A graph clustering method for community detection in complex networks

HongFang Zhou*, Jin Li, JunHuai Li, FaCun Zhang, YingAn Cui
School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China

Abstract: Information mining from complex networks by identifying communities is an important problem in a number of research fields, including the social sciences, biology, physics and medicine. First, two concepts are introduced, Attracting Degree and Recommending Degree. Second, a graph clustering method, referred to as AR-Cluster, is presented for detecting community structures in complex networks. Third, a novel collaborative similarity measure is adopted to calculate node similarities. In the AR-Cluster method, vertices are grouped together based on calculated similarity under a K-Medoids framework. Extensive experimental results on two real datasets show the effectiveness of AR-Cluster.

Keywords: Complex networks, Collaborative similarity, Graph clustering, Community detection

1 Introduction

In recent years, complex networks have been used in many domains. In these applications, networks can be modeled as graphs because of the richly latent expressive capabilities, such as scientific collaborative networks, social networks, sensor networks and the Web. The research on complex networks is important to understand the structure of the networks and the interaction of the entities in the networks. Some relevant graph clustering techniques are described in the literature [1-5].

* Corresponding author.
E-mail address: zhouhf@xaut.edu.cn (HongFang Zhou),
286374431@qq.com (Jin Li), lijunhuai@xaut.edu.cn (JunHuai Li), zfc@xaut.edu.cn (FaCun Zhang),
1503404966@qq.com (YingAn Cui)

The structure of a community is one of the most important network attributes and has attracted widespread attention. Many methods have been proposed for community structure detection, and they have been applied successfully to real complex networks [6-12]. However, the majority of methods typically consider topological structures or attribute resemblances [13-15]. The main goal of graph clustering is to discover densely connected subgraphs in a large graph, so that the two most closely related ones are considered synthetically. (1) Vertices in the same subgraph should be highly cohesive but sparsely connected to other subgraphs. (2) Homogeneous vertices are partitioned into the same group, while heterogeneous vertices should be kept in different groups. Some graph clustering techniques, such as the heuristic method [3,16], can automatically discover the number of clusters in a graph; however, some require the number of clusters as an input parameter. Moreover, graph clustering approaches either consider the topological structures or homogeneous vertex properties. Until now, few methods have synthetically considered both.

In this paper, we propose a new graph clustering method, AR-Cluster, based on Attracting and Recommending Degrees for graph clustering. AR-Cluster can accomplish the graph clustering process by identifying structural and attribute similarities at low computational cost. Compared with other relevant methods, AR-Cluster has the following characteristics. (I) A novel pair-wise structural similarity approach based on Attracting Degree and Recommending Degree is used. (II) A novel path selection strategy based on maximum recommending degree is employed. The contributions of this paper are summarized below.

1. Two concepts, Attracting Degree and Recommending Degree, are presented.

2. A new structural similarity measure based on Attracting Degree and Recommending Degree is presented.

3. A graph clustering algorithm that combines structural and attribute properties concurrently is proposed.

The rest of this paper is organized as follows. Section 2 introduces and analyzes the existing graph clustering algorithms. Section 3 addresses the concepts of Attracting Degree and Recommending Degree. We present a new collaborative similarity measure, and an iterative partitioning strategy is used for graph clustering.

Section 4 carries out extensive experiments and shows the corresponding results. Finally, section 5 concludes the paper.

2 Related works

In this section, we introduce related works on graph clustering methods for community detection in complex networks. The ultimate goal of graph clustering technology is to partition vertices in a large graph into several subgraphs.

Network topology reflects the relationship among vertices, and vertex attributes reflect the characteristics of the vertex. These two sources of data are relatively independent. If we can also consider two types of data when depicting the system sufficiently, we can avoid noise and make the clustering results more accurate. When a network is used to perform clustering analysis, it is not appropriate to use a community detection algorithm that only considers network topology or feature information. Yang et al. [17] utilized vertex feature information to pretreat the network but not vertex attribute information. Ester et al. [18] extended the k-center problem, which required a set of points constituting a connected subgraph. However, Ulitsky et al. [19] transformed the problem into a graph of community detection. They used the feature information to calculate the similarity between vertices and then superimposed these nodes' similarities onto the graph. Finally, they realized community detection. Eustace et al. [20] proposed the NRATIO algorithm based on a vertex neighborhood matrix to detect communities, but this method does not hold in non-dynamic large networks. Yoshida [21] constructed a similarity matrix using topological similarity and feature similarity; they then utilized the spectral clustering method to detect communities.

Many graph clustering or partitioning algorithms [22,23] focus on the topological properties to achieve densely internal structures. The clustering algorithm based on normalized cut [24,25] is this type of method satisfied with global optimization criteria. However, the computational cost is high. Furthermore, it involves the problem of normalized segmentation criteria, which is an NP-Hard problem. The SCAN [26] algorithm uses structural similarity to detect clusters, hubs

and outliers in the networks, and it visits every vertex only once. S-Cluster only considers the structural similarities of vertices and assigns these vertices to different clusters based on the random walk model. Cai et al. [27] also utilized the random walk model to detect communities in heterogeneous social networks. Pons and Latapy [28] proposed a method that uses a random walk model based on a certain distance L to calculate the pair-wise similarity values among nodes. Some methods based on modularity [3,29] share two characteristics and remove certain edges to split vertices. Then, they calculate similarity values again to detect community structures in social networks. Furthermore, Wu et al. [30] proposed an efficient algorithm named ImDS. ImDS is an improvement of the original density shrink algorithm for community detection. It replaces the procedure of finding and merging micro-communities by finding and merging dense pairs, which increase the accuracy and decrease the runtime.

The above-mentioned algorithms are usually used to find densely connected parts. Unfortunately, they only consider the graph's topological structure and ignore the vertex properties. In many applications, vertex properties are also important. For example, they represent people's roles in social networks. Tian et al. [31] proposed an effective OLAP-style aggregation method for large graph datasets, in which two phases are involved, the SNAP and k-SNAP operations. It can attain homogeneous attribute values within clusters. However, it neglects the intra-cluster topological structures. Sun et al. [32] proposed a clustering algorithm called RankClus that directly integrates clustering with ranking in heterogeneous information networks. It is shown to be superior in terms of clustering quality. Huang et al. [33] uses a cell-based subspace clustering approach, SCMAG, for detecting communities in multi-valued attributed networks. Random walk with restart is used to measure structural connectivity and attribute similarity. Cheng et al. [34] proposed an algorithm based on a random walk strategy, W-Cluster, which combines both structural and attribute aspects. SA-Cluster [35] also uses a unified distance measure to integrate them. Vertex closeness is measured by a neighborhood random walk model in the augmented graph, but it is unsatisfactory in terms of scalability. Gü−nnemann et al. [36] proposed a new method named GAMer to find homogeneous

object groups in a single vertex-labeled graph. It combines the paradigms of dense subgraph mining and subspace clustering to obtain sets of objects that are densely connected within the associated graph and also show high similarity regarding their attributes. Nawaz et al. [37] proposed IGC-CSM, which also combines structural and attribute aspects, and utilized the K-Medoids [38] framework for clustering. This approach is simple for similarity measures, but it is difficult to scale up for large graphs.

