Journal of Experimental & Theoretical Artificial Intelligence

ISSN: 0952-813X (Print) 1362-3079 (Online) Journal homepage: http://www.tandfonline.com/loi/teta20

A novel clustering algorithm for attributed graphs based on K-medoid algorithm

Saeed Farzi & Sahar Kianian

To cite this article: Saeed Farzi & Sahar Kianian (2018): A novel clustering algorithm for attributed graphs based on K-medoid algorithm, Journal of Experimental & Theoretical Artificial Intelligence, DOI: 10.1080/0952813X.2018.1467498

To link to this article: https://doi.org/10.1080/0952813X.2018.1467498

Published online: 08 May 2018.

JOURNAL OF EXPERIMENTAL & THEORETICAL ARTIFICIAL INTELLIGENCE, 2018
https://doi.org/10.1080/0952813X.2018.1467498

Taylor & Francis
Taylor & Francis Group

A novel clustering algorithm for attributed graphs based on K-medoid algorithm

Saeed Farzi[a] and Sahar Kianian[b]

[a]Faculty of Computer Engineering, K. N. Toosi University of Technology, Tehran, Iran; [b]Faculty of Computer Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran

ABSTRACT

Articulateness and plasticity are two essential attributes that make a graph as an efficient model to real life problems. Nowadays, the attributed graph is received lots of attentions because of usability and effectiveness. In this study, a novel k-Medoid based clustering algorithm, which focuses simultaneously on both structural and contextual aspects using Signal and the weighted Jaccard similarities, are introduced. Two real life data-sets, Political Blogs and DBLP bibliography, are employed in order to evaluate and compare the proposed algorithm with state-of-the-art clustering algorithms. The results show the superiorities of the proposed algorithm in terms of cluster quality metrics.

ARTICLE HISTORY
Received 12 November 2017
Accepted 17 April 2018

KEYWORDS
Attributed graph; clustering algorithm; signal similarity; community detection

1. Introduction

Graphs have been extensively adopted to model objects or entities' characteristics and relationships by various interdependencies like friendship, kinship in varieties application domains, i.e. social network analysis, world wide web and sensor networks. It is also one of the fundamental primitive models to study human interactions in social systems. Topological characteristics of graphs represent individuals and relationships in social science. Detecting communities has an important role to study structure, function and evolution of graphs (Newman, 2003). It has received lots of attention because of universal applications, such as identifying important modules in biological graphs (Guimera & Amaral, 2005; Ravasz, Somera, Mongru, Oltvai, & Barabási, 2002; Wilkinson & Huberman, 2004), collecting the related topics' web pages (Dourisboure, Geraci, & Pellegrini, 2007) and detecting events/trends on social networks (Cazabet, Takeda, Hamasaki, & Amblard, 2012; Konstantinidis, Papadopoulos, & Kompatsiaris, 2017). The traditional algorithms are used to, employing topological characteristics, to detect communities. The topological characteristics, which are represented by individuals and their relationships, are useful to distinguish densely connected components such as communities. However, in real-world networks, node/edges attributes have the same importance as topological structures to study function and evolution of networks. Attributed graph is an extended graph using node and edge attributes. A node attribute represents a particular feature to describe context and semantic of the node. Similarly, an edge attribute introduces the kind of relationships among nodes. The attributed graph provides a rich model of a real-world network rather than the traditional graph which provides a partial model of a real-world network (Bothorel, Cruz, Magnani, & Micenkova, 2015). Nowadays, existing graph clustering methods are extended to handle information provided by node and edge attributes.

CONTACT Saeed Farzi saeedfarzi@kntu.ac.ir

Nevertheless, the most methods either concentrate on topological information of an input graph so that every cluster achieves a cohesive internal structure or concentrate on attributes' likeness known as contextual information of the graph so that every cluster contains homogenous nodes in terms of attribute values; however, few recent methods utilise both information resources simultaneously via balancing the structural and contextual information (Cheng, Zhou, & Yu, 2011; Nawaz, Khan, Lee, & Lee, 2015). By considering the attributed graph definition, an attributed graph clustering algorithm is aimed at grouping densely connected nodes with homogeneous attribute values. Here, the most important challenge is independence and even contradiction of structural and contextual similarities (Xu, Ke, Wang, Cheng, & Cheng, 2012).

