Computing
DOI 10.1007/s00607-016-0526-5

Weighted clustering of attributed multi-graphs

Andreas Papadopoulos[1] · **George Pallis**[1] ·
Marios D. Dikaiakos[1]

Received: 2 April 2016 / Accepted: 17 November 2016

Abstract An information network modeled as an attributed multi-graph contains objects described by heterogeneous attributes and connected by multiple types of edges. In this paper we study the problem of identifying groups of related objects, namely clusters, in an attributed multi-graph. It is a challenging task since a good balance between the structural and attribute properties of the objects must be achieved, while each edge-type and each attribute contains different information and is of different importance to the clustering task. We propose a unified distance measure for attributed multi-graphs which is the first to consider simultaneously the individual importance of each object property, i.e. attribute and edge-type, as well as the balance between the sets of attributes and edges. Based on this, we design an iterative parallelizable algorithm for CLustering Attributed Multi-graPhs called CLAMP, which automatically balances the structural and attribute properties of the vertices, and clusters the network such that objects in the same cluster are characterized by similar attributes and connections. Extensive experimentation on synthetic and real-world datasets demonstrates the superiority of the proposed approach over several state-of-the-art clustering methods.

Keywords Clustering · Information networks · Attributed multi-graphs

✉ Andreas Papadopoulos
andpapad@cs.ucy.ac.cy

George Pallis
gpallis@cs.ucy.ac.cy

Marios D. Dikaiakos
mdd@cs.ucy.ac.cy

[1] Department of Computer Science, University of Cyprus, P.O. Box 20537, 1678 Nicosia, Cyprus

Published online: 01 December 2016

Mathematics Subject Classification 05C22 · 05C40 · 05C78 · 68W10 · 68W15 · 62H30 · 91C20

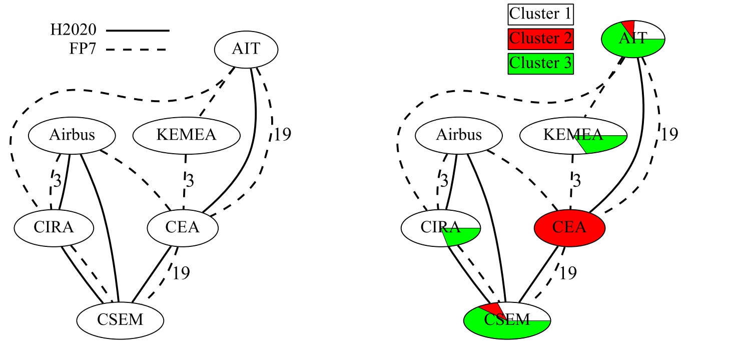
1 Introduction

Real world information networks can be effectively modeled by *attributed multi-graphs* where: (a) every object is represented by a vertex characterized by some heterogeneous (of different type) attributes to describe its properties; and (b) two objects may be connected by multiple edges each of which describes a different relationship.

The determination of the underlying clusters of such ubiquitous graphs has many interesting applications in diverse areas such as marketing, social networks, telecommunications and biology. For instance, clustering a collaboration network is practically useful in targeted advertisements, partnership recommendations, outliers detection e.t.c. [4,28]. Figure 1 presents an exemplary attributed multi-graph modeling a col-

Short Name	**Name**	**FP7 projects**	**H2020 projects**	**Country**
Airbus	Airbus SAS	14	1	France
AIT	Austrian Institute of Technology	137	9	Austria
KEMEA	Center for Security Studies	32	0	Greece
CIRA	Centro Italiano Ricerche Aerospaziali	44	1	Italy
CEA	Commissariat à l'energie atomique et aux energies alternatives	611	39	France
CSEM	Swiss Center for Electronics and Microtechnology	96	7	Switzerland

(a) Organization Attributes



(b) Organizations Collaboration Network. Weights denote the number of collaborations (unlabeled edges have weight 1).

(c) Fuzzy Clustering of the Network

Fig. 1 A sample of a real-world collaboration network modeled as attributed multi-graph. Each vertex represents an organization characterized by the attributes shown in **a**, and **b** shows the organizations collaboration network. A weighted edge indicates the number of collaborations. **c** A fuzzy clustering of the organizations into three clusters (*white*, *red*, *green*). A vertex may belong to multiple clusters according to a probability represented by a *circle* share (color figure online)

laboration network. A vertex is characterized by some heterogeneous (numerical and categorical) attributes (Fig. 1a) and represents an organization that participated in projects funded by European Union. Collaborations on FP7 and H2020 projects are modeled by weighted edges of different type in Fig. 1b. In order to provide reliable, evidence-based recommendations, we have to cluster the network by exploiting the information from both the heterogeneous organizations' attributes and the multiple collaborations.

Recently, there has been a lot of research in the area of attributed graph clustering [6] aiming to combine structural and attribute properties and consequently improve clustering quality [8]. However, these methods [1,7,18,19,31,32] cannot be directly applied to an attributed multi-graph such as the network of Fig. 1, because they either ignore the multiple types of edges and/or deal with only one attribute type. To overcome this, an attributed multi-graph is projected to an attributed graph with homogeneous attributes. Still, a projection results in information loss and consequently limit clustering accuracy. For example, by discretizing the attribute 'number of H2020 projects' to two bins we cannot distinguish if an organization participated in 1, 7 or 9 projects. Similarly, by transforming the multi-graph to a graph, i.e. by summing the weights of multiple edges, we discard collaborations type.

Moreover, majority of existing methods [7,31,32] detect densely connected components while identification of clusters of objects having similar connectivity may be of even greater importance [1,18]. For instance, AIT and CSEM share common partners (high similar connectivity) and close attributes. Since these two organizations have not collaborated in the past we can recommend a new collaboration. However, they are not a densely connected component and thus would not be clustered together. Specifically, methods such as SA-Cluster [7] and BAGC [31] seek to identify the clusters: {Airbus, CIRA, CSEM} and {AIT, KEMEA, CEA} which are densely connected components. On the other hand, PICS [1] and HASCOP [18] that optimize similar connectivity ignore the heterogeneity of the object attributes, and consequently on this example they identify one 'giant' cluster and six singleton clusters respectively. Still, these clusterings do not provide much insights, especially for recommending new collaborations.

Another important property of real world information networks is that objects are usually not completely well separated, i.e. an organization may be related-similar to organizations in multiple clusters. However, many of existing methods such as CAMIR [19], PICS [1], SA-Cluster [7] and BAGC [31] perform hard clustering instead of allowing multiple memberships, i.e. fuzzy clustering that identifies overlapping clusters in which an object belongs with a membership probability.[1] Figure 1c demonstrates a fuzzy clustering where each vertex slice represents the object's probability belonging to the respective cluster. For example, we observe that KEMEA is highly related to Airbus and CIRA, because of cluster 1; and it is also related to AIT and CSEM, because of cluster 3. Thus, we can recommend some of these organizations as partners to KEMEA by leveraging the membership probabilities.

[1] Similarly, overlapping clustering assigns an object to multiple clusters with binary memberships [32]. Though, membership probabilities provide more information, i.e. importance of an object in a cluster [14].

Moreover, object properties, i.e. attributes and edge-types, contain different information and some may be irrelevant to the clustering task. For instance, to cluster the collaboration network of Fig. 1 the attributes ‘number of FP7’ and ‘number of H2020’ projects are important, while attributes ‘number of employees’ and ‘year of establishment’ may introduce noise. Similarly, the sets of attributes and edges may not agree resulting in different clustering when used independently. One way to capture this challenging aspect is to assign proper weights to each object property (attribute and edge-type), as well as to the entire set of attributes and edges [7]. By doing so each object property is considered differently in the clustering and a balance between structural and attribute properties is achieved. However, tuning such weights is a difficult task requiring a priori knowledge or costly preprocessing of the information network under study.

In this paper we study the problem of **fuzzy clustering weighted directed attributed multi-graphs with heterogeneous attributes**. Our contributions are summarized as follows:

- We present a new unified distance measure for attributed multi-graphs which is the first to consider simultaneously the individual importance of the vertex properties and the balance between the sets of attributes and edges, by assigning different weight to each of them. Based on the presented unified distance measure, we propose a new algorithm for CLustering Attributed Multi-graPhs called CLAMP. To achieve a good balance between the structural and attribute properties of the vertices, CLAMP computes iteratively their importance-weight during the clustering process by adopting the gradient descent technique. By doing so the intermediate clustering results and the identified importance of the vertex properties mutually enhance each other until convergence. Thus computed weights efficiently capture the underlying structure of the network while costly preprocessing and a priori knowledge are not required. Identified clusters are characterized by attribute homogeneity and similar connectivity with respect to the identified importance of the vertex properties. To the best of our knowledge, this is the first work to address the problem of fuzzy clustering weighted directed attributed multi-graphs with heterogeneous attributes.
- The proposed algorithm is *highly parallelizable* and can exploit properly the computational power of modern many- and multi-core architectures so as to scale to large datasets and keep runtime low. We present the implementation of CLAMP in the MapReduce model, and perform extensive evaluation with multithreaded implementation showing that increasing computational power gradually decreases the runtime of proposed approach.
- Our extensive experimental evaluation on synthetic datasets and a diverse collection of real world information networks (bibliography items, research and innovation projects funded by European Union) against the state-of-the-art attributed graph clustering approaches: (a) confirms that the use of weighting scheme improves results quality; and (b) demonstrates the efficiency and effectiveness of the proposed approach in terms of similar connectivity and attribute homogeneity. Moreover, to the best of our knowledge, this is the first work to study

clustering results on the network of organizations participated in projects funded by European Union.[2]

2 Related work

Many methods have been proposed in the context of graphs and multi-graphs clustering [22] aiming to overcome the limitations of traditional data point clustering methods, such as k-means and FCM [5], which ignore the structural part of a graph and consequently achieve low quality results in terms of structural evaluation measures [6,7,27,30,31,34]. However, graph clustering methods ignore vertex attributes and cannot be directly applied to attributed graphs or multi-graphs.