3 AR-Cluster algorithm

In this section, we propose a graph clustering algorithm based on Attracting Degree and Recommending Degree. It can be applied in multiple graphs.

3.1 Problem statement

An undirected, weighted and attribute graph is denoted as $G = (V, E, W, \Lambda)$, where V is the set of vertices, E is the set of undirected edges and two vertices v_i and v_j are connected with an undirected edge having weight w_{ij}. Each vertex in a graph has a set of attributes, which is denoted as $\Lambda = \{attr_1, attr_2, \cdots, attr_n\}$, and the set of attribute values for any arbitrary vertex v_i on attributes $attr_j$ is presented as $attr_j(v_i) = \{attr_1(v_i), \cdots, attr_m(v_i)\}$. $d(v_i)$ represents the degree of a vertex v_i. If two vertices v_i and v_j in a graph have a direct link, we define them as directly connected; otherwise, they are indirectly connected.

The purpose of graph clustering based on structural and attribute measures is to partition a large graph into k disjoint subgraphs, where $V = \bigcup_{i=1}^{k} v_i$ and for any $i \neq j$, $v_i \bigcap v_j = \phi$. A graph clustering method should consider the following two aspects. (I) These vertices in a subgraph should be highly cohesive and sparsely connected to other subgraphs. (II) The homogeneous vertices are partitioned into the same subgroup, while heterogeneous vertices should be kept in different subgroups.

There are two core problems in our proposed algorithm. They are the similarity measure and the clustering strategy. We discuss them in the following sections.

3.2 Attracting and Recommending Degrees

A graph typically consists of its topological structure and vertex properties. Both are of great importance, but most of the existing approaches merely consider one of them. In view of these, we propose a graph clustering algorithm called AR-Cluster. This method is based on the Attracting Degree and Recommending Degree. Some frequently used symbols are presented in Table 1.

Table 1 Commonly used symbols and their descriptions.

Symbols	Description
$v_m \leftrightarrow v_n$	A direct link between two vertices
$v_m \Theta v_n$	An indirect link between two vertices
$v_m \otimes v_n$	Two vertices are disconnected
$sim(X,Y)$	Similarity between two arbitrary set of elements
$csim(v_m,v_n)$	Collaborative similarity measure between two vertices
$d(v_m)$	Number of edges incident on vertex m
F_{obj}	Objective function which can be defined as the weighted ratio of Density and Entropy

In reality, the graph is usually used to model complex networks. The connectivity relationships for a pair of directly connected vertices represent the intimating degree between objects. The close relationships reveal that these objects are attracted to each other. In addition, a path from one source to another one is a sequence of edges that contain important transitivity information in indirectly connected relationships. Considering this critical information, we present a new clustering method.

Definition 1 (Attracting Factor): In an undirected attribute graph, structural characteristics among a pair of directly connected vertices reveal their closeness relationships. We define these intimating connections as the Attracting Factor. Let v_m and v_n be a pair of directly connected vertices. $d(v_m)$ and $d(v_n)$ are the degrees of v_m and v_n, respectively. An undirected single link between v_m and v_n contains weight w_{mn}. The Attracting Factor $f(v_m,v_n)$ between v_m and v_n is defined as Eq. (1).

$$f(v_m,v_n)=\ln\left(1+\frac{d(v_m)}{\sum_{j=1}^{d(v_m)} w_{mj}} * w_{mn}\right), v_m \leftrightarrow v_n \tag{1}$$

Definition 2 (Attracting Degree): Let v_m and v_n be a pair of directly connected vertices in an undirected attribute graph. The Attracting Degree $A(v_m,v_n)$ between

v_m and v_n is defined as Eq. (2).

$$A(v_m, v_n) = \frac{f(v_m, v_n) + f(v_n, v_m)}{2} \tag{2}$$

$f(v_m, v_n)$ is the corresponding Attracting Factor. By analyzing the Attracting Factor, we can judge which vertex is more important and infer $f(v_m, v_n) \neq f(v_n, v_m)$ between two directly connected vertices due to different weights and degrees. $f(v_m, v_n)$ does not meet the symmetry, but $A(v_m, v_n)$ satisfies because of the mutual calculations between two vertices. We can also conclude that the Attracting Factor is affected by the connecting structure of the former vertex in Eq. (2). For example, an author can establish multiple coauthor relationships [39] with many authors on a real bibliographic network, whereas the degree of importance of each of these coauthorships is not symmetrical. A paper may be published by an author and his/her students, while another paper may be published by him/her and other famous professors. Different partners have different degrees of importance, and thus the connectivity relationships in a graph are not balanced. That is, the corresponding weights on undirected edges are different. Thus, the Attracting Factors among a pair of directly connected vertices in a graph present a striking contrast according to Eq. (1). As shown in Fig. 1, it is an undirected, weighted and multi-attribute graph. Each vertex considers two attributes, job and sports hobby. The corresponding Attracting Factors and Attracting Degrees are shown in Table 2 and Table 3, respectively.

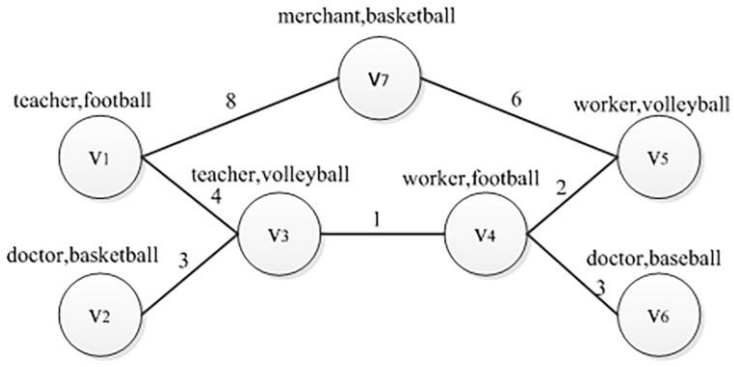


Fig. 1 An undirected, weighted and attribute graph sample.

Table 2 Attracting Factor among vertices given in Fig. 1.