In this study, a novel clustering algorithm is proposed to partition an attributed weighted graph. The proposed algorithm is based on the k-Medoid clustering algorithm and uses both structural and contextual information of an input graph. The algorithm utilises a balancing factor to inject structural and contextual information over clustering results. The proposed algorithm is used to employ N^{th} nearest neighbour to find a local neighbourhood of every node. The main idea is transforming the attributed graph into a spatial space and measuring structural and contextual similarities among nodes through the space. The proposed algorithm employs a signal similarity (Hu, Li, Zhang, Fan, & Di, 2008) to integrate structural information, and a weighted Jaccard similarity (Popescu, Keller, & Mitchell, 2006) to integrate contextual information of an input graph into a spatial space which provides us advantages of spatial clustering algorithms. A major difference between the graph clustering and the spatial clustering is that the former computes nodes' similarities based on structural connectivity of nodes, whereas the later computes data points' similarities based on both structural and contextual similarities among data points (Zhou, Cheng, & Yu, 2010).

The number of clusters and key initial seeds are two essential parameters that have to be determined in advance to start clustering algorithms. Within the new spatial space, key initial seeds are high influential data points surrounded by low influential data points. Therefore, the high influential data points which are located far from other the high influential data points are the best candidates for key initial seeds. An influential data point is a high density point among its neighbours. The number of clusters is determined when the key initial seeds have been selected. Eventually, the proposed algorithm is going to optimise an objective function aimed at maximising intra-cluster similarity and minimising inter-cluster similarity. Two real data-sets, Political Blogs Data-set (Nawaz et al., 2015), which includes 1490 nodes and DBLP Bibliography (Nawaz et al., 2015) data-set which includes 5000 nodes, have been employed to perform experiential evaluations. Several evaluation scenarios have been followed in order to show the performance of the proposed algorithm. In addition to investigating the impact of parameters, the proposed algorithm has been compared with four state-of-the-art algorithms, i.e. SA-Cluster (Cheng et al., 2011), W-Cluster (Cheng et al., 2011), S-Cluster (Cheng et al., 2011) and KSNAP (Tian, Hankins, & Patel, 2008). The results illustrate the superiorities of the proposed method rather than the mentioned algorithms in terms of density and entropy metrics known as two main clustering quality measures.

The rest of the paper is organised as follows. In Section 2, related works are reviewed. Problem definition is explained in Section 3. In Sections 4, the proposed method is described in more detail. In Section 5, the experimental studies are presented. Finally, some conclusions are drawn in Section 6.

2. Related works

Many graph partitioning algorithms mainly concentrate on the structural aspect based on various objective functions including modularity (Newman & Girvan, 2004), normalised cuts (Shi & Malik, 2000) and overall density (Xu, Yuruk, Feng, & Schweiger, 2007). The outputs of such clustering algorithms usually include densely connected nodes within clusters, whereas nodes' attributes are ignored by clustering algorithms.

Nowadays, the graph partitioning algorithms concentrate more on both structural and contextual aspects of the attributed graphs. The Metis and Markov Clustering are utilised by CODICIL (Ruan, Fuhry, & Parthasarathy, 2013) in order to mix both content and link similarities. The probability of belonging

to an edge in a cluster is utilised to estimate the link similarity, whereas Jaccard coefficient is utilised to estimate content similarity. The SA-Cluster (Cheng et al., 2011) has introduced a clustering algorithm based on random walk and unified distance measure in order to integrate structural and contextual similarities. The given graph is clustered into per-defined number of clusters. Every cluster includes densely connected nodes with homogenous attribute values. In comparison to the SA-Cluster, the S-cluster algorithm introduced by Xu et al. (Xu et al., 2007) has high structural similarity and low contextual similarity. The KSNAP algorithm introduced by Tian et al. (2008) concentrates only on attribute. It collects nodes with same attributes in the same cluster.