Recently, attributed graph clustering has received much attention [2,6]. The representative approaches are based on unified distance functions [7,34] or model definitions [27,30–32]. They aim to cluster the network using various optimization techniques, such as gradient descent [7,32], EM algorithm [27] and spectral clustering [10,15]. Also, some approaches initially project the attributed graph to a single weighted graph, where weights represent a combination of attribute and structural similarities, and then apply a clustering algorithm for weighted graphs [20,25]. However, these methods differ from ours for at least one of the following aspects: (a) they assume equal importance of structural and attribute properties of the vertices; (b) they ignore the existence of multiple edge types; (c) they deal with only one type of attributes; or (d) they aim to identify either community outliers [16] or strongly connected components [2].

The concept of similar connectivity has been studied on unattributed unweighted graphs [26,29]. SCAN [29] exploits the neighborhood of vertices to partition the network such that vertices sharing many neighbors are grouped into the same cluster. Although SCAN optimizes similar connectivity, it is sensible to a threshold parameter: the minimum number of common neighbors. To overcome the issue of selecting the threshold parameter of SCAN, gSkeletonClu [26], which additionally aims to find hubs and outliers, has been proposed. Although both approaches are useful, they do not apply to attributed multi-graphs.

Currently, PICS [1] and HASCOP [18] are the only approaches that identify clusters with similar connectivity in attributed graphs. Both PICS and HASCOP follow a hierarchical clustering approach. They both built a hierarchy of clusters by either splitting or merging clusters at each iteration. Specifically, PICS is a divisive distance based approach that clusters an unweighted attributed graph (only one edge type) with categorical attributes. It uses the encoding cost, based on the Minimum Description Length (MDL) lossless compression [21], as distance function in order to split the clusters. Its goal is to minimize the encoding cost of the final clustering. On the other hand, HASCOP is an agglomerative-like algorithm that uses a heuristic distance function to identify which vertices or clusters to merge at each iteration. Although, HASCOP considers the different importance of edge types and attributes it applies only to attributed multi-graphs with numerical attributes.

[2] This dataset is available online at EU Open Data Portal—http://open-data.europa.eu.

Weighting mechanisms to reflect the different importance of various properties and improve results quality have been encapsulated in both traditional data clustering [12] and attributed graph clustering methods [7,18]. Following the way paved by traditional data clustering techniques (which cannot be applied to attributed multi-graphs), SA-Cluster [34] and its extensions SA-Cluster-Opt and Inc-Cluster [7] use a weighting mechanism so as to assign different weight to each attribute. The key idea is to build an attribute augmented graph, equal to the initial graph enriched with new vertices each of which represents an attribute value. An edge from a graph vertex to an attribute vertex is added on the condition that the graph and attribute vertices have the same attribute value. The weight of the new edge depends on the importance that each attribute has, and it is updated at each iteration based on the current clustering and a scoring mechanism. Under the same concept, GenClus [27] assigns different weights to edge-types. GenClus also updates the edge-type weights at each iteration according to computed scores based on the current clustering. Although these methods consider the different importance of attributes or edge-types, they allow only one edge between two vertices. Also, their weighting mechanisms are tightly connected to the clustering process and cannot be easily generalized and adopted by other approaches. CAMIR [19] goes a step further by assigning different weight to each attribute and edge-type before it applies spectral clustering. Specifically, CAMIR computes the attribute and edge-type weights by iteratively computing the 'agreement' among the vertex attributes and the edge-types. Two vertex properties, attribute or edge-type, 'agree' if they assign vertices the same cluster labels when they are used individually. The vertex property with the highest agreement is assigned the highest weight-importance, while the property with the lowest agreement gets the lowest weight. Nevertheless, assigned weights may not completely agree with the underlying partitioning of the graph since they are not updated during the clustering process. Weighting schemas have been also adopted to balance the sets of attributes and edges [6]. However, these methods either do not apply to attributed multi-graphs, do not concern simultaneously the different importance of structural-attribute properties and the different importance of the sets of edges-attributes, or they do not automatically compute the weights.

3 Problem formulation

An **information network** can be modeled as a weighted attributed multi-graph where: $V = \{v_i : 1 \leq i \leq \mathscr{V}\}$ is the set of vertices; $T = \{t_i : 1 \leq i \leq \mathscr{T}\}$ is the set of edge types; $E_t = \{(u, v, t) : u, v \in V, t \in T\}$ is the set of edges of type t; $E = \bigcup_{t=1}^{\mathscr{T}}(E_t)$ is the set of all edges; $w_t : E_t \rightarrow (0, 1]$ returns the weight of the edge from u to v of type t; $A = \{\alpha_i : 1 \leq i \leq \mathscr{A}\}$ is the set of all numerical and categorical attributes; each vertex is characterized by $\mathscr{A}$ attribute values given by the functions $a_\alpha : V \rightarrow D_\alpha$ where D_α is the domain of attribute α.

In this paper we focus on clustering an attributed multi-graph. In the resulted clustering, denoted as $C = \{\mathscr{C}_k : \mathscr{C}_k \subseteq V, \cup\mathscr{C}_k = V\}$, vertices in the same cluster should have the maximum similarity based on both their attributes and outgoing edges, with respect to the importance of each particular edge type and attribute.

Formally, given an attributed multi-graph and a number of clusters K ($K \geq 1$), our goals are to:

- compute the importance of each edge type t, denoted as W_t, subject to $\sum_{\forall t} W_t = 1$; and the importance of each attribute α, denoted as W_α, subject to $\sum_{\forall \alpha} W_\alpha = 1$.
- calculate the weighting factors W_{links} and W_{attr} in order to balance the sets of attributes and edges.
- calculate a clustering denoted by a $\mathscr{V} \times K$ matrix Θ where $\Theta_{i,k}$ is the probability of vertex v_i belonging to cluster $\mathscr{C}_k$, subject to $\sum_{k=1}^{K} \Theta_{i,k} = 1$ for all vertices.

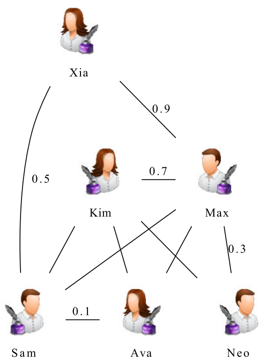
Note that, we will not study the general problem of how to determine the best number of clusters (K), which has attracted significant attention and has been covered in a large number of studies [11,24]. There are several ways to handle the selection of K such as to impose hard constraints on clusters quality and/or make use of various criteria such as the Calinski–Harabasz and the Krzanowski–Lai indices.

In the above problem the main challenges which we discuss in the following sections are: (a) a unified distance measure that considers both the structural and attribute properties of the vertices as well as the different importance of each attribute and edge-type; (b) a mechanism that automatically identifies the weights and consequently achieves a good balance between structural and attribute properties; and (c) an algorithm that based on the weighting mechanism and the unified distance measure clusters an attributed multi-graph by efficiently combining the structural and attribute properties of the vertices.

4 The CLAMP clustering model

In this section we present a new unified distance measure for attributed multi-graphs and CLAMP (CLustering Attributed Multi-graPh) algorithm. The proposed unified distance measure for attributed multi-graphs considers the individual importance of the vertex properties and balances the sets of attributes and edges by assigning different weight to each of them. In order to achieve high quality clustering of the network we develop CLAMP (CLustering Attributed Multi-graPh). CLAMP is a highly parallelizable algorithm that adopts the gradient descent technique in order to optimize the weights and cluster an attributed multi-graph. Specifically, it iteratively computes the importance-weight of the vertex properties, and consequently balances the sets of attributes and edges, during the clustering process. By doing so the intermediate clustering results and the identified importance of the vertex properties mutually enhance each other until convergence, thus the computed weights efficiently capture the underlying structure of the network. Also, in order to avoid dominance of some vertex properties denoted by high weights diversity, negative entropy regularization is applied to the weights. The goal of CLAMP is to assign each vertex: (a) to its closest cluster with the highest probability; and (b) to its furthest cluster with the lowest probability, with respect to the weights.

The remaining of this section is organized as follows. Section 4.1 presents the proposed unified distance measure for attributed multi-graphs, followed by the formalization of the clustering objective function (Sect. 4.2). Section 4.3 describes the optimization process and Sect. 4.4 discusses the parallel implementation of CLAMP in the MapReduce model.



(a) Example graph. The weight of unlabeled edges is 1.

	Kim	*Max*	*Sam*	*Xia*	*Neo*	*Ava*
Kim	1.0	0.7	1.0	0	1.0	1.0
Max		1.0	1.0	0.9	0.3	1.0
Sam			1.0	0.5	0	0.1
Xia				1.0	0	0
Neo					1.0	0
Ava						1.0

(b) Adjacency matrix

	Kim	*Max*	*Sam*	*Xia*	*Neo*	*Ava*
Kim	0	1.48	2.15	4.29	2.16	1.90
Max		0	1.15	1.85	3.88	1.80
Sam			0	1.52	2.75	1.87
Xia				0	3.61	3.17
Neo					0	2.50
Ava						0

(c) Similar connectivity matrix

Fig. 2 Similar connectivity

4.1 Distance measures

Similar connectivity is a measure that represents how dissimilar two vertices are based on their outgoing edges. We formally define Type-Similar Connectivity below.

