and ground-truth sub-graphs that share the highest Jaccard Similarity with them. It is defined as

$$JS = \sum_{C_i^*} \frac{\max_{C_j \subset C} S(C_i^*, C_j)}{2|C^*|} + \sum_{C_j} \frac{\max_{C_i^* \subset C^*} S(C_i^*, C_j)}{2|C|} \tag{31}$$

where (i) $S(C_i^*, C_j)$ denotes the Jarccard Similarity between the ground-truth, C_i^*, and detected sub-graph, C_j, (ii) |C| and $|C^*|$ denote the number of discovered sub-graphs, and that of ground-truth clusters, respectively. *JS* is a harmonic mean of the bi-directional Jaccard Similarity measure computed between the discovered and the ground truth sub-graphs. It first computes the arithmetic mean of the sum of the best Jaccard Similarity between each sub-graph and one particular sub-graph in the ground-truth database. Then, it does the same between each ground-truth sub-graph and one discovered sub-graph. *JS* is then determined by aggregating the above two arithmetic means using the weights of 0.5 for each. *JS* can evaluate the quality of discovered sub-graphs by considering the proportion of overlap between them and the ground truth sub-graphs without being affected by the sizes of sub-graphs identified and those that are known to be ground-truth.

Even though they all can be considered as measures of the differences between detected and ground-truth sub-graphs, the *NMI*, *Acc* and *JS*, do not measure exactly the same aspects. In fact, what they measure can be considered as complementary. It is for this reason that all three measures are adopted as part of our evaluation criteria.

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

	Syn1k		
Approach	*NMI*	*Acc*	*JS*
AP	0.152*	0.747*	0.028*
SC	0.232*	0.528*	0.329*
k-means	0.691*	0.835*	0.209*
MAC	0.745*	0.926	0.86
CESNA	0.792*	0.845*	0.748*
RTM	0.797*	0.797*	0.717*
ECDA	0.272*	0.466*	0.493*
MISAGA	**0.995**	**0.999**	**0.998**

THE SYMBOL * MEANS THE EXPERIMENTAL PERFORMANCE OBTAINED BY MISAGA IS SIGNIFICANTLY BETTER THAN THAT OBTAINED BY A BASELINE ALGORITHM AT THE 95% CONFIDENCE LEVEL. THE STATISTICAL TEST USED IN THE EXPERIMENT IS SINGLE-SIDED *Z*-TEST.

Besides directly comparing the performance obtained by MISAGA and other algorithms in our experiments, we also carried out some statistical tests to determine whether the performance obtained by MISAGA is significantly better than that obtained by other baselines. For such purpose, we used the single-sided *z*-test to determine whether MISAGA is significantly better than other baselines at the 95% confidence level.

B. Experimental results using synthetic data

1) Performance on identifying sub-graphs

For performance evaluation, we used a set of synthetic graph data containing 1000 vertices to test the effectiveness of all different algorithms in discovering interesting sub-graphs. The community structure of the synthetic data set is shown in Fig. 2. As mentioned above, the synthetic data are generated by assuming that the probability of vertices within the same sub-graph to be connected with other vertices to be higher than that of the probability between sub-graphs. For the purpose of our experiment, the data set *Syn1k* was generated by setting the probability of intra-sub-graph connections to be 0.05 and the probability of inter-sub-graph connections to be 0.01.

The performance of MISAGA and other algorithms on the synthetic data set *Syn1k* with respect to *NMI*, *Acc* and *JS* is given in Table I. As the table shows, MISAGA performs relatively better than other algorithms. When *NMI* measure is considered, the *NMI* obtained by MISAGA is better than RTM, CESNA, MAC, *k*-means, ECDA, SC and AP by 25%, 26%, 34%, 44%, 266%, 329% and 547%, respectively. When *Acc* is considered, MISAGA outperforms MAC, CESNA, *k*-means, RTM, AP, SC and ECDA by 8%, 18%, 20%, 25%, 34%, 68%, and 114%, respectively. When comparing performance using *JS*, MISAGA is better than MAC, CESNA, RTM, ECDA, SC, *k*-means and AP by 16%, 33%, 39%, 102%, 203%, 378% and 3464%, respectively. With the *z*-test, MISAGA is found to be better than any other baselines at a 95% confidence level when their discovered sub-graphs are evaluated by *NMI*. When evaluated by *Acc* and *JS*, the performance obtained by MISAGA is significantly better than all the baselines, except