Vertex / Attracting Factor / Vertex	v_1	v_2	v_3	v_4	v_5	v_6	v_7
v_1	0	0	0.5112	0	0	0	0.8422
v_2	0	0	0.6931	0	0	0	0
v_3	0.9163	0.7538	0	0.3185	0	0	0
v_4	0	0	0.4055	0	0	0.9163	0
v_5	0	0	0	0	0	0	0.8472
v_6	0	0	0	0.6931	0	0	0
v_7	0.7617	0	0	0	0.6931	0	0

Table 3 Attracting Degree among vertices given in Fig. 1.

Vertex / Attracting Degree / Vertex	v_1	v_2	v_3	v_4	v_5	v_6	v_7
v_1	0	0	0.4024	0	0	0	0.8020
v_2	0	0	0.7235	0	0	0	0
v_3	0.4020	0.7235	0	0.3620	0	0	0
v_4	0	0	0.3620	0	0.5490	0.8047	0
v_5	0	0	0	0.5490	0	0	0.7701
v_6	0	0	0	0.8047	0	0	0
v_7	0.8020	0	0	0	0.7701	0	0

Attracting Degree is used to compute relationships in a directly connected situation. However, there are also potential relationships among a pair of indirectly

connected vertices in a graph. A path from one source vertex v_m to another destination vertex v_n is a sequence of edges (v_m, v_{m+1}), (v_{m+1}, v_{m+2}), $\cdots$, (v_{n-1}, v_n) that have important guiding functions, where $e_i = (v_i, v_{i+1}) \in E$, $m \le i < n$. We refer to such passing information as recommending information. This recommending information accumulates continuously and finally reaches a destination vertex.

Definition 3 (Recommending Degree): Given a path $PT(v_m, v_{m+1}, \cdots, v_{m+i}, \cdots, v_n)$ from one source vertex v_m to one destination vertex v_n, where $i \in [0, n-m]$, the Recommending Degree $R(v_m, v_n)$ from v_m to v_n is defined as Eq. (3).

$$R(v_m, v_n) = \sum_{i=srcnode}^{desnode} f(v_i, v_{i+1}), v_m \Theta v_n \tag{3}$$

Here, $f(v_i, v_{i+1})$ is the Attracting Factor in a directly connected situation, as shown in Eq. (1).

In an attribute graph, there are multiple paths used for Recommending Degree calculations from source to destination nodes. We utilize the path that can attain the maximum recommending degree. We use an example to explain. Suppose we want to calculate the Recommending Degree between a pair of indirectly connected vertices, v_1 and v_5, as shown in Fig. 1. It is obvious that there are two paths between them, $v_1 - v_7 - v_5$ and $v_1 - v_3 - v_4 - v_5$. Through calculation, we obtain Recommending Degree v_1 and v_5; under these two paths, they are 1.5404 and 1.5288, respectively. According to the strategy of maximum recommending degree, we select the first path.

3.3 Structural and attribute similarities

In our method, the Jaccard similarity coefficient is used as a basic structural similarity measure.

$$sim(X, Y) = \frac{|X \bigcap Y|}{|X \bigcup Y|} \tag{4}$$

The Jaccard similarity coefficient is given in Eq. (4). It has been widely used to discover the relevance among objects. IGC-CSM [38] analyzes the topological structure of graphs and discovers that the important structural factor of a graph is the links that have related weights. For this reason, IGC-CSM redefines a similarity measure based on a fixed Jaccard similarity coefficient for an undirected, weighted

and multi-attribute graph. Connectivity relationships of all vertices defined by IGC-CSM have the following three forms. (a) directly connected; (b) indirectly connected; and (c) disconnected. As shown in Eq. (5) and Eq. (6), there are two concrete similarity calculation measures in situation (a) and (b), respectively. The structural similarity value is zero for disconnected vertices.

$$sim(v_m, v_n)_{connected} = \frac{w_{mn}}{\sum_{c=1}^{d(v_m)} w_{mc} + \sum_{c=1}^{d(v_n)} w_{nc} - w_{mn}}, v_m \leftrightarrow v_n \tag{5}$$

$$sim(v_m, v_n)_{indirconn} = \prod_{i=srcnode}^{desnode} sim(v_i, v_{i+1})_{connected}, v_m \Theta v_n \text{ and } v_i \in V \tag{6}$$

In Eq. (6), $sim(v_i, v_{i+1})$ is the structural similarity based on a pair of directly connected vertices. There may be a mass of paths from one source to another destination vertex. IGC-CSM [38] uses a weighted shortest path instead of all paths. As a result, the structural similarity value for a pair of indirectly connected vertices is calculated through the linear product of direct structural similarity values.

After obtaining the Attracting Degree and Recommending Degree, we can calculate the similarities among pairs of vertices. Inspired by the IGC-CSM method [38], our proposed similarity method also uses the collaborative similarity measure.

In Eq. (7), $sim(v_m, v_n)_{struct}$ presents the structural similarity between two vertices v_m and v_n. The concrete similarity calculation strategy is considered based on three types of connection. (a) A directly connected pair of vertices utilizes the Attracting Degree and a fixed Jaccard similarity coefficient measure to calculate the structural similarity value. (b) An indirectly connected pair of vertices employs the linear product of direct structural similarity values and their maximum recommending degree to estimate the similarity value. (c) The structural similarity value for a disconnected vertex is zero.

$$sim(v_m, v_n)_{struct} = \begin{cases} sim(v_m, v_n)_{connected} + A(v_m, v_n), \mathrm{v_m} \leftrightarrow v_n \\ sim(v_m, v_n)_{indirconn} + R(v_m, v_n), v_m \Theta v_n \text{ and } v_i \in V \\ 0, v_m \otimes v_n \end{cases} \tag{7}$$

In a weighted attribute graph, the similarity measure should consider the attributes of all vertices. A node can reflect multiple attribute characteristics. For instance, a vertex represents a person who has some relevant properties in social networks. When these nodes appear in different semantic environments, the

importance of their attributes is evident. Thus, we should consider attribute similarities to attain much better cohesiveness among nodes. A vertex can contain several associated attributes, and each attribute can contain multiple values. We utilize a unified number of attributes for each vertex. For example, we define two attributes for each vertex as shown in Fig. 1. Both of the two attributes have four values. The weight of each attribute is assumed to be 1. We adopt the IGC-CSM [38] method for calculating attribute similarities. The concrete formulas are given in Eq. (8) and Eq. (9), where w_{attr} represents the attribute weight.