3. Problem definition

An attributed weighted directed graph, hereafter called an AWD graph, is denoted as $G = (V, E, C, A, W)$ where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of nodes, $E = \{ < v_i, v_j > | v_i, v_j \in V\}$ is a set of directed edges with cost value $c_{ij} \in C$ and $A = \{a_1, a_2, \ldots, a_m\}$ is a set of m attributes for every node in order to describe node properties. A attribute a_q related to a node v_i, $a_q(vi) \in A$, with domain cardinality $d_q = |Dom(a_q)|$, is weighted by $w_q(v_i) \in W$. Thus, node v_i has an attribute vector $\vec{A}(v_i)$ with $|\vec{A}(v_i)| = \sum_{i=1}^{m} d_i$. Each pair of < attribute, value > maps to an index of attribute vector.

Figure 1 depicts an authorship graph as an example of an AWD graph which contains authors as nodes and co-author relationships as edges. Here, there are two attributes for every node (research topic and native language).

The clustering of the AWD graphs is a process which distinguishes K disjoint sub-graphs $G^k = (V^k, E^k, C^k, A^k, W^k)$ where $V = \bigcup_{k=1}^{K} V^k$, $E = \bigcup_{k=1}^{K} E^k$ and $V^i \cap V^j = \varnothing$ for any i≠j.

A clustering algorithm is aimed at finding a good balance to optimise the following two independent and even contradictory objectives: (1) Structural Objective: nodes in clusters are similar and nodes among clusters are dissimilar in terms of structural aspect. (2) Contextual Objective: nodes in clusters are similar to each other and nodes among clusters are dissimilar in terms of contextual aspect.

In order to balance the structural and contextual objectives, a balancing factor, $\lambda \in [0, 1]$, is used to linearly combine the both objective functions as follows:

$$O_f = \lambda \times O_{str} + (1 - \lambda) \times O_{con} \tag{1}$$

where O_{str} and O_{con} are the structural and contextual objective functions, respectively, which are going to be defined in Section 4.3.

In order to design a clustering algorithm, two important questions must be answered: (1) How to calculate the structural and contextual similarities among pairs of nodes? and (2) How to optimise the aforementioned objective function? In order to answer the first question, the structural and contextual similarities are integrated into a normalised spatial space named Spatial Integrated Platform and subsequently the dot product is utilised to measure the similarities. In order to answer to the last question, the proposed algorithm utilises the k-Medoid-based approach to optimise the objective function. The number of clusters and key initial seeds are two important challenges which have to be taken into account. In the following section, first, an overall architecture of the proposed method is introduced and then all parts are explained in detail.

4. Proposed method

As shown in Figure 2, the proposed method, hereafter called SS-cluster, takes an AWD graph as an input and outputs a partitioned graph. The SS-cluster contains three main parts: (1) a transition algorithm, (2) a seed finder algorithm and (3) a partitioning algorithm. The first part, the transition algorithm, integrates structural and contextual information of an input AWD graph to a Spatial Integrated Platform (SIP). For this purpose, a signal similarity (Hu et al., 2008) and a weighted Jaccard similarity (Popescu et al., 2006) are utilised to measure structural and contextual information, respectively.