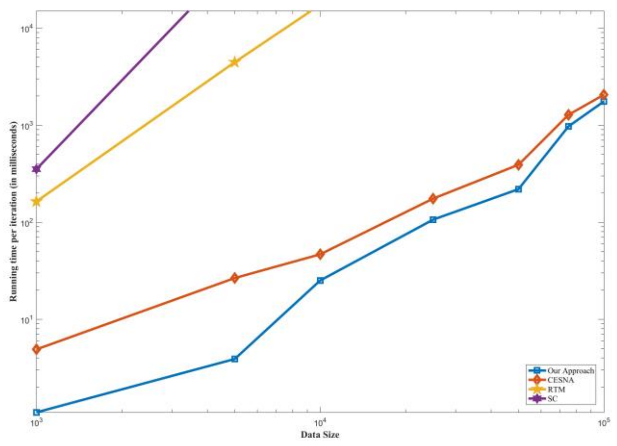

Fig. 3. Scalability comparison between MISAGA and other algorithms

with MAC. These experimental results show that MISAGA can be very effective with the discovering of interesting sub-graphs.

2) Scalability test

In order to find out how MISAGA can scale up when data set 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-sub-graph vertex connections as is with *Syn1k*. Given these generated data, the scalability of MISAGA was studied in a number of experiments involving different data sets. The results obtained were compared with those obtained with CESNA, RTM and SC. As MISAGA and these algorithms are all iterative in nature, comparison is made based on the average execution time of each iteration. The results are shown in Fig. 3.

The results show that MISAGA scales up well when compared with CESNA, RTM and SC. Even with the data sets containing as many as 100,000 vertices, MISAGA was able to 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 is required for the two algorithms to complete the sub-graph discovery tasks, it should be noted that CESNA needed at least 300 iterations whereas MISAGA converges much fewer than 300 iterations. Given this to be the case, MISAGA is more computationally efficient.

When compared with RTM and SC, the computation time used by them was much more demanding than MISAGA. When the data set size was increased to 10,000, RTM and SC was already not being able to cope. The computation time required was intolerable.

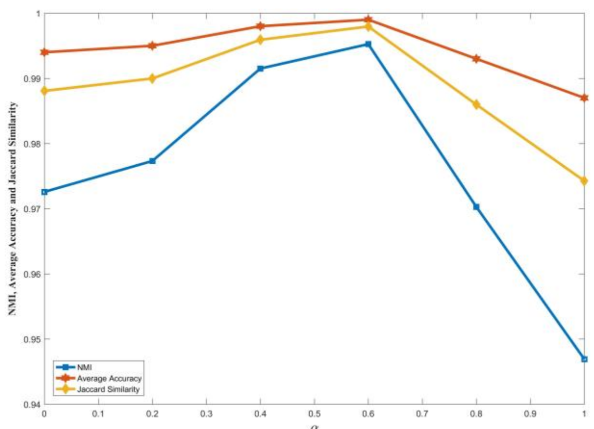

Fig. 4. Sensitivity test of MISAGA using different α

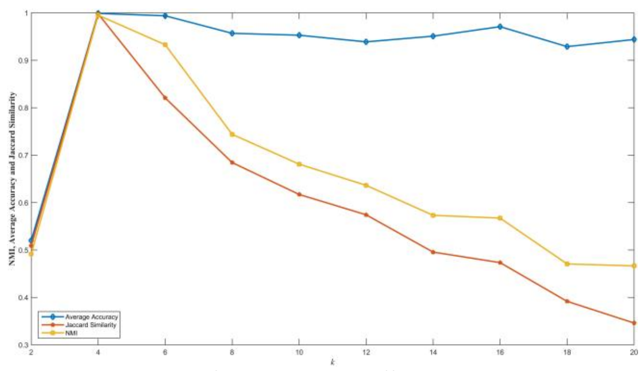

Fig. 5. Sensitivity test of MISAGA using different k

3) Sensitivity test of the parameters

As described in Section 3, for MISAGA to performs its tasks, it requires the setting of a parameter α. The parameter is used to adjust the weight between edge density and attribute association for interesting sub-graph discovery. How the parameter may affect the performance of MISAGA 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 MISAGA was used under these different settings to try to discover interesting sub-graphs. The performance was measured with *NMI*, *Acc* and *JS* and the results are shown in Fig. 4.