$$sim(v_m, v_n)_{attribute} = \begin{cases} \dfrac{\sum_{i=1}^{M} common(v_m, v_n) * w_{attr_i}}{\sum_{j=1}^{M} w_{attr_j}}, v_m \leftrightarrow v_n \text{ and } v_m \otimes v_n \\ \prod_{i=srcnode}^{desnode} sim(v_i, v_{i+1})_{attribute}, v_m \Theta v_n \text{ and } v_i \in V \end{cases} \quad (8)$$

$$common(v_m, v_n) = \begin{cases} 1, if\ v_m \text{ and } v_n \text{ have some value on } i^{th} \text{ attribute} \\ 0, otherwise \end{cases} \quad (9)$$

In Eq. (8) and Eq. (9), all parameters are initialized at the beginning of our algorithm. The ultimate similarity measure combining both structural and attribute similarities is given in Eq. (10).

$$csim(v_m, v_n) = \alpha * sim(v_m, v_n)_{struct} + (1-\alpha) * sim(v_m, v_n)_{attribute} \quad (10)$$

The problem studied in this paper is clustering a large-scale graph associated with attributes based on both structural and attribute similarities. The parameter α (alpha) is a weight factor that is used to control the influence of both connectivity and semantics. This method is simple, but it requires that the parameter be given in advance. The appropriate value is to partition the graph into k clusters with cohesive intra-cluster structures and homogeneous attribute values.

3.4 Algorithm description

After calculating the final similarity values, we utilize distance values to finish the clustering process for all vertices in a graph. The distance function is defined in Eq. (11), and it is the reciprocal of the similarity value. It is expected that the distance value in close proximity is low because of the transitivity property. The smaller the distance is, the better the clustering quality is.

$$distance(v_m, v_n) = \begin{cases} \dfrac{1}{csim(v_m, v_n)}, v_m \leftrightarrow v_n \ and \ v_m \Theta v_n \\ \infty, v_m \otimes v_n \end{cases} \quad (11)$$

In the specific implementation process of our algorithm, we adopt a K-Medoids framework using distance value for partitioning the vertices. At the beginning of the algorithm, we select top k maximum degree vertices as k centroids. In each iteration, these centroids in the clustering process are updated, and the remaining vertices are assigned to the nearest centroids according to the minimum distance criterion. Our clustering algorithm AR-Cluster is described as follows.

Algorithm AR-Cluster

Input: an undirected, weighted and multi-attribute graph G, the number of clusters k, the weight factor α, the maximum of iteration $MaxIterationNumber$.

Output: k clusters $C_1, C_2, \cdots, C_k$.

1. **Initialization**

distance[v_i][v_j]=0, iteration=0, ClusterCentriod[]=0, $w_{attr_1}, w_{attr_2}, \cdots, w_{attr_M} = 1$.

2. **Similarity Calculation**

for each vertex pair v_i and v_j in V where $i \neq j$

$$csim(v_i, v_j) = \alpha * sim(v_i, v_j)_{struct} + (1-\alpha) * sim(v_i, v_j)_{attribute}$$

$$distance(v_i, v_j) = \frac{1}{csim(v_i, v_j)}$$

end for

3. **K-Medoids clustering**

Select top k maximum degree vertices in V as initial k centroids for $C_1, C_2, \cdots, C_k$,

ClusterCentroid[]=TopK(V).

while(F_{obj} is not maximized || iterations $\leq$ $MaxIterationNumber$)

for each vertex v_i in V

Cluster[i]= $min_{i,j}\{distance(i,j)\}$ for all centroids j =1...k

end for

for(each cluster j ; $j \leq k$; j ++**)**

if(the sum of distance values is minimum**) then**

update ClusterCentroid[j]

end if

end for

end while

return k clusters: C_1, C_2,…, C_k.

3.5 Computational complexity analysis

This method is suitable in the undirected, weighted and multi-attribute graphs. The AR-Cluster algorithm requires two predetermined parameters: the number of clusters and the impact factor α.

In the initial stage of the algorithm, it must confirm the number of attributes. Each attribute also ensures the number of values. The data information of the graph is stored in the main memory. Because of the clustering strategy, the distance function is the key of this algorithm. Distance values are based on the collaborative similarity measure using Eq. (11). In addition, in an undirected graph, the first parts of Eq. (7) and Eq. (8) satisfy the symmetry of values. For example, v_m and v_n are a pair of directly connected vertices, and thus $csim(v_m, v_n) = csim(v_n, v_m)$.

There are multiple paths among source-to-destination vertices. Our algorithm uses the path with the maximum recommending degree to calculate the Recommending Degree of Eq. (3) and also adopts the weighted shortest path for calculating the linear product of the direct structural similarity values of Eq. (6). This strategy avoids massive calculation. We suppose that the number of vertex set V is n. The process of path selection uses the binomial heap, so the time complexity is $O(n \log n)$. The variable $MaxIterationNumber$ is the number of iterations, and k is the number of clusters. Finally, the time complexity is $O[n * n \log n + MaxIterationNumber * n * k]$, so our method is suitable for small[1] and medium[2] scale graphs.

We adopt a K-Medoids framework for graph clustering, and this process is an iterative partitioning method. We refer to the objective function F_{obj} proposed by

[1] $< 10^4$ nodes

[2] 10^4—10^6 nodes

IGC-CSM [38]. F_{obj} is given in Eq. (12), and it is the ratio of density and entropy. The formulas of density and entropy are given in Eq. (13) and Eq. (14), respectively. In each iteration process, F_{obj} is maximized. We select the top k maximum degree vertices as initial k centroids, and the rest of the vertices are assigned to their closest centroids. In each iteration, the centroid of each cluster is replaced with a vertex that has maximum aggregated similarity compared with the remaining vertices of the same cluster. The clustering process is repeated until convergence.

$$F_{obj} = max_k \left[\frac{\alpha * Density(\{V_j\}_{j=1}^{k})}{(1-\alpha) * Entropy(\{V_j\}_{j=1}^{k})} \right] \tag{12}$$

To obtain good clustering effectiveness, the objective function F_{obj} needs to achieve the maximum value for clustering quality improvement. A high density value represents a densely structured connection, and a low entropy value demonstrates that most vertices in the same cluster have similar attributes. In Eq. (10) and Eq. (12), the weight factor α is used to balance the influences of both structure and attributes. Its value is in the interval of [0,1]. When α is 0, vertices having similar attributes become clustered in one region, irrespective of their interconnection and associated weights. However, value 1 has the opposite impact of grouping densely connected regions of vertices instead of their context. The choice of appropriate value for this parameter is critical. The appropriate value is to partition the graph into k clusters with cohesive intra-cluster structures and homogeneous attribute values. In the experimental parts, we discuss it in detail.