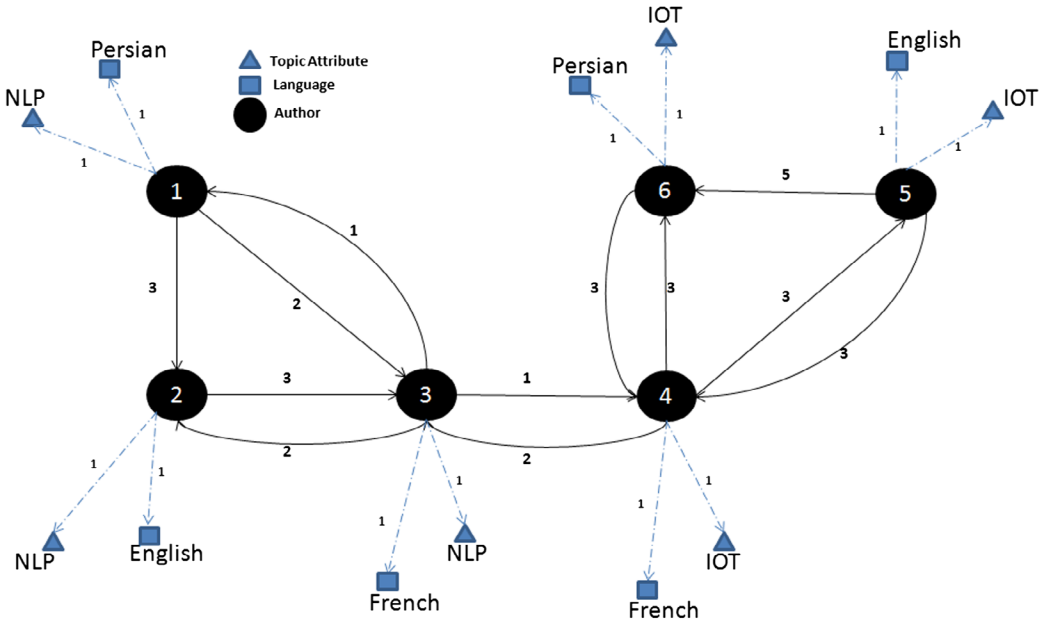


Figure 1. An example of a AWD graph.

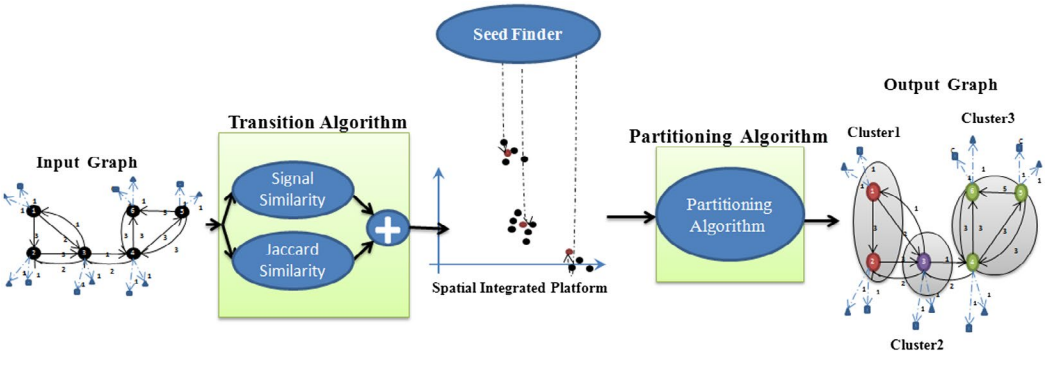


Figure 2. System Architecture.

The number of clusters and key initial seeds are two main challenges of the partitioning algorithms (Kianian, Khayyambashi, & Movahhedinia, 2016). In order to come up with these challenges, the second part of SS-cluster, *seedFinder* algorithm, determines high density nodes surrounded by low density ones as key initial seeds. The last part, the partitioning algorithm, avoids common pitfalls via finding key initial seeds and determining the number of clusters. The partitioning algorithm optimises an objective function defined with regard to maximising intra-cluster similarity and minimising inter-cluster similarity. Cluster similarity is measured using a linear combination of the structural and contextual similarities through the spatial integrated platform.

In following, the transition, seed finder, and partitioning algorithms are described in detail.