It is seen that when α was set to 0, which means that only the attribute values are considered in the sub-graph detection process, and when it is set to 1, which means that only the edge structures are considered, the performance of MISAGA is affected negatively. When setting α to the value between 0.4 and 0.6, MISAGA obtains very good results with most data sets. 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 MISAGA.

In addition, to investigate how the settings of k, the total number of sub-graphs to be identified, may affect the performance of MISAGA, we used it to discover interesting sub-graphs in *Syn1k* by varying k from 2 to 20, with an increment of 2. The interesting sub-graphs identified were then evaluated by *NMI*, *Acc* and *JS* and the results are shown in Fig. 5. As shown in the figure, MISAGA performs the best according to *NMI*, *Acc*, and *JS* when k is configured appropriately (the best k for *Syn1k* is 4). Compared with the results of *Acc*, the *NMI* and *JS* as obtained by MISAGA seems to be more sensitive to the settings of k. This is because *NMI* and *JS* are two evaluation metrics which consider the extent of overall matching between discovered sub-graphs and the ground truth. An inappropriate setting of k might degrade the performance of MISAGA when its performance is measured by *NMI* and *JS*. As *Acc* mainly considers the best matching between a discovered sub-graph and a ground truth subgraph, MISAGA can obtain robust results using different settings of k.

With the results measured using *NMI*, *Acc* and *JS*, we can conclude that we can use MISAGA to discover interesting

TABLE II
EXPERIMENTAL RESULTS IN SOCIAL NETWORK DATA

Data set	*Caltech*			*Ego-facebook*			*Rice*			*Villa*		
Approach	*NMI*	*Acc*	*JS*	*NMI*	*Acc*	*JS*	*NMI*	*Acc*	*JS*	*NMI*	*Acc*	*JS*
AP	0.279*	0.458	0.066*	0.58*	0.416*	0.049*	0.139*	0.257*	0.019*	0.178*	0.441*	0.057*
SC	0.305*	0.375*	0.167*	0.569*	0.447*	0.107*	0.242*	**0.455**	0.066*	0.191*	0.492*	0.032*
k-means	0.176*	0.268*	0.131*	0.385*	0.276*	0.077*	0.055*	0.174*	0.044*	0.2*	0.381*	0.042*
MAC	0.106*	0.209*	0.081*	0.37*	0.257*	0.086*	0.024*	0.138*	0.056*	0.109*	0.325*	0.045*
CESNA	0.221*	0.384*	0.203	0.399*	0.384*	0.196	0.114*	0.222*	0.109*	0.336*	0.633	**0.285**
RTM	0.277*	0.23*	0.135*	0.592*	0.39*	0.161*	0.114*	0.195*	0.065*	0.375*	0.417*	0.102*
ECDA	0.156*	0.202*	0.242	0.353*	0.234*	0.105*	0.056*	0.147*	0.053*	0.245*	0.389*	0.169*
MISAGA	**0.453**	**0.487**	**0.257**	**0.675**	**0.582**	**0.203**	**0.368**	0.429	**0.222**	**0.495**	**0.69**	0.203

sub-graphs for real world applications using different values of *k*. However, we recommend that *k* can be either manually set between 2 and $n_V/2$ or done automatically by using the techniques described in [49], [50], and [51].

C. Experimental results in real world data

1) Community detection in social networks

The social communities in a social network can be considered interesting sub-graphs in a social network graph. The identification of such social communities is important to social network analysis. For performance evaluation of MISAGA, we used four sets of real social network data, including *Ego-facebook, Caltech, Rice* and *Villa* as testing data sets. All these data sets have known ground-truth communities that have been verified in the previous work and for this reason, performance of the different algorithms can be more objectively compared.