4 Experiments

In this section, we analyze our proposed algorithm, AR-Cluster, by performing extensive experiments. All experiments are performed on a PC with Windows XP, an i3 CPU (2.16GHz) and 1GB main memory. The programming environment is JDK 1.6.

4.1 Datasets

In our experiments, we use two real datasets—political blogs and DBLP.

Political Blogs Dataset The political blogs dataset is a network that contains 1490 web blogs on United States politics with 19090 hyperlinks between these web blogs. Each blog contains an attribute value that represents political leaning. '0' indicates liberal, and '1' indicates conservative. If there is a connection between the two blogs, the weight of the connection edge is 1. The political blogs network dataset can be downloaded from http://www-personal.umich.edu/~mejn/netdata/.

DBLP Dataset We use the subset of DBLP bibliography information data. The DBLP dataset can be downloaded from http://dblp.uni-trier.de/db/. Our selected subset contains four research areas[3] of Artificial Intelligence (AI), Information Retrieval (IR), Data Mining (DM) and DataBase (DB). We build a coauthor graph including 10,000 authors and their coauthor relationships. A coauthor relationship is interpreted as nodes and weighted edges to represent the number of combined publications in a graph. In addition, a vertex in a coauthor graph has two relevant attributes, "prolific topic" and "primary topic". The attribute "prolific topic" has three possible values. If an author has greater than or equal to 20 papers, he/she is labeled as highly prolific; and if the number of his/her papers is between 10 and 20, it is labeled as prolific; and if the number of papers is less than or equal to 10, it is labeled as low prolific. For the attribute "primary topic", we extract 100 research topics as the second attribute using a topic modeling method [39]. These research topics are based on a document collection of paper titles from these selected authors. Each extracted topic is related to the probability distribution of keywords associated with the topic. Each author is assigned one out of 100 topics as a primary topic.

4.2 Contrast methods

W-Cluster [34] This algorithm also considers the structural and attribute aspects with a random walk strategy. The weighted function between v_i and v_j is $d(v_i, v_j) = \alpha * d_S(v_i, v_j) + \beta * d_A(v_i, v_j)$ and both of the weight factors α and β are 0.5. $d_S(v_i, v_j)$ and $d_A(v_i, v_j)$ represent the structural distance and attribute distance, respectively.

[3] The detailed conference list is **DB:** SIGMOD, VLDB, PODS, ICDE, EDBT; **DM:** KDD, ICDM, SDM, PAKDD, PKDD; **IR:** SIGIR, CIKM, ECIR, WWW; **AI:** IJCAI, AAAI, UAI, NIPS.

SA-Cluster [35] This uses a unified distance measure to combine structural and attribute similarities. This method inserts a set of attribute vertices and edges to the original graph to connect vertices that share the same attribute values. In addition, it utilizes a weight adaptive strategy to learn the contribution degree of different attributes. Unfortunately, it is time-consuming.

IGC-CSM [38] The algorithm utilizes a collaborative similarity measure that combines both structural and attribute similarities. The K-Medoids framework based on an iterative partitioning method is used for graph clustering. In addition, a shortest path strategy is adopted for reducing extensive computational cost and the search space. However, it requires the number of clusters and a weight factor as input parameters.

AR-Cluster This is our proposed algorithm. It also combines the structural and attribute similarities for graph clustering. In addition, Attracting and Recommending Degrees are referred to in this algorithm.

4.3 Evaluation Measures

Considering that these compared methods adopt density and entropy for evaluating the clustering quality, we also use the two criteria. The definitions are shown as follows:

(a) Density

The ultimate goal of what we expect is to obtain the structural closeness of clusters, and thus density is an ideal choice.

$$Density(\{V_c\}_{c=1}^{k}) = \sum_{c=1}^{k} \frac{|\{(v_m, v_n) \mid v_m, v_n \in V_c, (v_m, v_n) \in E\}|}{|E|} \tag{13}$$

(b) Entropy

Entropy is used to determine the attributed relevance among vertices.

$$Entropy(\{V_c\}_{c=1}^{k}) = \sum_{c=1}^{M} \left(\frac{w_{attr_c}}{\sum_{s=1}^{M} w_{attr_s}} \sum_{i=1}^{k} \frac{|V_i|}{|V|} Entropy(attr_c, V_i) \right) \tag{14}$$

$$Entropy(attr_c, V_i) = -\sum_{n=1}^{n_c} Prcnt_{cin} \log_2 Prcnt_{cin} \tag{15}$$

where i is the number of clusters, $i=\{1,2,\cdots,k\}$, n is attribute values and n_c is the number of attribute values. $Prcnt_{cin}$ is defined as the percentage of vertices in the same cluster j that have the value $attr_{cn}$ on attribute $attr_c$.

(c) NMI (Normalized Mutual Information)

Given two partitions A and B of a network, N is defined as a confusion matrix, where the rows correspond to the "real" communities, and the columns correspond to the "found" communities. The element of N, N_{ij}, is the number of nodes in the real community i that appear in the found community j. Normalized Mutual Information $I(A,B)$ is defined as Eq. (16).

$$I(A,B)=\frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B}N_{ij}\log(\frac{N_{ij}*N}{N_{i.}*C_{.j}})}{\sum_{i=1}^{C_A}N_{i.}\log(\frac{N_{i.}}{N})+\sum_{j=1}^{C_B}N_{.j}\log(\frac{N_{.j}}{N})} \tag{16}$$

Where the number of real communities is denoted C_A and the number of found communities is denoted C_B, the sum over row i of matrix N_{ij} is denoted $N_{i.}$, and the sum over column j is denoted $N_{.j}$. If the found partitions are identical to the real communities, then $I(A,B)$ takes a value of 1. If the partition found by the algorithm is totally independent of the real partition, then $I(A,B)$ takes a value of 0.

4.4 Results

In our proposed AR-Cluster and other contrast algorithms, α is set to be 0.5 in the absence of special instructions to achieve an effective comparison. The quality of the results are evaluated using state-of-the-art evaluation measures, i.e., density and entropy. The ultimate results are shown as follows.

To achieve a tight connecting structure among vertices in the same cluster, our experimental results should have high density values.

Fig. 2 shows the density comparison for the four algorithms on Political Blogs. We set the cluster number k = 3, 5, 7 and 9. As k is increasing gradually, the density values of all four methods are declining. W-Cluster is the lowest compared with other methods when the number of clusters is the same. When k = 3 or 5, the density value of SA-Cluster is higher than that of IGC-CSM. The density value of SA-Cluster is low,

and it falls more quickly than IGC-CSM when k = 7 or 9; the structural similarity of SA-Cluster becomes even worse when k is increasing gradually. Our proposed AR-Cluster method is superior to the other three methods. The density value declines gently with the number of clusters increasing. We can conclude that AR-Cluster is much better than the other algorithms in terms of structure.