Definition 1 (*Type-similar connectivity*) The type-similar connectivity of two vertices u, v for edge type t, denoted as $SC_t(u, v)$, is given by:

$$SC_t(u, v) = \sum_{i=1}^{\mathscr{V}} (w_t(u, v_i) - w_t(v, v_i))^2 \tag{1}$$

where

$$w_t(u, v) = \begin{cases} w_t(u, v) & \text{if } (u, v, t) \in E \\ 1 & \text{if } u = v \\ 0 & \text{else} \end{cases} \tag{2}$$

Figure 2 presents a toy co-authorship graph.[3,4] Specifically, Fig. 2b depicts the upper part of the adjacency matrix of the toy graph of Fig. 2a, c presents the similar connectivity computed by Eq. (1). For instance, similar connectivity of authors Kim and Max is computed as follows:

$$\begin{aligned} SC(Kim, Max) &= [w(Kim, Kim) - w(Max, Kim)]^2 + [w(Kim, Max) - w(Max, Max)]^2 \\ &\quad + [w(Kim, Sam) - w(Max, Sam)]^2 + [w(Kim, Xia) - w(Max, Xia)]^2 \\ &\quad + [w(Kim, Neo) - w(Max, Neo)]^2 + [w(Kim, Ava) - w(Max, Ava)]^2 \\ &= (1 - 0.7)^2 + (0.7 - 1)^2 + (1 - 1)^2 + (0 - 0.9)^2 + (1 - 0.3)^2 \\ &\quad + (1 - 1)^2 = 1.48 \end{aligned}$$

[3] Edge weights have been scaled to [0, 1].

[4] Type-similar Connectivity can be calculated on directed graphs as well.

We see that $SC(Kim, Max) = 1.48$ while $SC(Xia, Neo) = 3.61$. Hence, as it is also seen from Fig. 2a, authors $\{Kim, Max\}$ are more similar than authors $\{Xia, Neo\}$. The lowest similar connectivity between Sam and Max is justified by the fact that they are connected to each other and they have three out of four common neighbors (only Neo is not a common neighbor). Thus, low *Similar Connectivity* among two vertices denotes that they share common neighbors and should be grouped together.

Total similar connectivity measures how dissimilar two vertices are based on all their outgoing edges. We calculate the Total-Similar Connectivity of two vertices as follows:

$$SC(u, v) = \frac{1}{\mathcal{V}} \cdot \sum_{\forall t} W_t \cdot SC_t(u, v), \sum_{\forall t} W_t = 1 \tag{3}$$

where W_t is the importance of edge type t.

Attribute distance of two vertices is computed using a distance measure suitable for both numerical and categorical attributes [13]. Specifically, the attribute distance of vertices u and v, denoted as $AD(u, v)$, is given by:

$$AD(u, v) = \sum_{\forall \alpha} W_\alpha \cdot \delta_\alpha(u, v), \sum_{\forall \alpha} W_\alpha = 1 \tag{4}$$

where $\delta_\alpha(u, v)$ is the attribute distance of vertices u and v for attribute α. Function δ_α depends only on the type of attribute α. For numerical attributes scaled to [0, 1] we use the function: $\delta_\alpha(u, v) = (a_\alpha(u) - a_\alpha(v))^2$, and for categorical attributes we use the Kronecker's delta function:[5]

$$\delta_\alpha(u, v) = \begin{cases} 0 & \text{if } a_\alpha(u) = a_\alpha(v) \\ 1 & \text{else} \end{cases} \tag{5}$$

Unified distance measure To compute the total unified distance between two vertices, we balance their structural and attribute properties by using appropriate weighting factors. That is, we combine their similar connectivity and attribute distances under a unified distance measure as follows:

$$d(u, v) = W_{attr} \cdot AD(u, v) + W_{links} \cdot SC(u, v) \tag{6}$$

where W_{attr} and W_{links} are weighting factors that represent the importance of the sets of attributes and edges respectively. These weighting factors are fine-tuned by CLAMP based on the network properties and the intermediate clustering results during the clustering process. We defer more detailed discussion of the weighting mechanism to Sect. 4.3.

4.2 Clustering model

Recall that our goal is to find the optimal solution that assigns each vertex to: (a) its closest cluster with the highest probability; and (b) its furthest cluster with the

[5] Other distance functions such as Minkowski or Semantic could be adopted as well.

lowest probability, which ideally should be zero (usually it is a very small value). Intuitively, the optimal solution minimizes the weighted distance between all possible pairs of vertices and clusters. Thus, a naive approach would be to adopt traditional fuzzy clustering. That is, to find the optimal clustering that minimizes the function:

$$\sum_{i=1}^{\mathscr{V}}\sum_{k=1}^{K}(\Theta_{i,k})^f \cdot d(v_i, \mathscr{C}_k) \tag{7}$$

where $d(v_i, \mathscr{C}_k)$ is the distance of vertex v_i to cluster $\mathscr{C}_k$; and f is the so-called fuzzifier, a free parameter that takes real values in the range $(1, +\infty)$ and determines by how much clusters overlap [5,14]. The fuzzifier is necessary for getting fuzzy clusters, but its optimal value depends on the dataset and can be defined only through experimentation [5].

However, the above objective is not suitable for the problem we study because using the unified distance measure of Eq. (6) it expands to weighted sum of individual terms corresponding to attribute and structural distances. Hence, to minimize it we must assign weight 1 to the lowest term, i.e. $W_{attr} = 1$, and consequently zero weight to the highest term. The same holds for the attribute and edge-type weights, i.e. $W_{\alpha 1} = 1$ and $W_{\alpha i} = 0\,\forall i \neq 1$, if attribute $\alpha 1$ yields the lowest sum of distances. Withal, the above objective is minimized if: (a) we ignore either structural or attribute properties of the vertices; and (b) we consider only the edge type or the attribute that yields the minimum distance among all possible pairs of vertices and clusters. To address these issues we perform negative entropy regularization [17] on all the weights, so as to 'force' them to be close to each other. Specifically, we perform regularization on weighting factors W_{links} and W_{attr} to limit weights diversity and make it impossible for attributes to dominate edge types and vice versa. Similarly, we regularize attribute and edge-type weights to prohibit the dominance of a specific attribute and edge type.

In this manner, we formulate the problem of fuzzy clustering an attributed multi-graph as the identification of the optimal clustering (membership probabilities) and weights that minimize the following objective function:

$$\begin{aligned}&\sum_{i=1}^{\mathscr{V}}\sum_{k=1}^{K}(\Theta_{i,k})^f \cdot d(v_i, \mathscr{C}_k) + \lambda \cdot [W_{links} \log(W_{links})\\&\quad + W_{attr}\log(W_{attr}) + \sum_{\forall t} W_t \log(W_t) + \sum_{\forall \alpha} W_\alpha \log(W_\alpha)]\end{aligned} \tag{8}$$

subject to

$$\begin{cases} \sum_{k=1}^{K}\Theta_{i,k} = 1, & \forall v_i \in V \\ W_{attr} + W_{links} = 1, & W_{attr} > 0, W_{links} > 0 \\ \sum_{\forall t} W_t = 1, & W_t > 0 \\ \sum_{\forall \alpha} W_\alpha = 1, & W_\alpha > 0 \end{cases} \tag{9}$$

where:

- $\mathscr{C}_k$ is the representation of cluster k. Each cluster is characterized by $\mathscr{A}$ attribute values and connects to vertices by weighted edges of $\mathscr{T}$ types.[6] The attribute

[6] Hence, $\mathscr{C}_k$ is a valid parameter to Eqs. (1)–(6).

values of a cluster and the weights of its outgoing edges are computed based on its members, as we present in Sect. 4.3 below.

- $d(v_i, \mathscr{C}_k)$ is the unified distance of vertex v_i and cluster $\mathscr{C}_k$ (Eq. (6)).
- $\lambda > 0$ is the regularization parameter that controls a solution penalty according to weights entropy. High entropy equals to high weight diversity. Thus, the more the weights deviate the higher the regularization term (entropy) becomes, and the solution is penalized accordingly depending on the value of λ. The λ parameter needs to be tuned empirically or using cross validation techniques [33].

4.3 Objective function optimization

Finding the global optimum of Eq. (8) is computationally difficult (NP-hard). To tackle the high complexity of the problem, we adopt the technique of gradient descent [9]. Gradient descent requires minimum computations at each iteration, converges quickly to a local optimum, does not impose new parameters into the model, and is being widely used by many clustering algorithms, i.e. k-means.[7] According to gradient descent we iteratively optimize either the membership probabilities, the weights or the cluster representations while considering the other parameters fixed. Therefore, we make the following propositions that suggest how to update the parameters at each iteration so as to reach a minimum of the CLAMP objective function.

4.3.1 Cluster representations

Proposition 1 (Cluster outgoing edges) *If memberships and weights are fixed then the objective function is minimized when the weight of the edge from cluster $\mathscr{C}_j$ to a vertex u of type t is given by:*

$$w_t(\mathscr{C}_k, u) = \frac{\sum_{i=1}^{\mathscr{V}} (\Theta_{i,k})^f \cdot w_t(v_i, u)}{\sum_{i=1}^{\mathscr{V}} (\Theta_{i,k})^f} \tag{10}$$

In order to calculate the attribute values of a cluster we have two cases: (a) categorical attributes; and (b) numerical attributes.