The experimental results of *NMI*, *Acc* and *JS* obtained with these data sets are summarized in Table II. As the table shows, MISAGA performs more robustly than other algorithms. For the data set *Caltech*, MISAGA outperforms SC, AP and RTM by 49%, 62% and 64% when they are evaluated by *NMI*. When *Acc* is considered, MISAGA is better than AP, CESNA and SC by 6%, 27% and 30%, respectively. When evaluated by *JS*, MISAGA surpasses ECDA, CESNA and SC by 6%, 27% and 54%, respectively.

For the data set *Ego-facebook*, MISAGA surpasses, RTM by 14%, AP by 16% and SC by 19% in *NMI*. When *Acc* is considered, MISAGA outperforms SC by 30%, AP by 40% and RTM by 49%, respectively. When evaluated by *JS*, MISAGA outperforms CESNA, RTM and SC by 4%, 26% and 90%, respectively.

For the data set *Rice*, MISAGA outperforms SC, AP and CESNA by 52%, 165% and 223% when they are evaluated by *NMI*. MISAGA also ranks the second best in *Acc*. When evaluated by *JS*, MISAGA outperforms CESNA, SC and RTM by 104%, 236% and 242% respectively.

For the data set *Villa*, MISAGA is better than RTM, CESNA and ECDA by 32%, 47% and 102% when *NMI* is considered. MISAGA ranks the second best evaluated by *JS*, following CESNA. When *Acc* measure is considered, MISAGA is better than CESNA, SC, and AP by 9%, 40% and 56% respectively.

In evaluating the performance of the different algorithms using the *z*-test, we can see that MISAGA outperforms most algorithms in most datasets. For examples, in the case of the *NMI* in all the datasets, MISAGA is statistically significantly better. When evaluated by *Acc*, MISAGA is also significantly better than most baselines in all the datasets, except AP in *Caltech*, SC in *Rice*, and CESNA in *Villa*. When evaluated by *JS*, MISAGA is statistically significantly better than any other baselines, except CESNA in *Caltech*, *Ego-facebook*, and *Villa*. Given such results obtained by the *z*-test, it is concluded that MISAGA is significantly more robust when used to discover meaningful sub-graphs in social network graphs. The experimental results also indicate that the interesting sub-graphs detected by MISAGA are consistently better matched with the ground-truth than the other algorithms.

2) Structural modules detection in PPI networks

Structural modules in biological networks, such as protein complexes in PPI network graphs can be considered interesting sub-graphs of the larger biological network graphs.

To further test the effectiveness of MISAGA, we used two sets of PPI network data in our experiments. They included the *Krogan* and *DIP*. These data sets were chosen as the

TABLE III
EXPERIMENTAL RESULTS IN PPI NETWORK DATA

Data Set	*Krogan*			*DIP*		
Approach	*NMI*	*Acc*	*JS*	*NMI*	*Acc*	*JS*
AP	0.385*	0.187	0.168*	0.263*	**0.117**	0.138
SC	0.347*	0.129*	0.129*	0.174*	0.038*	0.038*
k-means	0.398	0.128*	0.12*	0.274	0.078*	0.086*
MAC	0.363*	0.114*	0.103*	0.3	0.062*	0.077*
CESNA	0.203*	0.025*	0.018*	0.122*	0.015*	0.011*
RTM	0.225*	0.054*	0.042*	0.157*	0.032*	0.029*
ECDA	0.416	0.142*	0.153*	0.303	0.058*	0.093*
MISAGA	**0.441**	**0.198**	**0.244**	**0.316**	0.113	**0.146**