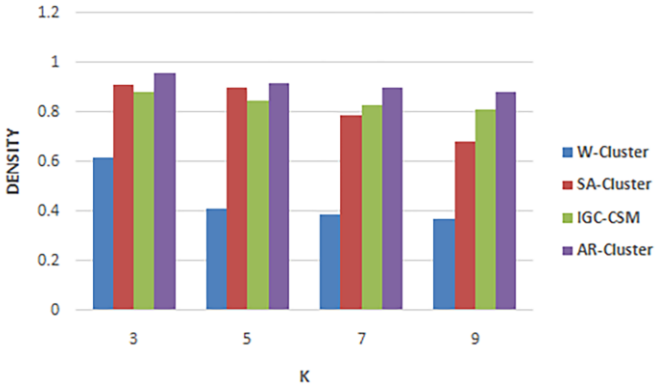


Fig. 2 Density value comparisons on Political Blogs.

Fig. 3 shows the density comparison for four algorithms on DBLP when the cluster number is set as k = 10, 30, 50 and 70. The density of W-Cluster is the lowest, and that of AR-Cluster is the highest. The density values of AR-Cluster and IGC-CSM are almost the same when k = 10 or 30. However, when k is over 30, the density value of IGC-CSM becomes lower than that of AR-Cluster. The density value of SA-Cluster stands in between. It decreases sharply when k increases from 10 to 30. The density value is the lowest when k = 30. When k = 50 or 70, SA-Cluster is rising slightly compared with k=30, but it remains low.

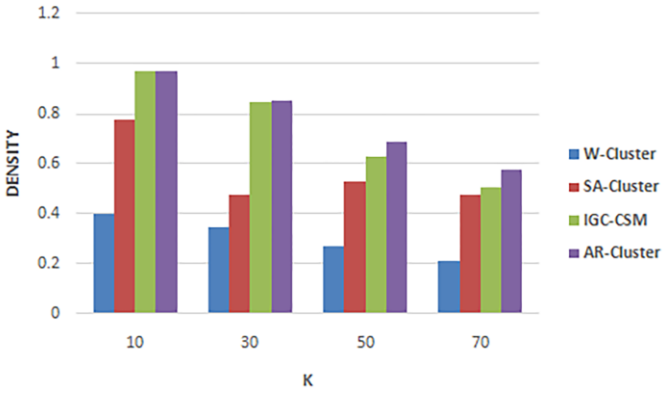


Fig. 3 Density value comparisons on DBLP.

Entropy is used to estimate the attribute relevancies among vertices. Low entropy values ensure that most of the vertices in the same cluster have similar attribute

values.

Fig. 4 shows the entropy comparison for four methods on Political Blogs for cases where cluster number k = 3, 5, 7 and 9. The entropy values of AR-Cluster and IGC-CSM are almost close. They remain below 0.1 and are steady when k is increasing. When k increases from 7 to 9, IGC-CSM has a slightly higher entropy than that of AR-Cluster. We can infer that IGC-CSM and AR-Cluster strictly enforce attribute similarity. In terms of the entropy of SA-Cluster, it is always steady below 0.1 when k = 3, 5 or 7, but the entropy undergoes a sharp rise when the value of k increases from 7 to 9. In addition, the entropy of W-Cluster is rising rapidly when k = 3 or 5. Combining the density in Fig. 2 and the entropy of W-Cluster, we can infer that the distance function compromises structural similarity and attribute similarity.

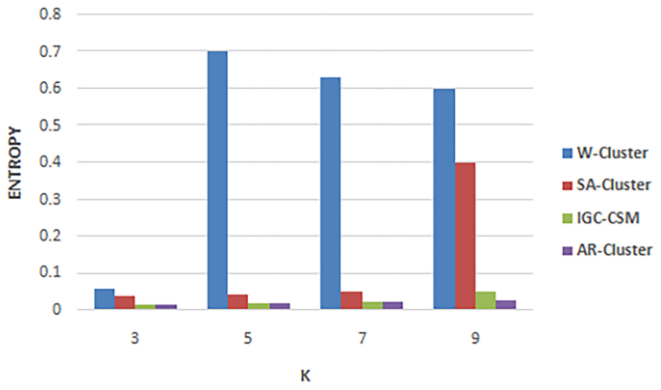


Fig. 4 Entropy value comparisons on Political Blogs.

Fig. 5 shows the entropy values of four algorithms on DBLP when we set the cluster number k = 10, 30, 50 and 70. In all four algorithms, the entropy of IGC-CSM is highest and remains around 3.5 to 4. W-Cluster exhibits an extremely low entropy of around 0 to 0.5, which is better than those of the other three methods. When k = 30, the entropy value of SA-Cluster falls rapidly and shows the lowest value. When the k value increases from 50 to 70, the entropy of AR-Cluster is declining while that of SA-Cluster is rising; moreover, its value exceeds that of AR-Cluster. Compared with SA-Cluster, AR-Cluster partitions a graph into several clusters, where each cluster contains nodes with much better attribute similarity.

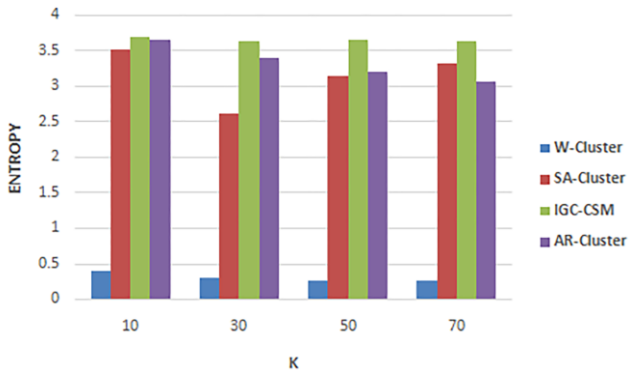
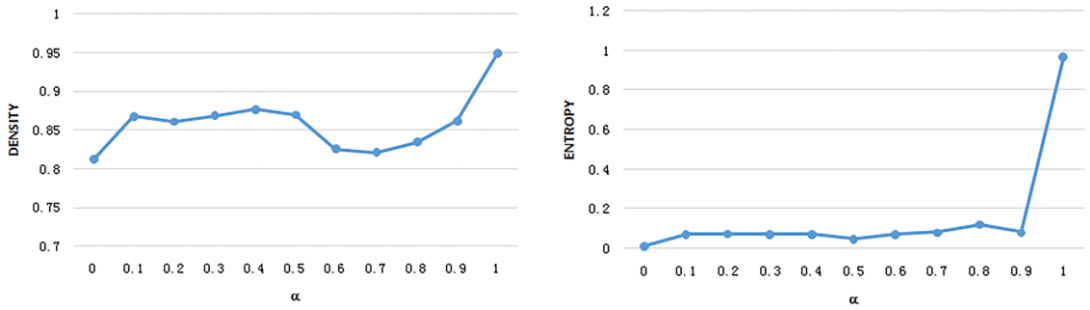


Fig. 5 Entropy value comparisons on DBLP.