4.1. Transition algorithm

In order to adopt spatial space clustering algorithms to partition a given AWD graph, structural and contextual information of the graph is integrated into a Spatial Integrated Platform.

4.1.1. Structural transition

To transfer structural and topological information provided by a given AWD graph into a spatial space, a signal diffusion (Hu et al., 2008), which is one of the popular and well-known information diffusion algorithms, is employed to define a pairwise signal similarity function between two nodes. Based on experimental studies (Li, Jia, & Yu, 2015), the signal similarity is better than other similarity functions such as Cosine and Jaccard. Signal propagation is the basic principle to compute signal similarity. To calculate the signal similarity for a graph with *n* nodes, each node is considered as a system with abilities to send, receive and record signals.

During an iterative process, each node is elected as an initial signal source to stimulate all other nodes of the given graph. By assigning a unit signal to the initial source node, the process is started. After propagating signals, the initial node and all its neighbours save the received signal in an n-dimensional vector and propagate it again to all their neighbours. After *t* steps, the influence of source nodes on whole graph is calculated by the amount of the received signals shown in the n-dimensional vector of nodes. Finally, the topological relationship of nodes is transferred to an n-dimensional spatial space (Hu et al., 2008). Thus, a similarity between node *i* and all other nodes can be calculated via Euclidean distance in the n-dimensional space (Farzi & Dastjerdi, 2010). In fact, the mentioned signal propagation method can be formulated by a simple and clear mathematical process as shown by Equation (2).

$$S = (P + I)^t \tag{2}$$

where *P*, *I* and *t* are an adjacency matrix of the given graph, an n-dimensional identity matrix and the number of iterations, respectively. The column *i* of the matrix *S* represents an n-dimensional impact vector of node *i* after *t* steps. To normalise each column of the matrix *S*, Equation (3) is used.

$$\widehat{S}_{ij} = \frac{S_{ij}}{\sum_i S_{ij}} \tag{3}$$

where $\widehat{S}_{ij}$ represents a normalised structural similarity between node *i* and *j*.

4.1.2. Contextual transition

Jaccard similarity is one of the mostly used functions to calculate similarities between two sets of items. Here, a node of a graph is represented as a set of attributes with different degrees of importance shown by attribute weights. Because of weighted attributes, the weighted Jaccard (Ioffe, 2010) (Ioffe, 2010) similarity function is adopted. Therefore, to transform the contextual information provided by the attributed graph into an n-dimensional spatial space, the weighted Jaccard (Ioffe, 2010) similarity is calculated between attribute vectors of two nodes as follows:

$$\widehat{C}_{ij} = \frac{\sum_{q=1}^{\sum_{i=1}^{m} d_i} \min(\vec{A}_q(i), \vec{A}_q(j))}{\sum_{q=1}^{\sum_{i=1}^{m} d_i} \max(\vec{A}_q(i), \vec{A}_q(j))} \tag{4}$$

where $\vec{A}(i)$ and $\vec{A}(j)$ are non-negative attribute vectors of node *i* and node *j*, respectively. *min(…)* and *max(…)* are minimum and maximum functions, respectively.

4.1.3. The proposed transition algorithm

A transition algorithm is used to generate a Spatial Integrated Platform, SIP, via integrating structural and contextual information. The normalised n-dimensional SIP is constructed by a linear combination of normalised structural and contextual similarities as follows:

Input P: Adjacency Matrix of n*n

A: Attribute vector of $\sum_{i=1}^{m} d_i$

λ : Balancing factor

t: Total steps in signaling propagation

Output SIP: Matrix of n*n

```
1: function Transition_Algorithm
2: begin
3:     S, Ŝ, Ĉ, SIP : Matrix of n*n
4:     I: identity Matrix of n*n
5:     S=pow(P+I,t)   // using Eq.2
6:     for j=1 to n
7:         sum= Σ_{h=1}^{n} S_hj
7:         for i=1 to n
8:           begin
9:             Ŝ_ij = S_ij / sum        // using Eq.3
10:            Ĉ_ij = Min(i,j) / Max(i,j)   // using Eq.4
11:            SIP_ij = λ×Ŝ_ij + (1−λ)×Ĉ_ij   // using Eq.5
12:          end
13: end
```

Figure 3. Transition Algorithm.

$$\mathrm{SIP}_{ij} = \lambda \times \hat{S}_{ij} + (1 - \lambda) \times \hat{C}_{ij} \tag{5}$$

where SIP_{ij} computes integrated similarities between node i and j.