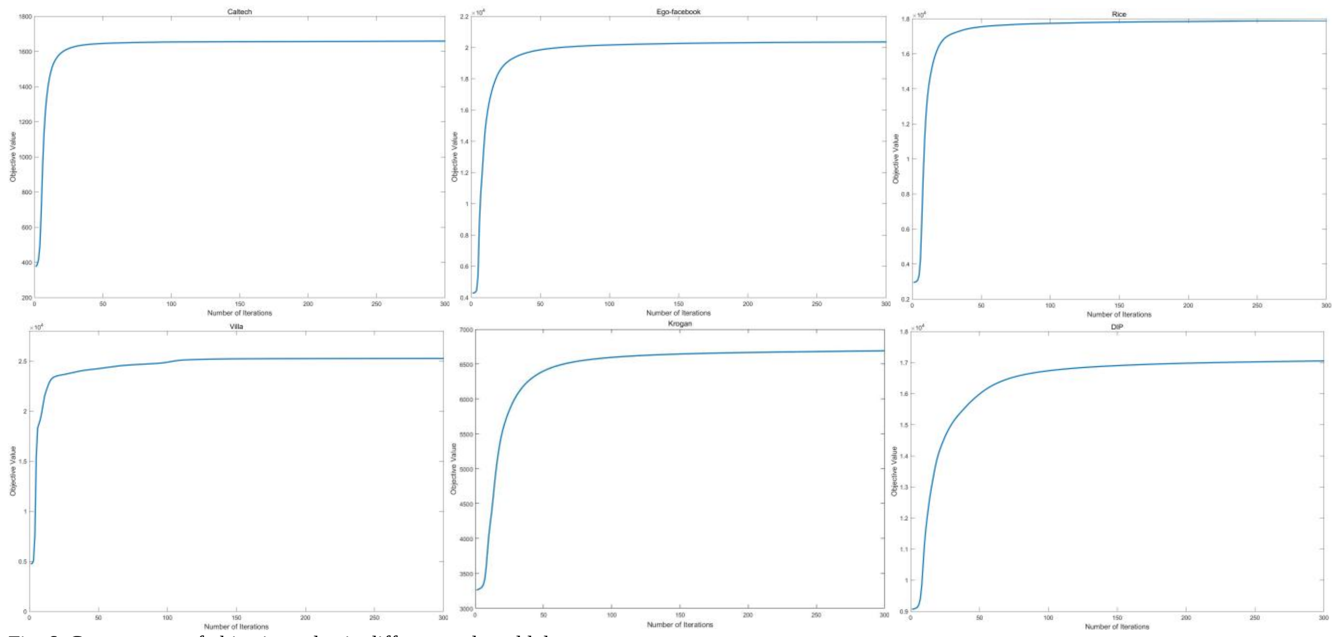

Fig. 6. Convergence of objective value in different real world data

ground-truth sub-graphs, which correspond to known protein complexes, could be found. Performance data based on *NMI*, *Acc* and *JS* were obtained from the experiments. The results obtained with these two data sets are shown in Table III.

As shown in the table, MISAGA obtains better performance than most of the other algorithms regarding of what performance measure are used. For the *Krogan* set, for example, MISAGA outperforms ECDA, *k*-means and AP by 6%, 11% and 15%, respectively, when evaluated by *NMI*. It surpasses AP, ECDA and SC by 6%, 39% and 53% when evaluated by *Acc*. When evaluated by *JS*, MISAGA outperforms AP, ECDA and SC by 45%, 59% and 89%, respectively.

For data set of *DIP*, MISAGA outperforms ECDA, MAC, and *k*-means by 4%, 5% and 15% when evaluated by *NMI*. When the detected sub-graphs are evaluated by *Acc*, MISAGA ranks the second best among all the algorithms. In the case that it is evaluated by *JS*, MISAGA performs better than other algorithms. MISAGA outperforms AP, ECDA, *k*-means by 6%, 57% and 70%, respectively. These results show that MISAGA is also very effective in identifying meaningful functional modules in protein-protein interaction networks.

According to the *z*-test, the performance of MISAGA is also statistically significantly better than most baselines in the case of the biological graphs. *NMI*, *Acc*, and *JS* obtained by MISAGA are significantly better than those obtained by most baselines, except only five cases. Given the relatively robust performance of MISAGA with social network and PPI network data, we demonstrate that MISAGA can be a useful algorithm for mining interesting sub-graphs as clusters.