Proposition 2 (Cluster categorical attributes) *If membership probabilities and weights are fixed then for categorical attributes the objective function is minimized when:*

$$a_\alpha(\mathscr{C}_k) = \arg\max_{y \in D_\alpha} \sum_{i=1}^{\mathscr{V}} \Theta_{i,k} | a_\alpha(v_i) = y \tag{11}$$

Equation (11) represents the weighted mode of the cluster ([13]). That is, a categorical attribute of a cluster takes the attribute value that characterizes most of its members, according to the membership probabilities.

[7] Also, it is suitable for our problem since Eq. (8) is differentiable. Alternatively, optimization techniques such as simulated annealing and Newton's optimization method could be adopted. However, these techniques may impose new parameters to the model, i.e. temperature parameter, or require expensive computations at each iteration, i.e. second order derivatives, while they do not guarantee better results.

Proposition 3 (Cluster numerical attributes) *If membership probabilities and weights are fixed then for numerical attributes the objective function is minimized when:*

$$a_\alpha(\mathscr{C}_k) = \frac{\sum_{i=1}^{\mathscr{V}} \left((\Theta_{i,k})^f \cdot a_\alpha(v_i)\right)}{\sum_{i=1}^{\mathscr{V}} (\Theta_{i,k})^f} \tag{12}$$

Intuitively, Propositions 1–3 update the outgoing edges and the attribute values of a cluster according to the properties of its members with respect to the membership probabilities.

4.3.2 Clustering—membership probabilities

Proposition 4 (Optimize membership probabilities) *If clusters and weights are fixed, the objective function is minimized when:*

$$\Theta_{i,k} = \left[\sum_{j=1}^{K} \left(\frac{d(v_i, \mathscr{C}_k)}{d(v_i, \mathscr{C}_j)}\right)^{\frac{1}{f-1}}\right]^{-1} \tag{13}$$

where $d(v_i, \mathscr{C}_k)$ is given by Eq. (6).

Equation (13) confirms that $\Theta_{i,k}$ must be high for low $d(\mathscr{C}_k, v_i)$ and vice versa.

4.3.3 Importance of vertex properties

Proposition 5 (Optimize attribute and edge-type weights) *If membership probabilities and clusters are fixed then the objective function is minimized when:*

$$W_t = \frac{\exp\left[\frac{S_t \cdot \ln(2)}{-\lambda}\right]}{\sum_{i=1}^{\mathscr{T}} \exp\left[\frac{S_i \cdot \ln(2)}{-\lambda}\right]}, \quad W_\alpha = \frac{\exp\left[\frac{AD_\alpha \cdot \ln(2)}{-\lambda}\right]}{\sum_{i=1}^{\mathscr{A}} \exp\left[\frac{AD_i \cdot \ln(2)}{-\lambda}\right]} \tag{14}$$

where

$$S_t = W_{links} \cdot \sum_{i=1}^{\mathscr{V}} \sum_{k=1}^{K} (\Theta_{i,k})^f \cdot SC_t(v_i, \mathscr{C}_k) \tag{15}$$

$$AD_\alpha = W_{attr} \cdot \sum_{i=1}^{\mathscr{V}} \sum_{k=1}^{K} (\Theta_{i,k})^f \cdot \delta_\alpha\,(v_i, \mathscr{C}_k) \tag{16}$$

Intuitively, Proposition 5 suggests that we assign a ‘score’ to each edge-type and attribute according to Eqs. (15) and (16) respectively, based on the individual contribution of vertices to each edge-type and attribute, i.e.

$$S_{t,i} = \sum_{k=1}^{K} (\Theta_{i,k})^f \cdot SC_t(v_i, \mathscr{C}_k) \tag{17}$$

is the *contribution of vertex v_i* to edge-type t. Then, edge-type and attribute weights are computed accordingly using Eq. (14).

Proposition 6 (Optimize global weighting factors) *If membership probabilities, clusters and attribute and edge-type weights are fixed then the objective function is minimized when:*

$$W_{links} = \frac{\exp\left[\frac{S_{links} \cdot \ln(2)}{-\lambda}\right]}{\mathscr{W}}, \quad W_{attr} = \frac{\exp\left[\frac{S_{attr} \cdot \ln(2)}{-\lambda}\right]}{\mathscr{W}} \tag{18}$$

where

$$S_{links} = \frac{1}{W_{links}} \cdot \sum\nolimits_{\forall t} W_t \cdot S_t, \quad S_{attr} = \frac{1}{W_{attr}} \cdot \sum\nolimits_{\forall \alpha} W_\alpha \cdot AD_\alpha \tag{19}$$

$$\mathscr{W} = \exp\left[\frac{S_{attr} \cdot \ln(2)}{-\lambda}\right] + \exp\left[\frac{S_{links} \cdot \ln(2)}{-\lambda}\right] \tag{20}$$

Similarly to Proposition 5, Eq. (19) assign ‘scores’ to the set of edges and attributes according to the current clustering and the contribution of each vertex. The updated weights of structural and attribute properties are given by Eq. (18).

We note that according to Propositions 5, 6 and Eq. (6) vertex property weights and global weighting factors are interrelated. That is, an edge-type importance is the product of the global edges weight and the weight of the edge-type. The same holds for attributes. However, if we combine global and individual weights to reduce model parameters regularization of both global and individual weights will not be feasible and the set of attributes or edges may dominate. Moreover, weights combination should be done by modifying the proposed unified distance measure and consequently the objective function. Although such model sounds much simpler it will consider of the same type both vertex attributes and connections. For instance, the attribute ‘gender’ and the edge-type ‘friends’ would be incorrectly considered of the same type. That is because property weights will be computed using the same formulas derived from a simplified unified distance measure that does not consider simultaneously the individual importance of the vertex properties and the sets of attributes and edges. Overall, a simplified model would neither capture the different type of information encoded in an attributed multigraph nor balance properly the vertex structural and attribute properties.

4.4 Implementation of CLAMP algorithm in MapReduce model

In this section we present the parallel implementation of CLAMP algorithm in the MapReduce model. The implementation in MapReduce confirms that CLAMP can be executed on modern cloud many- and multi-core architectures so as to scale to large

Algorithm 1 CLAMP - Attributed Multi-graphs Clustering Algorithm

```
Input: Attributed multi-graph $G$, number of clusters $K$, convergence delta $\delta$
Output: Clustering $\Theta$, Weights
Initialize iteration number: $r \leftarrow 0$
Select randomly $K$ vertices as initial clusters
Initialize weights using Eq. (21)
while true do
    MAP 1
    Input: Clusters $\mathscr{C}$, Weights, Set of vertices $S \subset V$
    Output: <$k$, [$i$, $\Theta_{i,k}$]> $\forall k$
    for all $v_i \in S$ do
        for all $\mathscr{C}_k \in \mathscr{C}$ do
            Compute membership of vertex $v_i$ to cluster $\mathscr{C}_k$ using Eq. (13)
            Output: <$k$, [$i$, $\Theta_{i,k}$]>
        end for
    end for
    REDUCE 1
    Input: <$k$, list of [$i$, $\Theta_{i,k}$]>
    Output: <$k$, [$\mathscr{C}_k$]>
    Compute cluster properties by Eqs. (10)-(12)
    if $\|\mathscr{C}_k^r - \mathscr{C}_k^{r-1}\| \le \delta$ then
        Increase counter of converged clusters
    end if

    if all clusters converged then
        return
    end if
    MAP 2
    Input: Clusters $\mathscr{C}$, Weights, Set of vertices $S \subset V$, Vector $\Theta_i$ $\forall v_i \in S$
    Output: <{$t|\alpha|T|A$}, $v_i$ contribution> $\forall v_i \in S$
    for all $v_i \in S$ do
        for all $t \in T$ do
            Compute contribution of $v_i$ to $t$
            Output <$t$, $v_i$ contribution>
        end for
        for all $\alpha \in A$ do
            Compute contribution of $v_i$ to $\alpha$
            Output <$\alpha$, $v_i$ contribution>
        end for
        Output <$A$, $v_i$ contribution to set of attributes>
        Output <$T$, $v_i$ contribution to set of edges>
    end for
    REDUCE 2
    Input: <{$t|\alpha|T|A$}, list of contributions (one value for each vertex)>
    Output: $S_{links}$, $S_{attr}$, $S_t$, $AD_\alpha$
    Compute $S_{links}$, $S_{attr}$, $S_t$ or $AD_\alpha$ by Eqs. (15), (16) or (19)

    Update weights using Eqs. (14), (18)
end while
```

datasets and keep runtime low. CLAMP iteratively updates the vertex memberships, the clusters and the weights according to the propositions in Sect. 4.3.

To initialize the clusters we follow the approach of random selection. That is, K vertices are randomly selected as clusters. We adopt random selection because it is the fastest approach and does not require specialized knowledge or preprocessing-analysis of the network [24], although it yields slightly different results for the same input.[8]

[8] Alternatively, several centroid initialization methods could be extended and used in the proposed approach, such as the works of Bahmani et al. [3] and Shen and Meng [23], to preprocess the network aiming to reduce the number of iterations and/or improve clustering accuracy.

The initial weights are computed by equations:

$$W_t = \frac{1}{\mathscr{T}}, \quad W_\alpha = \frac{1}{\mathscr{A}}, \quad W_{attr} = W_{links} = \frac{1}{2} \tag{21}$$

We experimentally observed that weights initialization does not affect results quality, but it may affect (increase or decrease) the number of iterations until convergence.

Algorithm 1 depicts the pseudo code of the proposed CLAMP algorithm (r denotes the iteration number). CLAMP process consists of two MapReduce jobs. The first job calculates the membership probabilities and updates the clusters; and the second job updates the weights. The clustering process terminates when at the end of the first job all clusters change insignificantly, i.e. less than a small threshold $\delta \approx 10^{-3}$.