The transition algorithm is shown in Figure 3.

where *pow(…)* computes a sparse matrix multiplication. Its time complexity is $O(t(d+1)n^2)$ (in implementation $t = 3$ and $d =$ average degree of the given graph). Note that the function is not time-consuming practically because of graph sparseness. *Min(…)* and *Max(…)* are a numerator and a denominator of Equation (4), respectively. Their time complexity is $O(\sum_{i=1}^{m} d_i)$. The complexity of the transition_algorithem function is $O(t(d+1)n^2 + n(n + n \times 2)) = O((t(d+1) + 2 + 1)n^2)$.

4.2. Parameter determination

Two very important factors of partitioning algorithms are the number of clusters and key initial seeds. In this study, an innovative heuristic is introduced in order to determine and select the number of clusters and key initial seeds. We assume that every cluster is conducted by a node as a header. The header has high influence in its cluster because of its high local centrality. The high influential nodes are usually surrounded by low influential nodes. Thus, a high centrality node which is surrounded with low centrality nodes is more likely to be selected as a key initial seed. In order to define a neighbourhood for every node, k-nearest neighbour of a header is adopted. k is the only parameter of the proposed method which has to be determined in advance. The k-neighbourhood of node n, which is denoted by $N^k(n)$, is defined as follows:

$$N^k(n) = \{v \in V | dist(n, v) < dist(n, v^k)\} \tag{6}$$

Input: *SIP: Matrix // normalized Spatial Integrated Platform matrix*

Output: *Seeds: Set // contains the highest central nodes*
Count: Integer // the number of the clusters

1: ***function*** *seedFinder*
2: *Seeds: Set ={}*
3: *Count: Integer =0*
4: ***begin***
5: *Pr: Array // Pr.size= SIP.cols*
7: ***for each*** *col of SIP*
8: ***begin*** *//first for*
9: $\Pr_i = \frac{\sum_j SIP_{ij}}{SIP.cols}$
10: ***end***
11: *Pr=**Sort**(Pr)*
12: *Seeds.**add**(*$s_1 \in \Pr$*)*
13: *Count=Count+1;*
14: ***for i=2 to n*** *// second for*
15: ***begin***
16: $s_i \in \Pr$
17: ***if*** $s_i \notin \bigcup_{s_j \in Seeds} N^K(s_j)$ ***then***
18: *Seeds.**add**(si)*
19: *Count=Count+1;*
20: ***end***
21: ***end***
22: ***end***

Figure 4. The seed Finder Algorithm.

where *dist(…)* calculates the Elucidation distance between two nodes and v^k is k-nearest neighbour of node *n*.

The centrality of each node is calculated by a simple heuristic in SIP based on the density of the nodes. The density of a node is computed by summing the weights of the output edges of the given node.

Although high central nodes are more important and more likely to be chosen as seeds, choosing two high central nodes located within the neighbourhood of each other is not good idea. Here, the highest one is chosen as a seed and the lowest one is ignored. The *seedFinder* algorithm is implemented as follows:

where *Sort(.)* function sorts a given array via a HeapSort method in descending order which has *O(nlogn)* time complexity. The overall complexity of the *seedFinder* algorithm is $O(n^2 + nlogn + skn)$ where *s* is the average number of seeds and *k* determines a k-nearest neighbour. The space complexity of the algorithm is $O(n^2)$.