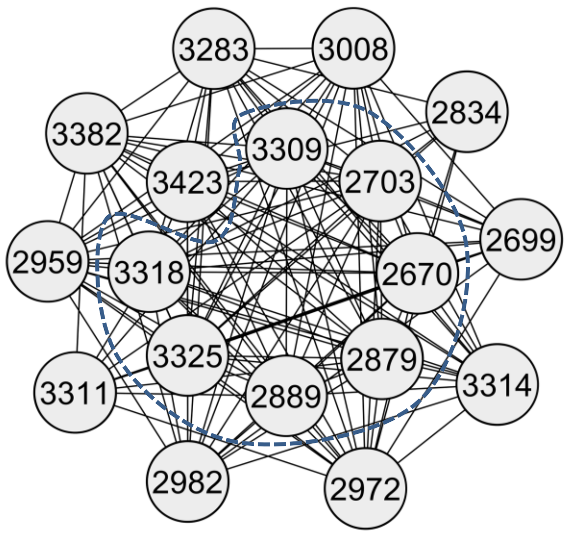

Fig. 7. The structure of a ground truth cluster in the data set of *Ego-facebook*. MISAGA identifies all the vertices successfully. The structure identified by CESNA is in the dashed circle.

D. Convergence of the objective value

To find out if MISAGA can converge in a finite number of iterations, variations of the values of the objective functions were considered in different experiments. For experimentation, we randomly selected one execution of MISAGA when each set of real data is run and the variations of the values of the objective function is recorded as shown in Fig. 6. From it, it can be seen that the objective function of MISAGA can achieve approximate convergence within 200 iterations. As the improvement of each iteration become less evident, we may take the objective values after 200 iterations as the approximately convergent ones.

E. Case study of the detected sub-graphs

Besides evaluating the detected sub-graphs with the use of different measures such as *NMI*, *Acc*, and *JS*, we also investigated into the details of some of the interesting sub-graphs detected by MISAGA. In particular, we have looked into the details from the aspects of both edge structure and attribute values associated with these sub-graphs.

TABLE IV

ASSOCIATED ATTRIBUTE VALUES IN THE DETECTED SOCIAL COMMUNITY

education;school;id;anonymized feature# 1040	education;school;id;anonymized feature 1040
education;school;id;anonymized feature 52	education;school;id;anonymized feature 52
education;school;id;anonymized feature 52	education;school;id;anonymized feature 52
education;school;id;anonymized feature 1040	education;year;id;anonymized feature 64
education;school;id;anonymized feature 1040	hometown;id;anonymized feature 935
education;year;id;anonymized feature 64	hometown;id;anonymized feature 935
education;school;id;anonymized feature 52	location;id;anonymized feature 128
location;id;anonymized feature 128	location;id;anonymized feature 128

#: THE USER PROFILE DATA ARE PROCESSED AS ANONYMIZED ATTRIBUTES TO PROTECT THE PRIVACY. IN THE TABLE, EACH ROW CONTAINS A PAIR OF SIGNIFICANTLY ASSOCIATED ATTRIBUTE VALUES. ASSOCIATED ATTRIBUTE VALUES MIGHT BE DIFFERENT FROM EACH OTHER.

For example, in the data set, *Ego-facebook*, one sub-graph identified by MISAGA completely matches with the community that is in the ground truth database. The structure of this community is shown in Fig. 7. Simultaneously, CESNA also identified this community, but the structure was not entirely identical (see the sub-graph inside the dashed circle in Fig. 7). Having checked each attribute value that is associated with each vertex in this community, we found that, even though two vertices in the sub-graph are connected, their respective attribute values are not exactly the same. Among the vertices in this sub-graph, only 15 out of over 1,200 attribute values that are associated with them. And these attribute values are not always the same. If a similarity measure is computed based on the attribute values, these vertices would probably not be considered to be in the same sub-graphs. Using MISAGA, however, we found 20 pairs of attribute values that are significantly associated with each other but not the same. For the purpose of discussion, we listed 8 pairs of them in Table IV.