MAP 1 Mappers of the first job compute the clustering (membership probabilities). Since each vertex can compute independently its memberships by knowing only the weights and the clusters, a mapper responsible for a vertex v_i emits <k, [i, $\Theta_{i,k}$]>, where k is the id of cluster k. In practice, a mapper may be responsible for a set of vertices. The time complexity for calculating a value in Θ is $O(\frac{|E|}{\mathscr{V}} + \mathscr{A})$.

Computed memberships are grouped by cluster id and thus each reducer receives the membership probabilities of vertices to a specific cluster, i.e. $\mathscr{C}_k$.

REDUCE 1 A reducer is responsible to update a cluster and output its description, i.e. <k, [C_k]> where [C_k] consists of two vectors representing the attribute values and the outgoing edges of cluster k. It also checks whether the specific cluster has converged. The time complexity for updating cluster $\mathscr{C}_k$ is $O(|\mathscr{C}_k| \cdot (|E| + \mathscr{A}))$.

The second MapReduce job starts the weight update process.

MAP 2 The mapper responsible for a vertex v_i outputs its contribution to every attribute and edge-type. Thus, each mapper outputs many key-value pairs, with key being the id of an attribute or an edge-type.

The individual contributions for specific attribute and edge-type are grouped into a list and forwarded to the reducers.

REDUCE 2 Each reducer sums up the contributions and outputs S_t and AD_a depending on the input key.

Weights for the next iteration are then computed sequentially. The time complexity of updating the weights is $\approx O(\mathscr{V} \cdot K \cdot (\mathscr{T} + \mathscr{A}))$.

Summarizing, the total complexity of the proposed CLAMP algorithm is $\approx O(R \cdot [\mathscr{V} \cdot K \cdot (|E| + \mathscr{T} + \mathscr{A})])$, where R is the number of iterations. As we observed during our experiments CLAMP converges in less than 10 iterations.

5 Experimental study

5.1 Datasets

Synthetic datasets To generate synthetic datasets, we modified the state-of-the-art generator presented in [20] to capture the multiple edge types and the similar connec-

Table 1 Datasets

Dataset	$\mathcal{V}$	$\|E\|$	$\mathcal{A}$	$\mathcal{T}$	Weighted
Synthetics	$\{100, 500, 1000, 5000\}$	≈1000–1,230,000	$\{2, 4, 8, 16\}$	$\{1, 2, 4, 8, 16\}$	No
DBLP-10K	10,000	65,734	2 (N = 1, C = 1)	1	Yes
EU-Projects	1965	178,623	7 (N = 6, C = 1)	2	Yes

tivity aspects. Specifically, we split the $\mathcal{V}$ vertices into K clusters. For each cluster two parameters specify its similar connectivity and attribute homogeneity. If a vertex in cluster connects to a vertex u then the *Similar Connectivity parameter* specifies the least fraction of vertices in the cluster that also connect to vertex u. Vertex outgoing degrees follow a uniform distribution. The *Attribute Homogeneity parameter* specifies the least fraction of vertices in the cluster that share the same attribute value for each attribute. Numerical and categorical attributes are drawn from uniform and Bernoulli distributions respectively. In our experiments both parameters were set to 0.8. We generated synthetic attributed graphs and multi-graphs for various cluster properties as shown in Table 1. Particularly, we vary the number of vertices in $\{100, 500, 1000, 5000\}$ with four attributes; the number of attributes in $\{2, 4, 8, 16\}$ with 1000 vertices; and the number of edge types in $\{2, 4, 8, 16\}$ for 1000 vertices and four attributes. For each variation, we generate five graphs.

Real-world datasets We used bibliography (DBLP-10K) and research-innovation projects (EU-Projects) datasets which we describe below. Table 1 summarizes these datasets.

DBLP bibliography—(DBLP-10K) dataset[9] consists of 10,000 vertices representing the top authors from the complete DBLP dataset. Each author is described by two attributes: the number of publications and the primary area of interest. We consider authors with four research areas, namely databases (DB), data mining (DM), information retrieval (IR) and artificial intelligence (AI). A weighted edge between two authors represents the number of publications they have co-authored.

Clustering this dataset into clusters with similar connectivity and attribute homogeneity is expected to identify groups of authors from the same area that have worked with common researchers probably from different areas. Such clusters can help us identify outliers or recommend new collaborations.

EU-Projects dataset consists of organizations that participated in projects funded by the European Union under the FP7 and the H2020 framework programmes. FP7 programme ran from 2007 to 2013, and H2020 programme started in 2014 and is expected to finish by 2020. In this dataset, we consider only the projects for which information was available by early 2015. That is 25808 and 2400 FP7 and H2020 projects respectively. Vertices represent organizations which are described by the following attributes: the number of H2020 and FP7 projects they are involved, coordinated, and participated in, and their country of origin. The 1965 organizations present in this

[9] The full DBLP dataset is available at http://kdl.cs.umass.edu/data/dblp/dblp-info.html.

dataset participated in at least ten projects and they are connected by two types of weighted edges representing the number of H2020 and FP7 collaborations. To execute the algorithms that do not apply to attributed multi-graphs we consider only one edge between two organizations that represents the total number of collaborations.

Clustering this dataset is expected to identify groups of organizations that share mutual H2020 and/or FP7 partners, have participated in approximately the same number of projects, and have the same country of origin. Such clusters can help us identify outliers and recommend new collaborations.

5.2 Evaluation measures and comparison methods

In the following experiments, we use **similar connectivity** (Eq. 3), **entropy**, and **normalized mutual information (NMI)** to evaluate the results. Average entropy and similar connectivity are weighted by cluster sizes. Entropy ranges in $[0, \infty)$ and is measured for an attribute α as follows:

$$entropy(\alpha) = -\sum_{k=1}^{K} \frac{|\mathscr{C}_k|}{\mathscr{V}} \cdot \sum_{j=1}^{|domain(\alpha)|} \left[p_{kj}.log\left(p_{kj}\right)\right] \tag{22}$$

where p_{kj} is the number of vertices in $\mathscr{C}_k$ that have the jth value of all possible values of attribute α $(domain(\alpha))$ divided by the size of cluster $\mathscr{C}_k$. Low entropy is equivalent to high attribute homogeneity between the vertices in the same cluster. The overall entropy for a clustering is the average entropy for all attributes. The goal of each clustering method is to achieve low overall entropy.

NMI represents the similarity between the obtained clustering and the ground truth. NMI takes values in the range of [0, 1], with 1 corresponding to clustering that perfectly matches ground truth. Given a clustering $C = \{\mathscr{C}_1, \ldots, \mathscr{C}_K\}$ and the ground-truth $B = \{\mathscr{B}_1, \ldots, \mathscr{B}_{K2}\}$, NMI is calculated as follows:

$$NMI(B, C) = \frac{H(C) - H(B|C)}{\min(H(B), H(C)} \tag{23}$$

where:

$$H(C) = -\sum_{k=1}^{K} \frac{|\mathscr{C}_k|}{\mathscr{V}} \cdot \log\left(\frac{|\mathscr{C}_k|}{\mathscr{V}}\right), \quad H(B|C) = -\sum_{i=1}^{K}\sum_{j=1}^{K_2} \frac{m_{ij}}{\mathscr{V}} \cdot \log\left(\frac{m_{ij}/\mathscr{V}}{|\mathscr{C}_j|/\mathscr{V}}\right)$$

where m_{ij} is the number of common vertices between clusters $\mathscr{B}_i$ and $\mathscr{C}_j$. In case $NMI(B, C) = 1$, then the two clusterings are identical. At this point we must mention that *NMI* can be calculated only if the ground-truth is available. Thus, we report NMI only for the experiments on synthetic datasets where clusters are known.

We evaluated CLAMP against PICS [1], CAMIR [19], BAGC [30], HASCOP [18], SA-Cluster [34] and ClampNoW. The latter is a fictitious algorithm that uses CLAMP unified distance function but treats all the weights as constants as given by Eq. (21), while the other algorithms have been discussed in Sect. 2. Our experimentation presented below demonstrates an overall better performance of CLAMP in terms of similar connectivity, attribute homogeneity and normalized mutual information.

CLAMP achieves overall better results because it successfully identifies the importance of each edge-type and attribute, and balances correctly the structural and attribute properties of the vertices.

Following experiments show average measurements out of five runs. Final clusterings were defuzzified by assigning each vertex to the cluster it belongs with the highest probability so as to use the same evaluation measures for all algorithms. The results of CLAMP are from a multi-threaded implementation in Java 1.6. All experiments conducted on a Dell Server equipped with two 12 core Intel Xeon 3.47 GHz processors and 80 GB RAM.

5.3 CLAMP parameters

Fuzzifier factor f In order to demonstrate how the fuzzifier affects clustering, Fig. 3 presents in *log-log* scale the standard deviation of the membership probabilities of a vertex as the fuzzifier value increases. Results are for a random vertex in a synthetic attributed graph. Membership probabilities follow the same distribution for all vertices since the fuzzifier is a shared constant and has the same effect on all vertices. The higher the value of fuzzifier is, the lower the standard deviation of the memberships is. The rate of decline is greater when fuzzifier is closer to 1, as demonstrated at the zoomed part of the figure. Equally, for very high values of fuzzifier the standard deviation of the memberships is almost zero. This suggests that a vertex belongs to all the clusters with almost equal probability. In such condition, vertices that have high pair-wise distances belong to the same clusters with approximately the same probability. On the other hand, as the fuzzifier is very close to 1 the standard deviation of the memberships raises, thus resulting in having a clear 'winner cluster' for each vertex. We experimentally observed that CLAMP is relatively insensitive to the parameter f as long as it is chosen to be close to 1. Our observation is inlined to other works studying the impact of the fuzzifier to the clustering task [5,14]. In the following experiments we set the fuzzifier to 1.1.