4.3 Partitioning algorithm

The objective is maximising intra-cluster and minimising inter-cluster similarities. In order to introduce an adapted partitioning algorithm, two important issues must be addressed. First one is how to assign each item to a cluster and last one is how to define an objective function to achieve high structural and contextual similarities within clusters and low structural and contextual similarities between clusters. To answer the former, an adapted partitioning algorithm is introduced based on the k-Medoid algorithm (Kaufman, 1987). The k-Medoid algorithm is an extended version of k-Means clustering algorithm that chooses medoids as seeds and works on medoids' distance. The proposed k-Medoid-based partitioning algorithm defines medoids by key initial seeds derived from the *seedfinder* algorithm. During each iteration, seeds are re-determined and points are re-assigned to clusters. The process is repeated until seeds are not changed for some iteration.

To answer the last question, density and entropy scores are employed to calculate structural and contextual similarities, respectively. Therefore, the objective function is computed as follows:

$$O_f = \lambda \times O_{str} + (1-\lambda)O_{con} = \lambda D\Big(\big[G^j\big]_{j=1}^{j=k}\Big) + (1-\lambda) * 1/E\Big(\big[G^j\big]_{j=1}^{j=k}\Big) \tag{10}$$

where λ is the balancing factor, $D(.)$ is a density function used to calculate structural similarities and $E(.)$ is a entropy function used to calculate contextual similarities.

The density function represents how clusters are tightly packed. Equation (11) computes the overall density score of the partitioned graph G.

$$D\Big(\big[G^j(V^j,E^j)\big]_{j=1}^{j=k}\Big) = \sum_{j=1}^{k} \frac{\sum\limits_{<v_p,v_q>\in E^j} C(<v_p,v_q>)}{\sum\limits_{<v_p,v_q>\in E} C(<v_p,v_q>)} = \frac{1}{\sum\limits_{<v_p,v_q>\in E} C(<v_p,v_q>)} \times \sum_{j=1}^{k} \sum_{<v_p,v_q>\in E^j} C(<v_p,v_q>) \tag{11}$$

where $C(.)$ is cost function of a given graph.

Also the overall cluster entropy is utilised to calculate relevancy of nodes of a given graph with respect to attribute values (Equation 12).

$$E\Big(\big[G^j(V^j,E^j,A^j)\big]_{j=1}^{j=k}\Big) = \frac{1}{k}\sum_{j=1}^{k} E^C(G^j(V^j,E^j,A^j)) \tag{12}$$

$$E^C(G^j(V^j,E^j,A^j)) = -\frac{1}{m}\sum_{q=1}^{m}\sum_{a_q\in d_q} p(a_q,V^j)\log p(a_q,V^j) \tag{13}$$

$$p(a_q,V^j) = \frac{|v_h \in V^j | a_q(h) = a_q|}{|V^j|} \tag{14}$$

The more nodes with same attributes in clusters, the less the overall entropy. Density and entropy values lie in the interval of [0, 1].

5. Experimental study

The performance of the proposed method is evaluated by extensive experiments. Three main evaluation scenarios are followed: (1) investigating the proposed method's convergence, (2) investigating the impact of parameters over clusters' quality as well as running time of the proposed algorithm and (3) density and entropy analyses of the proposed method when comparing with state-of-the-art clustering algorithms.

5.1. Data

Two real-life data-sets, called Political Blogs data-set (PBLOG)[1] (Nawaz et al., 2015) and DBLP data-set (Nawaz et al., 2015), are utilised to evaluate the performance of the proposed method. Table[2] 1 reports some statistics about them.