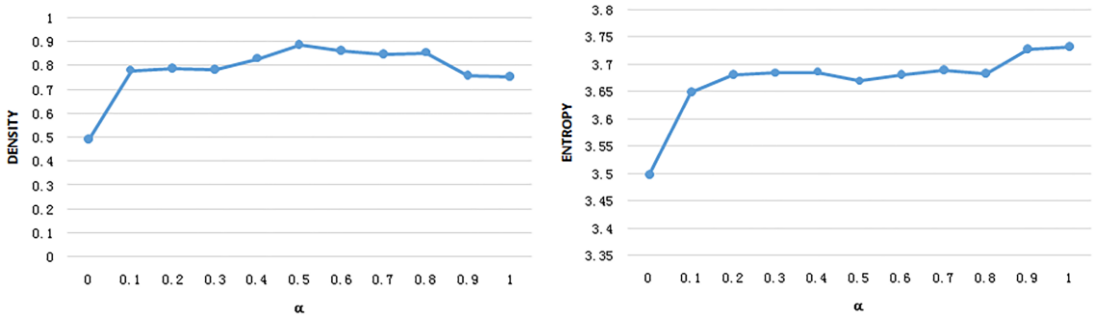
Fig. 6 (a) and (b) show density and entropy values for Political Blogs with different α ($\alpha \in [0,1]$) values for AR-Cluster. Here, we set the cluster number k = 15 and take the average value of the five experimental results. As shown in Fig. 6(a), the density value declines rapidly when α is between 0.5 and 0.7. Thus, we can conclude that α =0.5 is the start drop point. As shown in Fig. 6(b), the entropy value is always steady and low when α is between 0 and 0.9. However, both of these values are sharply rising when α is greater than 0.9.



(a) density (b) entropy

Fig. 6 Impact of α on Political Blogs.

Fig. 7 (a) and (b) show density and entropy values for DBLP with different α ($\alpha \in [0,1]$) values for AR-Cluster. We set the cluster number k as 25. We take the average value of the five experimental results. As shown in Fig. 7(a), the density value reaches the maximum when α is 0.5. It drops quickly when α is between 0.8 and 0.9. As shown in Fig. 7(b), the entropy value remains around 3.65 to 3.7 when α is between 0.2 and 0.8. It slightly declines with α =0.5. When α increases from 0.8 to 1, the density value declines, while the entropy value rises. Therefore, we can infer that the structural and attribute similarities can be helpful in obtaining the best results with α =0.5.



(a) density (b) entropy

Fig. 7 Impact of α on DBLP.

We also used NMI (Normalized Mutual Information) to measure the proposed algorithm. In Fig. 8, we compare the AR-Cluster algorithm with three reported algorithms as applied to two real datasets. The experimental results show that our proposed algorithm outperforms the other three algorithms on two real datasets. In the case of the Political Blogs dataset, we set the cluster number k as 3. The NMI value of AR-Cluster is the highest, so AR-Cluster can correctly partition the network. For the DBLP, we set the cluster number k as 10. AR-Cluster achieves the maximum value among the four algorithms.

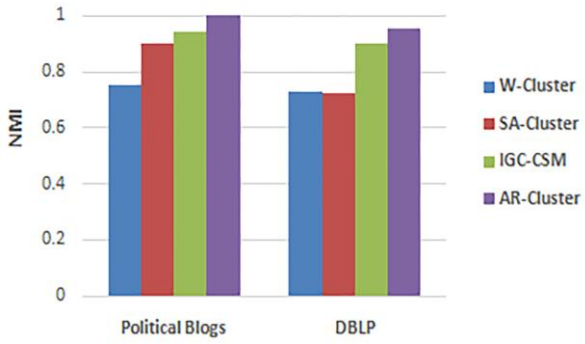


Fig. 8 NMI comparisons on two real datasets.

5 Conclusions

In this paper, we propose an effective graph clustering algorithm called AR-Cluster for community detection in complex networks. The proposed method adopts two concepts, Attracting Degree and Recommending Degree, to strengthen the structural similarities among vertices. In addition, the path with the maximum

recommending degree is taken for the calculating pairs of indirectly connected relationships. The experimental results show that the approach performs well compared to the other three methods. However, it is difficult to use for large graphs[4]. In the future, we will discuss it in detail.

Acknowledgments

This research was supported by the National Science Foundation of China under Grant 61402363, and the Education Department of Shaanxi Province Key Laboratory Project under Grant 15JS079, and the Ministry of Education of Shaanxi Province Research Project under Grant No. 14JK1545, and the Xi'an Science Program Project under Grant CXY1509(7), and the Beilin district of Xi'an Science and Technology Project under Grant GX1625.

References

[1] P. Drineas, A. Frieze, R. Kannan, S. Vempala, V. Vinay, Clustering large graphs via the singular value decomposition, Machine Learning 56(1)(2004) 9-33.

[2] G.W. Flake, R.E. Tarjan, K. Tsioutsiouliklis, Graph clustering and minimum cut trees, Internet Mathematics 1(4)(2003) 385-408.

[3] X. Huang, W. Lai, Clustering graphs for visualization via node similarities, Journal of Visual Languages & Computing 17(3)(2006) 225-253.

[4] M.E.J. Newman, Detecting community structure in networks, The European Physics Journal B - Condensed Matter and Complex Systems 38(2004) 321-330.

[5] H. Zanghi, C. Ambroise, V. Miele, Fast online graph clustering via Erdös-Rényi mixture, Pattern Recognition 41(12)(2008) 3592-3599.

[6] H. Zhang, T. Ma, G.B. Huang, Z. Wang, Robust global exponential synchronization of uncertain chaotic delayed neural networks via dual-stage impulsive control, IEEE Transaction on Systems Man & Cybernetics Part B Cybernetics 40(3)(2010) 831-844.

[7] F. Wei, W.N. Qian, C. Wang, A.Y. Zhou, Detecting overlapping community structures in networks, World Wide Web 12(2)(2009) 235-261.

[8] Y.J. Liu, Y.Q. Zheng, Adaptive robust fuzzy control for a class of uncertain chaotic systems, Nonlinear Dynamics 57(3)(2009) 431-439.