Regularization parameter λ. To study the effect of the λ parameter we executed CLAMP on ten synthetic attributed multi-graphs for $\lambda = \{0.01, 0.1, 1.0, 10.0, 100.0\}$. High value of λ penalizes more the objective function and forces the weights to be closer to each other (consequently, all weights are closer to the average). Since the datasets were generated by setting the same parameters for each edge type and attribute, we expect the edge type and attribute weights to be almost evenly distributed. Table 2

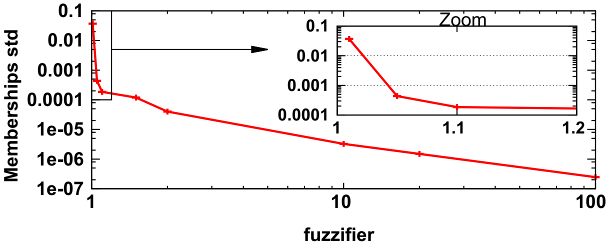


Fig. 3 Impact of fuzzifier to memberships

Table 2 Impact of regularization parameter (λ) to weights

λ	Weights variance		
	W_T	W_A	W_{Attr}, W_{links}
0.01	0.0482 ± 0.0233	0.0477 ± 0.0411	0.1741 ± 0.1070
0.1	0.0101 ± 0.031	0.0157 ± 0.0349	0.0234 ± 0.0655
1.0	0.0022 ± 0.000539	0.0037 ± 0.00377	0.0203 ± 0.0232
10.0	0.0015 ± 0.000304	0.0021 ± 0.000847	0.0157 ± 0.0202
100.0	$5.97e{-}05 \pm 1.02e{-}05$	$5.38e{-}05 \pm 1.4e{-}05$	0.0002 ± 0.0001
1000.0	$7.39e{-}07 \pm 1.47e{-}07$	$7.96e{-}07 \pm 1.88e{-}07$	$1.33e{-}06 \pm 1.28e{-}06$

confirms that high value of λ results in negligible weights variance. High value of λ leads the weighting mechanism to 'push' the weights to be all almost the same independently of the network properties. The lower the value of parameter λ the higher the variance of the weights is. Withal, if value of λ is very small the variance of the weights is high because no or limited regularization is performed and some weights may be very close to zero. In this case, some attributes/edge types or even the entire set of attributes/edges are ignored. However, a 'perfect' λ value does not exist and it must be fine tuned for each dataset either empirically or automatically, i.e. using cross-validation [33]. In the following experiments we adopt the tenfold cross-validation technique to set λ.

5.4 Evaluation on synthetic graphs

To study the clustering performance of CLAMP we generated synthetic attributed graphs and multi-graphs varying the number of vertices, attributes and edge-types. According to Figs. 4 and 5, CLAMP outperforms all its competitors in terms of entropy, similar connectivity and NMI. The high clustering accuracy of CLAMP is based on the proposed unified distance measure and the presented weighting mechanism that correctly identifies the importance of the different vertex properties. PICS results in high entropy, high similar connectivity and low NMI, because it converges too early and returns few clusters, by using a self-tuning strategy to determine the number of clusters. BAGC and SA-Cluster achieve low entropy on all datasets, but they do not identify clusters characterized by low similar connectivity because they search for densely connected components. Specifically, they achieve similar connectivity of at least one order of magnitude higher than the other approaches, and thus we omit them from Figs. 4c, d, 5b. These figures depict in logarithmic scale the similar connectivity of the results, demonstrating that CLAMP achieves much lower similar connectivity than its competitors on all synthetic datasets. HASCOP and CAMIR also achieve very good results on both attributed graphs (Fig. 4) and attributed multi-graphs (Fig. 5), since they also weigh the vertex properties efficiently. However, CLAMP results on synthetic attributed multi-graphs are characterized by at least 16% lower entropy, 20% lower similar connectivity and 7% higher NMI compared to HASCOP and CAMIR

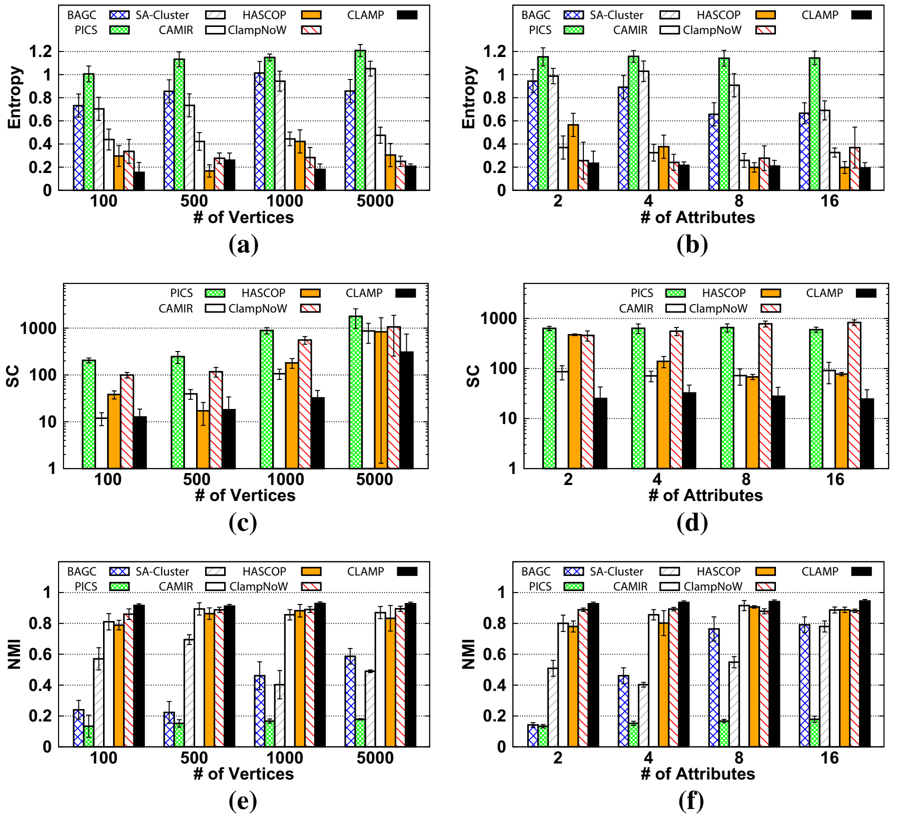


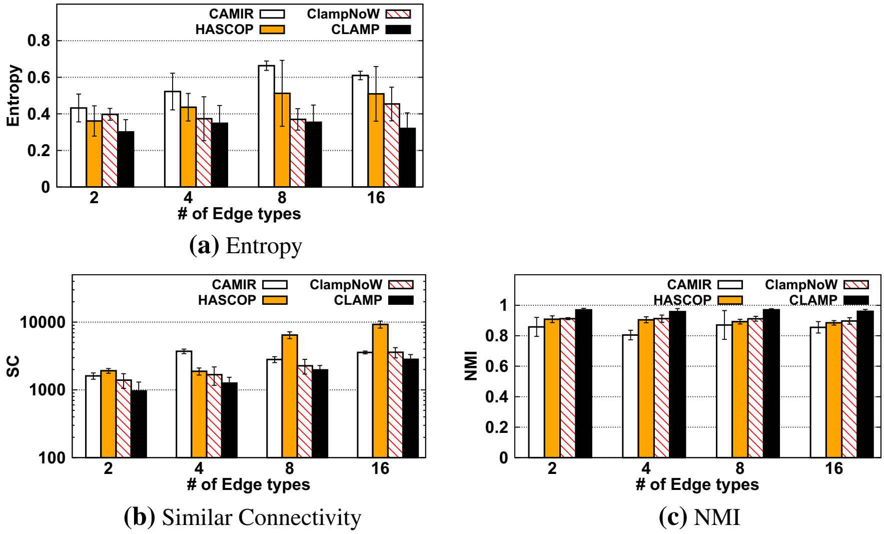
Fig. 4 Clustering performance on synthetic attributed graphs. BAGC and SA-Cluster are omitted from **c** and **d** because they achieve similar connectivity of at least one order of magnitude higher than the other approaches

results. Furthermore, there is a rather steady performance of CLAMP on all synthetics datasets. We conclude that CLAMP outperforms all its competitors on both attributed graphs and multi-graphs with respect to entropy, similar connectivity and NMI. That is because it adapts the vertex property weights and the global weights according to the properties of the datasets during the clustering process.

5.5 Evaluation on real-world graphs

Clustering the DBLP-10K dataset For DBLP-10K dataset we executed the algorithms for $K = \{20, 40, 60, 80, 100\}$. For CLAMP λ has been set to 1000 using tenfold cross validation. PICS and HASCOP returned 18 and 763 clusters respectively.