Although the real user profiles are anonymized, we can infer that this social community might be related to a community of individuals that were friends of a school or university. As shown in the table, significant associations exist between different attribute-value pairs. For example, “School 64” and “Hometown 935”, “School 1040” and “Hometown 935” between the “School” and “Hometown” attributes are found to be significantly associated. Such associations are expected as students from the same school may be friends and communicate more with each other if they are also from same district location. As another example, “School 1040” and “Year 64” being significantly associated may indicate that users in this community tend to communicate with others who entered the school in the same year. In addition, MISAGA has also discovered that there are significant interesting associations between users coming from the same school.

Instead of requiring that vertices in the same sub-graph share mostly the same attribute values, MISAGA is able to discover attribute values that are not the same but are associated with each other statistically significantly. The determination of sub-graph membership based on edge structure and such attribute association is the reason why MISAGA can better identify social communities.

V. CONCLUSION

In this paper, a novel algorithm, that is called MISAGA, for mining interesting sub-graphs in attributed graphs is presented. MISAGA performs its tasks by considering both the edge structure and the attribute values that are associated with each vertex. By computing a degree of association for vertices that are statistically significantly associated with each other, MISAGA formulates and solves the problem of discovering interesting sub-graphs in an attributed graph as a constrained optimization problem. MISAGA has been tested with different sets of synthetic and real data. It is found MISGA is effective in finding optimal or near-optimal solutions. For future work, how MISAGA can be enhanced to allow different vertices to belong to different sub-graph with different degrees of freedom will be interesting. In addition, it can also be enhanced to deal with attributed graphs with structures that change with time. The discovering of the dynamics of the changes will allow prediction of future structures to be made.

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, et al., “Fast unfolding of communities in large networks,” *J. Stat. Mech.*, vol. 2008, no. 10, pp. P10008, Oct. 2008.

[7]. R. Guimera, M. Sales-Pardo, and L. A. N. Amaral, “Modularity from fluctuations in random graphs and complex networks,” *Phys. Rev. E*, vol. 70, no. 2, pp. 025101, Aug. 2004.

[8]. 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.

[9]. G. Palla, et al., “Uncovering overlapping community structure of complex networks in nature and society,” *Nature*, vol. 435, pp. 814-818, Jun. 2005.

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

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

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

[13]. 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.

[14]. L. Yang, et al., “A unified semi-supervised community detection framework using latent space graph regularization,” *IEEE Trans. Cybern.*, vol. 45, no. 11, pp. 2585-2598, Nov. 2015.