[9] X. Liu, J.Y.-L. Forrest, Q. Luo, D.Y. Yi, Detecting community structure using biased random merging, Physica A: Statistical Mechanics and its Applications 391(4)(2012) 1797-1810.

[10] L.L. Cui, H.G. Zhang, B. Chen, Q.L. Zhang, Asymptotic tracking control scheme for

[4] 10^6—10^9 nodes

mechanical systems with external disturbances and friction, Neurocomputing 73(7-9)(2010) 1293-1302.

[11] H.W. Shen, X.Q. Cheng, K. Cai, M.B. Hu, Detect overlapping and hierarchical community structure in networks, Physica A: Statistical Mechanics and its Applications 388(8)(2009) 1706-1712.

[12] Y.J. Liu, S.C. Tong, D. Wang, T.S. Li, C.L.P. Chen, Adaptive neural output feedback controller design with reduced-order observer for a class of uncertain nonlinear SISO systems, IEEE Transactions on Neural Networks 22(8)(2011) 1328-1334.

[13] Y.Y. Ahn, S. Han, H. Kwak, S. Moon, H. Jeong, Analysis of topological characteristics of huge online social networking services, in: Proceedings of the 16th International Conference on World Wide Web, WWW'07, ACM, New York, NY, USA, 2007, pp. 835-844.

[14] J. Leskovec, K.J. Lang, A. Dasgupta, M.W. Mahoney, Statistical properties of community structure in large social and information networks, in: Proceedings of the 17th International Conference on World Wide Web, WWW'08, ACM, New York, NY, USA, 2008, pp. 695-704.

[15] H.F. Zhou, J. Guo, Y.H. Wang, A feature selection approach based on term distributions, SpringerPlus 5(1)(2016) 1-14.

[16] M.E.J. Newman, Fast algorithm for detecting community structure in networks, Physical Review E 69(6)(2004) 066133.

[17] S.Q. Yang, B. Wu, H.Y. Long, B. Wang, Community detection in large-scale social networks, in: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on web mining and social network analysis, ACM, 2007, pp. 16-25.

[18] M. Ester, R. Ge, B.J. Gao, Z. Hu, B. Ben-Moshe, Joint cluster analysis of attribute data and relationship data: the connected k-center problem, in: Proceedings of Siam International Conference on Data Mining, SDM'06, 2(2)(2006) 90-98.

[19] I. Ulitsky, R. Shamir, Identification of functional modules using network topology and high-throughput data, BMC systems biology 1(1)(2007) 1-8.

[20] J. Eustace, X.Y. Wang, Y.Z. Cui, Overlapping community detection using neighborhood ratio matrix, Physica A: Statistical Mechanics and its Applications 421(2015) 510-521.

[21] T. Yoshida, Toward finding hidden communities based on user profile, Journal of Intelligent Information Systems 40(2)(2013) 380-387.

[22] Z. Liu, J.X. Yu, Y. Ke, X. Lin, L. Chen, Spotting significant changing subgraphs in evolving graphs, in: Proceedings of the 8th IEEE International Conference on Data Mining, ICDM'08, Pisa, Italy, 2008, pp. 917-922.

[23] J. Sun, Y. Xie, H. Zhang, C. Faloutsos, Less is more: Sparse graph mining with compact matrix decomposition, Statistical Analysis and Data Mining 1(1)(2008) 6-22.

[24] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 22(8)(2000) 888-905.

[25] H.F. Zhou, X.H. Zhao, X. Wang, An effective ensemble pruning algorithm based on frequent patterns, Knowledge-Based Systems 56(3)(2014) 79-85.

[26] X. Xu, N. Yuruk, Z. Feng, T.A.J. Schweiger, Scan: a structural clustering algorithm for networks, in: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD'07), ACM, New York, NY, USA, 2007, pp. 824-833.

[27] D. Cai, Z. Shao, X. He, X. Yan, J. Han, Mining hidden community in heterogeneous social

networks, in: Proceedings of Workshop on Link Discovery: Issues, Approaches and Applications, LinkKDD'05, Chicago, IL, 2005, pp. 58-65.

[28] P. Pons, M. Latapy, Computing communities in large networks using random walks, Journal of Graph Algorithms and Applications 10(2)(2006) 191-218.

[29] M.E.J. Newman, M. Girvan, Finding and evaluating community structure in networks, Physical Review E 69(2)(2004) 026113.

[30] J.S. Wu, Y.T. Hou, Y. Jiao, Y. Li, X.X. Li, L.C. Jiao, Density shrinking algorithm for community detection with path based similarity, Physica A: Statistical Mechanics and its Applications 433(2015) 218-228.

[31] Y.Y. Tian, R.A. Hankins, J.M. Patel, Efficient aggregation for graph summarization, in: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, SIGMOD'08, ACM, New York, NY, USA, 2009, pp. 567-580.

[32] Y. Sun, J. Han, P. Zhao, Z. Yin, H. Cheng, T. Wu, RankClus: Integrating clustering with ranking for heterogeneous information network analysis, in: Proceedings of the 12th International Conference on Extending Database Technology, EDBT'09, Saint Petersburg, Russia, 2009, pp. 565-576.

[33] X. Huang, H. Cheng, J.X. Yu, Dense community detection in multi-valued attributed networks, Information Sciences 314(2015) 77-99.

[34] H. Cheng, Y. Zhou, J.X. Yu, Clustering large attributed graphs: A balance between structural and attribute similarities, Acm Transactions on Knowledge Discovery from Data 5(2)(2011) 190-205.

[35] Y. Zhou, H. Cheng, J.X. Yu, Graph clustering based on structural/attribute similarities, in: Proceedings of the VLDB Endowment, VLDB'09, ACM, Lyon, France, 2(1)(2009) 718-729.

[36] S. Günnemann, I. Färber, B. Boden, T. Seidl, GAMer: a synthesis of subspace clustering and dense subgraph mining, Knowledge & Information Systems 40(2)(2014) 243-278.

[37] W. Nawaz, K. Khan, S.Y. Lee, Intra graph clustering using collaborative similarity measure, Distributed and Parallel Databases, 33(4)(2015) 583-603.

[38] L. Kaufman, P.J. Rousseeuw, Clusteing by means of medoids, Statistical Data Analysis based on the L1 Norm (1987) 405-416.

[39] Y. Sun, R. Barber, M. Gupta, J. Han, C.C. Aggarwal, Co-Author relationship prediction in heterogeneous bibliographic networks, in: Proceedings of the 2011 International Conference on Advances in Social Networks Analysis and Mining, 2011, pp. 121-128.