It is noted that since authors follow different careers and usually co-work with researchers from different locations and organizations, it is hard to find a group of authors that has many common co-authorships. This is also evident from the small average outgoing degree of a vertex (approximately 6, 5). CLAMP weighting mecha-

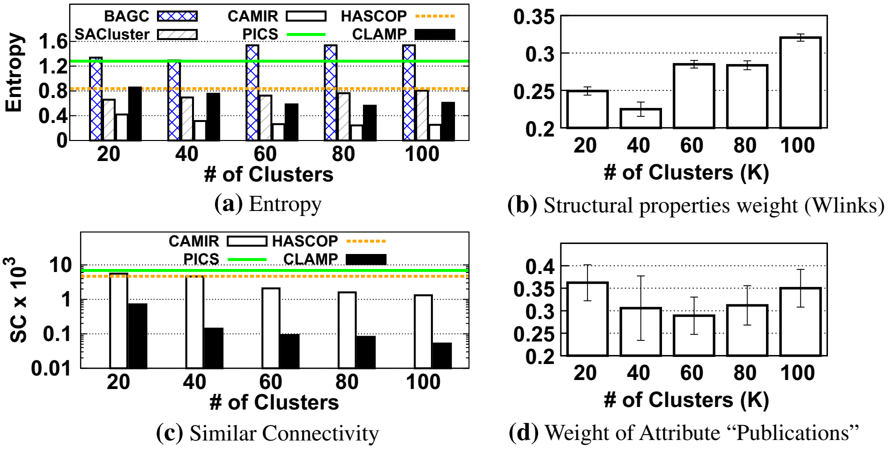


(a) Entropy

(b) Similar Connectivity

(c) NMI

Fig. 5 Clustering performance on synthetic attributed multi-graphs. BAGC, SA-Cluster and PICS are omitted because they ignore multiple edge-types



(a) Entropy

(b) Structural properties weight (Wlinks)

(c) Similar Connectivity

(d) Weight of Attribute "Publications"

Fig. 6 Clustering quality on DBLP-10K bibliography dataset. Since HASCOP and PICS use a self-tuning strategy to determine the number of clusters, both methods are denoted by *straight lines*. BAGC and SA-cluster achieve at least one order of magnitude higher similar connectivity

nism 'captures' this property of the dataset and automatically assigns lower importance to structural properties of the vertices, as shown in Fig. 6b ($W_{attr} = 1 - W_{links}$). Moreover, CLAMP has identified the importance of each attribute as 'Area' is considered more important attribute than 'Publications' (Fig. 6d). This is expected since 'Publications' entropy is higher and the objective function is optimized by assigning lower weight-importance to it. In addition, because of the negative entropy regularization

Table 3 Case study on DBLP-10K dataset

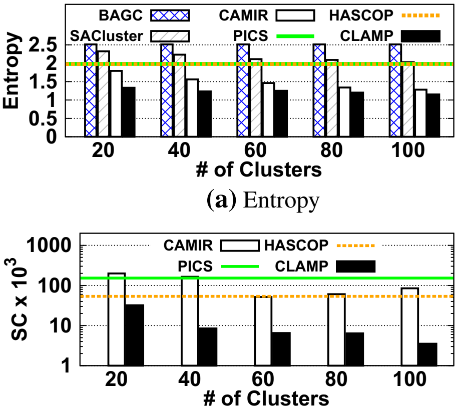
Cluster id	Author	Area	Publications	Co-authored publications	Mutual co-authors
1	Wei Wang	DM	83	7	21
	Jian Pei		70		
2	Hector Garcia-Molina	DB	100	0	8
	Gerhard Weikum		92		
3	Philip S. Yu	DB	216	25	29
	Jiawei Han	DM	168		
4	David Hogg	ML	2	–	–

all weights are at least 0.2. Hence, CLAMP successfully prevents the dominance of a specific attribute and edge-type.

Furthermore, CLAMP achieves the lowest similar connectivity (Fig. 6c), and on average the second lowest entropy after CAMIR (Fig. 6a) among its competitors. Generally, the more the clusters are the lower the entropy is. This is due to the fact that the more the clusters are the easier it is to group together only vertices with the same (or very close) attributes. CAMIR resulted in approximately 50% lower entropy than CLAMP because it identifies clusters of arbitrary sizes and also assigned much higher importance-weight to attributes. However, CAMIR results in at least 87% more similar connectivity than CLAMP as shown in logarithmic scale in Fig. 6c. Combining both the entropy and similar connectivity measures, Fig. 6 demonstrates an overall better performance of CLAMP, while it confirms the efficiency of the proposed weighting mechanism.

We further examined the actual clusters obtained by CLAMP and we present some representative ones in Table 3. We see that CLAMP revealed some interesting clusters of authors. For example, in the first cluster there are two authors (Wei Wang and Jian Pei) who both work on databases, have approximately the same number of publications and share 21 mutual collaborators. On the other hand, although Hector Garcia-Molina and Gerhard Weikum (cluster 2) have not co-authored any publication, they are placed in the same cluster because they both work on databases and they also have eight mutual co-authors. Such clusters can be definitely used for collaboration recommendations. Their identification is strongly based on the similar connectivity which CLAMP tries to optimize. The first two clusters exhibit both attribute homogeneity and similar connectivity, while authors in the first cluster are also connected to each other. Cluster 3 consists of two well-known authors that are interested in different areas and have high difference on their number of publications. They are placed on the same cluster because they have 29 mutual co-authors and collaborated on 25 articles. Lastly, as an example we present a cluster which consists of only one author. David Hogg has been categorized by it self since it has no collaborators in the DBLP-10K dataset. We conclude that the experiment on DBLP10-K confirms that *CLAMP successfully balances the structural and attribute properties of the vertices, discovers clusters with high attribute homogeneity, and reveals meaningful clusters in the network under study*.

Fig. 7 Clustering quality on EU-Projects dataset. Since HASCOP and PICS use a self-tuning strategy to determine the number of clusters, both methods are denoted by *straight lines*. BAGC and SA-cluster achieve at least one order of magnitude higher similar connectivity



(a) Entropy

(b) Similar Connectivity

Clustering the EU-Projects dataset For EU-Projects dataset we executed the algorithms for $K = \{20, 40, 60, 80, 100\}$. Parameter λ has been set to 1000. HASCOP and PICS returned 64 and 8 clusters respectively. Figure 7 presents the average entropy and similar connectivity for multiple number of clusters demonstrating that CLAMP achieves the lowest entropy and similar connectivity among its competitors. Specifically, there is an improvement on entropy and similar connectivity by at least 11 and 40% respectively. The success of CLAMP is strongly based on the proposed unified distance measure and the weighting mechanism that adapts the weights based on the clustering results at each iteration.

Furthermore, a sample of the obtained clusters is depicted in Table 4. The first two clusters consist of organizations that have not collaborated in any project, they are from different countries, and they participated in approximately the same number of projects. These organizations have been grouped together in clusters corresponding to potential new collaborations because they have common partners. Cluster 3 consists of three organizations which have 23 mutual partners. “Tractebel Engineering” and “Vuje” have participated in three common projects but none of them has collaborated with “ANDRA”. However, “ANDRA” has been categorized in this cluster because it shares more than 50 mutual partners with each of them. The last clusters (cluster 4 and cluster 5) consist of only one organization. Although these organizations share a high number of common partners they have not been categorized in the same cluster because of the weight of each individual collaboration. For instance, “University of Oxford” and “École polytechnique fédérale de Lausanne” collaborated on 27 FP7 projects, while “University of Cambridge” collaborated with the same organization on only 15 FP7 projects. These differences lead to high ‘Total Similar Connectivity’ distance, and consequently to their categorization in different clusters. Overall, the above clusters confirm that *CLAMP exploits properly the multiple weighted edges connecting the organizations in order to identify meaningful clusters*.

Table 4 Case study on EU-Projects dataset

Cluster id	Organization	FP7partners	H2020 partners	Projects	Country	Mutual partners
1	Montanuniversität Leoben	35	10	11	Austria	2
	Spinverse	17	17	6	Finland	
2	Ateknea Solutions	38	0	27	Hungary	7
	Nederlands Normalisatie Instituut (NEN)	63	14	17	Netherlands	
3	Agence nationale pour la gestion des déchets radioactifs (ANDRA)	116	0	12	France	23
	Tractebel Engineering	192	0	12	Belgium	
	Vuje	172	0	13	Slovakia	
4	University of Oxford	994	115	793	United Kingdom	–
5	University of Cambridge	917	126	798	United Kingdom	–

5.6 Efficiency study

To examine the efficiency of our algorithm, Table 5 presents the execution time of the algorithms. The average runtime for all presented numbers of clusters is shown, and thus there is high standard deviation. PICS and BAGC do not consider the importance of the vertex properties which leads to lower complexity and thus they are quite faster than the other algorithms; though, they have limited clustering accuracy as shown, in Fig. 4a–f. SACluster, HASCOP and CLAMP are the slowest because they update the weights during the clustering process. CLAMP is quite faster than HASCOP and SACluster for almost all used datasets. HASCOP is the slowest because as an agglomerative algorithm during the first iterations the number of clusters is in the order of the number of vertices, which increases significantly the iteration runtime. Around 50% of CLAMP runtime is spent on the weights update process as we observed in our experiments. Despite that slight time overhead, as previous experiments demonstrate CLAMP achieves results of high quality.

We further examine the efficiency of the proposed algorithm by measuring the execution time for the above datasets. Figure 8 shows the average runtime of the algorithm on synthetic multi-graphs of 1000 vertices for multiple number of threads and clusters. We observe that 24 threads gain an average speed up of 10× against 2 threads, thus CLAMP requires less than two minutes to complete the clustering. Moreover, it is evident that CLAMP scales almost linearly to the number of clusters. For the real-world datasets (Fig. 9), as expected, DBLP-10K requires most of the time because it is the largest used attributed graph. Also, CLAMP takes less than a minute to cluster the EU-Projects dataset, a network of about 2000 vertices and 7 attributes. Concluding Figs. 8 and 9, results are consisted with our observations in complexity analysis section. Moreover, Fig. 8 confirms that *CLAMP is highly parallelizable* since more threads gradually decrease the runtime of the algorithm.