[15]. E. M. Airoldi, et al., "Mixed Membership Stochastic Blockmodels," *J. Mach. Learn. Res.*, vol. 9, pp. 1981-2014, Sep. 2008.
[16]. B. Karrer, and M. E. J. Newman, "Stochasitc blockmodels and community structure in networks," *Phys. Rev. E*, vol. 83, no. 1, pp. 016107, Jan. 2011.
[17]. C. Liu, J. Liu, and Z. Jiang, "A multiobjective evolutionary algorithm based on similarity for community detection from signed social networks," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2274-2287, Dec. 2014.
[18]. H. Chang, Z. Feng, and Z. Ren, "Community detection using dual-representation chemical reaction optimization," *IEEE Trans. Cybern.*, vol. PP, no. 99, pp. 1-14, 2016.
[19]. S. Bechikh, A. Chaabani, and L. B. Said, "An efficient chemical reaction optimization algorithm for multiobjective optimization," *IEEE Trans. Cybern.*, vol. 45, no. 10, pp. 2051-2064, Oct. 2015.
[20]. D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*, Cambridge, UK: Cambridge Univ. Press, 2003.
[21]. M. Frank, et al., "Multi-assignment clustering for boolean data," *J. Mach. Learn. Res.*, vol. 13, pp. 459-489, Feb. 2012.
[22]. 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.
[23]. Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," in *Proc. VLDB* 2009, pp. 718-729.
[24]. 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.
[25]. S. Gunnermann, et al., "Efficient mining of combined subspace and subgraph clusters in graphs with feature vectors," in *Proc. PAKDD*, 2013, pp.261-275.
[26]. Z. Xu, et al., "Gbagc: a general bayesian framework for attributed graph clustering," *ACM Trans. Knowl. Discov. Data*, vol. 9, no. 1, article 5, 2014.
[27]. 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.
[28]. J. McAuley, and J. Leskovec, "Discovering social circles in ego networks," *ACM Trans. Knowl. Discov. Data*, vol. 8, no. 1, article 4, 2014.
[29]. T. He, and K. C. C. Chan, "Evolutionary community detection in social networks," in *Proc. CEC*, 2014, pp. 1496-1503.
[30]. D. Blei, "Probabilistic topic models," *Commun. ACM*, vol. 55, no. 4, pp. 77-84, 2012.
[31]. R. Nallapati, et al., "Joint topic models for text and citations," in *Proc. 14th ACM Int. Conf. Kowl. Discov. Data Mining, 2008*, pp. 542-550.
[32]. J. Chan, and D. Blei, "Relational topic models for document networks," in *Proc. 12th Int. Conf. Artificial Intel. Stat.*, 2009, pp. 81-88.
[33]. Y. Sun, et al., "itopicmodel: information network-integrated topic modeling," in *Proc. IEEE 9th Int. Conf. Data Mining*, 2009, pp. 493-502.
[34]. T. Yang, et al., "Combining link and content for community detection: a discriminative approach," in *Proc. 15th ACM Int. Conf. Kowl. Discov. Data Mining*, 2009, pp. 927-936.
[35]. 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.
[36]. 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.
[37]. 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.
[38]. A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of Royal Statistics Society*, vol. 39, pp. 1-38, 1977.
[39]. A. L. Traud, et al., "Comparing community structure to characteristics in online collegiate social networks," *SIAM Rev.*, vol. 53, no. 3, pp. 526-543, Aug. 2011.
[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]. M. Ashburner, et al., "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics*, vol. 25, no. 1, pp. 25-29, 2000.
[42]. S. Pu, et al., "Up-to-date catalogues of yeast protein complexes," *Nucleic Acids Res.*, vol. 37, no. 3, pp. 825-831, Feb. 2009.
[43]. I. Xenarios, et al., "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.
[44]. 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.
[45]. Y. Wan, and M. Meila, "Graph Clustering: Block-models and model free results," in *Proc. Advances in Neural Information Processing Systems*, 2016, pp. 2478-2486.
[46]. T. He, and K. C. C. Chan, "Evolutionary graph clustering for protein complex identification," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. pp, no. 99, pp. 1-1, Dec. 2016.
[47]. M. Shan, S. Li, Z. Ding, and Y. Fu, "Deep linear coding for fast graph clustering," in *Proc. 24th International Joint Conference on Artificial Intelligence*, 2016, pp. 3798-3804.
[48]. X. Huang, H. Cheng, and J. X. Yu, "Dense community detection in multi-valued attributed networks," *Information sciences*, vol. 314, pp. 77-99, Sep. 2015.
[49]. U. V. Luxburg, *Clustering Stability*, Now Publishers Inc., 2010.
[50]. A. Bertoni, and G. Valentini, "Model order selection for bio-molecular data clustering," *BMC Bioinformatics*, vol. 8, no. 2, pp. S7, 2007.
[51]. G. Valentini, "Clusterv: a tool for assessing the reliability of clusters discovered in DNA microarray data," *Bioinformatics*, vol. 22, no. 3, pp. 369-370, 2006.
[52]. L. Geng, and H. J. Hamilton, "Interestingness measures for data mining: A survey," *ACM Computing Surveys*, vol. 38, no. 3, article 9, 2006.

Tiantian He received BEng degree in computer science and technology from North China University of Technology, Beijing, China in 2008 and MSc degree in information systems from The Hong Kong Polytechnic University, Hong Kong in 2012. Currently he is working towards his doctor degree in Department of Computing, The Hong Kong Polytechnic University. His research interests include graph clustering, evolutionary computation 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 200 publications in these areas, and his research is supported both by government research funding agencies and the industry. Chan serves on the editorial board of five journals and has also been serving on the program committees of numerous conferences.