Table 5 Runtime (s)

	BAGC	SACluster	PICS	CAMIR	HASCOP	CLAMP
Synthetic graphs	0.5 ± 0.1	376 ± 55	22 ± 4	4.2 ± 1	$24{,}136 \pm 472$	108 ± 17
Synthetic multigraphs	–	–	–	56 ± 7	$27{,}341 \pm 486$	157 ± 23
DBLP-10K	0.55 ± 0.07	433 ± 68	495 ± 25	520 ± 104	$32{,}957 \pm 675$	$22{,}110 \pm 215$
EU-Projects	0.61 ± 0.2	52 ± 19	38 ± 1.9	18 ± 2.7	3830 ± 159	29 ± 14

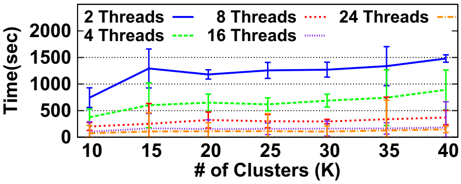


Fig. 8 CLAMP runtime on synthetic graphs versus number of clusters for multiple threads

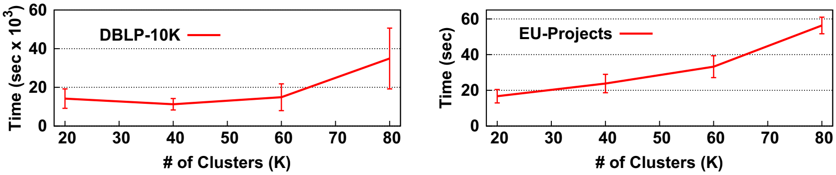


Fig. 9 CLAMP (24-threads) runtime

6 Conclusions

In this work we studied the problem of attributed multi-graph clustering. Specifically, we propose a new unified distance measure for attributed multi-graphs and develop CLAMP (CLustering Attributed Multi-graPh), which is to the best of the authors' knowledge, the first to perform fuzzy clustering on weighted directed attributed multi-graphs with heterogeneous attributes. CLAMP considers simultaneously the individual importance of the attributes and edge-types as well as the balance between the sets of attributes and edges, by assigning them different weights that are identified during the clustering process in an automatic manner. The goal is to determine clusters of objects that are characterized by similar connectivity and attribute homogeneity, which have been shown to be of great interest to many applications [1,18]. Lastly, CLAMP is highly parallelizable, thus it can exploit properly the computational power of modern many- and multi-core architectures so as to scale to large datasets. Our extensive experimental evaluation on synthetic datasets and a diverse collection of real world information networks demonstrates the efficiency and effectiveness of the proposed approach.

Acknowledgements This work was partially supported by the EU Commission in terms of the PaaSport 605193 FP7 Project (FP7-SME-2013).

References

1. Akoglu L, Tong H, Meeder B, Faloutsos C (2012) PICS: parameter-free identification of cohesive subgroups in large attributed graphs. In: Proceedings of the 12th SIAM international conference on data mining, SDM 2012
2. Akoglu L, Tong H, Koutra D (2015) Graph based anomaly detection and description: a survey. Data Min Knowl Discov 29(3):626–688
3. Bahmani B, Moseley B, Vattani A, Kumar R, Vassilvitskii S (2012) Scalable k-means++. Proc VLDB Endow 5(7):622–633
4. Barbieri N, Bonchi F, Galimberti E, Gullo F (2015) Efficient and effective community search. Data Min Knowl Discov 29(5):1406–1433
5. Bezdek JC, Ehrlich R, Full W (1984) FCM: the fuzzy c-means clustering algorithm. Comput Geosci 10(2–3):191–203
6. Bothorel C, Cruz JD, Magnani M, Micenkova B (2015) Clustering attributed graphs: models, measures and methods. Netw Sci 3:408–444
7. Cheng H, Zhou Y, Huang X, Yu J (2012) Clustering large attributed information networks: an efficient incremental computing approach. Data Min Knowl Discov 25(3):450–477
8. Galbrun E, Gionis A, Tatti N (2014) Overlapping community detection in labeled graphs. Data Min Knowl Discov 28(5–6):1586–1610

9. Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-completeness. W. H. Freeman & Co., New York
10. Gunnemann S, Farber I, Raubach S, Seidl T (2013) Spectral subspace clustering for graphs with feature vectors. In: 2013 IEEE 13th international conference on data mining (ICDM), pp 231–240. doi:10.1109/ICDM.2013.110
11. Hu X, Xu L (2004) Investigation on several model selection criteria for determining the number of cluster. Neural Inf Process Lett Rev 4(1):1–10
12. Huang HC, Chuang YY, Chen CS (2012) Multiple kernel fuzzy clustering. IEEE Trans Fuzzy Syst 20(1):120–134
13. Huang Z (1998) Extensions to the k-means algorithm for clustering large data sets with categorical values. Data Min Knowl Discov 2(3):283–304
14. Klawonn F, Höppner F, (2003) What is fuzzy about fuzzy clustering? Understanding and improving the concept of the fuzzifier. Advances in Intelligent Data Analysis V, vol 2810, Lecture Notes in Computer Science. Springer, Berlin, pp 254–264
15. Kumar A, Rai P, Daume H (2011) Co-regularized multi-view spectral clustering. In: Shawe-Taylor J, Zemel R, Bartlett P, Pereira F, Weinberger K (eds) Advances in neural information processing systems, vol 24. Curran Associates, Inc., pp 1413–1421
16. Li N, Sun H, Chipman KC, George J, Yan X (2014) A probabilistic approach to uncovering attributed graph anomalies. In: Zaki MJ, Obradovic Z, Tan P, Banerjee A, Kamath C, Parthasarathy S (eds) Proceedings of the 2014 SIAM international conference on data mining, Philadelphia, SIAM, pp 82–90
17. Mann GS, McCallum A (2007) Efficient computation of entropy gradient for semi-supervised conditional random fields. Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume. Short Papers, Association for Computational Linguistics, pp 109–112
18. Papadopoulos A, Pallis G, Dikaiakos MD (2013) Identifying clusters with attribute homogeneity and similar connectivity in information networks. IEEE/WIC/ACM international conference on web intelligence
19. Papadopoulos A, Rafailidis D, Pallis G, Dikaiakos M (2015) Clustering attributed multi-graphs with information ranking. In: database and expert systems applications, Lecture Notes in Computer Science. Springer International Publishing
20. Perozzi B, Akoglu L, Sánchez PI, Müller E (2014) Focused clustering and outlier detection in large attributed graphs. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining, ACM, KDD '14
21. Rissanen J (1978) Modeling by shortest data description. Automatica 14(5):465–471
22. Schaeffer SE (2007) Graph clustering. Comput Sci Rev 1(1):27–64
23. Shen S, Meng Z (2012) Optimization of initial centroids for k-means algorithm based on small world network. In: Shi Z, Leake D, Vadera S (eds) Intelligent information processing VI, IFIP Advances in Information and Communication Technology, vol 385. Springer, Berlin, pp 87–96
24. Steinbach M, Kumar V (2005) Cluster analysis: basic concepts and algorithms. In: Introduction to data mining, 1st edn. Pearson Addison Wesley
25. Steinhaeuser K, Chawla N (2008) Community detection in a large real-world social network. In: Liu H, Salerno J, Young M (eds) Social computing, behavioral modeling, and prediction. Springer, USA, pp 168–175
26. Sun H, Huang J, Han J, Deng H, Zhao P, Feng B (2010) gSkeletonClu: density-based network clustering via structure-connected tree division or agglomeration. In: Proceedings of the 2010 IEEE international conference on data mining. IEEE Computer Society, Washington, DC, ICDM '10, pp 481–490. doi:10.1109/ICDM.2010.69
27. Sun Y, Aggarwal CC, Han J (2012) Relation strength-aware clustering of heterogeneous information networks with incomplete attributes. Proc VLDB Endow 5
28. Vuokko N, Terzi E (2010) Reconstructing randomized social networks. In: Proceedings of the SIAM international conference on data mining, SDM 2010, April 29–May 1, 2010, Columbus, pp 49–59
29. Xu X, Yuruk N, Feng Z, Schweiger TAJ (2007) SCAN: a structural clustering algorithm for networks. In: Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, KDD '07, pp 824–833. doi:10.1145/1281192.1281280

30. Xu Z, Ke Y, Wang Y, Cheng H, Cheng J (2012) A model-based approach to attributed graph clustering. In: Proceedings of the 2012 international conference on management of data. ACM, New York, SIGMOD '12
31. Xu Z, Ke Y, Wang Y, Cheng H, Cheng J (2014) GBAGC: a general bayesian framework for attributed graph clustering. ACM Trans Knowl Discov Data 9(1):5:1–5:43
32. Yang J, McAuley J, Leskovec J (2013) Community detection in networks with node attributes. In: IEEE international conference on data mining, IEEE, pp 1151–1156. doi:10.1109/ICDM.2013.167
33. Zhong E, Fan W, Yang Q, Verscheure O, Ren J (2010) Cross validation framework to choose amongst models and datasets for transfer learning. In: Proceedings of the 2010 European conference on machine learning and knowledge discovery in databases: part III. Springer, Berlin, ECML PKDD'10, pp 547–562
34. Zhou Y, Cheng H, Yu JX (2009) Graph clustering based on structural/attribute similarities. Proc VLDB Endow 2(1):718–729