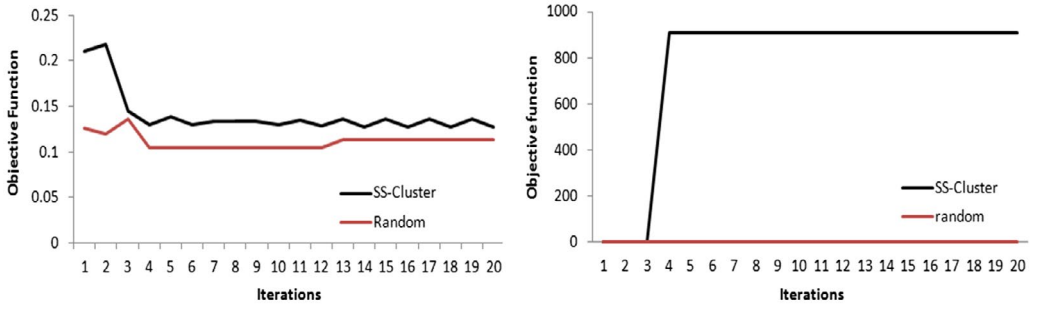
5.2. Configuration

The experiments have been performed on an Intel ® Core™ with 5 cores @3.2 GHz and 8 GB main memory. The algorithms have been implemented in Java. The proposed method, SS-Cluster, has been compared with four state-of-the-art clustering algorithms, SA-cluster, S-cluster, W-cluster and KSNAP. SA-Cluster has been introduced by Cheng et al. (Cheng et al., 2011) and considered both structural and contextual aspects of networks. Nodes' closeness has been calculated by random walk in S-Cluster

Table 1. Some statistics about data-sets.

	Description	#nodes	#edges	Attributes (values)
PBLOG	Political Weblog network	1490	19090	1. Political leaning (*liberal or Conservative*)
DBLP	Co-authorship network	5000	103844	1. Prolific (*low or prolific or high*) 2. Primary topic (*100 topics*)

Note: The topic modelling method introduced by [17].



(a) objective function variations per iterations over DBLP dataset

(b) objective function variations per iterations over PBLOG dataset

Figure 5. Objective function variations per iteration.

algorithm (Cheng et al., 2011), which considers just the structural aspect. W-Cluster has been introduced by Cheng et al. while both aspects are combined by a linear combination (Cheng et al., 2011). KSNAP has been introduced by Tian et al. (Tian et al., 2008). It collects nodes with same attributes in the same cluster. In order to compare performance and effectiveness of the algorithms, the density and entropy metrics have been computed, respectively, by Equations (11) and (12).

5.3. Algorithm convergence

Figure 5 depicts the objective function per iteration of SS-Cluster with/without the *seedFinder* algorithm over the DBLP and PBLOG data-sets. SS-cluster without the *seedFinder* algorithm uses a random seed selection method. For both evaluation tasks, Max-iteration = 20, $\lambda = 0.5$ and $N = 100$.

As shown in Figure 5, it is observed that the maximum objective values are achieved in 2 and 4 iterations for the DBLP and PBLOG, respectively. It is implies that determining key initial seeds based on the *seedFinder* algorithm helps the clustering algorithm to achieve the best results during few iterations.

5.4. Parameters evaluation

Three main parameters are analysed as follows: N, N^{th} nearest neighbour, is employed in order to compute local neighbourhood nodes, λ is a balancing factor of structural and contextual aspects and *Max-seeds* is used to determine the maximum number of seeds. Figure 6 illustrates the impact of λ on the quality of the clusters where $N = 130$ and *Max-seeds* = 10.

Figure 7 and Figure 8 demonstrate the impact of N on the quality of clusters over the PBLOG and DBLP data-sets, respectively. (Here $\lambda = 0.5$, *Max_Seeds*(PBLOG)=3 and *Max-Seeds*(DBLP)=10).

As shown in Figures 7 and 8, it is observed that the maximum density and entropy values are achieved by $N = 130$ in both data-sets. The best point, which is a point with high density and low entropy values, is achieved here at $N = 60$ for both data-sets.

Figures 9 and 10 demonstrate the impact of the maximum number of seeds on the quality of the clusters over PBLOG and DBLP data-sets, respectively (Here $\lambda = 0.5$, N (PBLOG and DBLP) = 100).

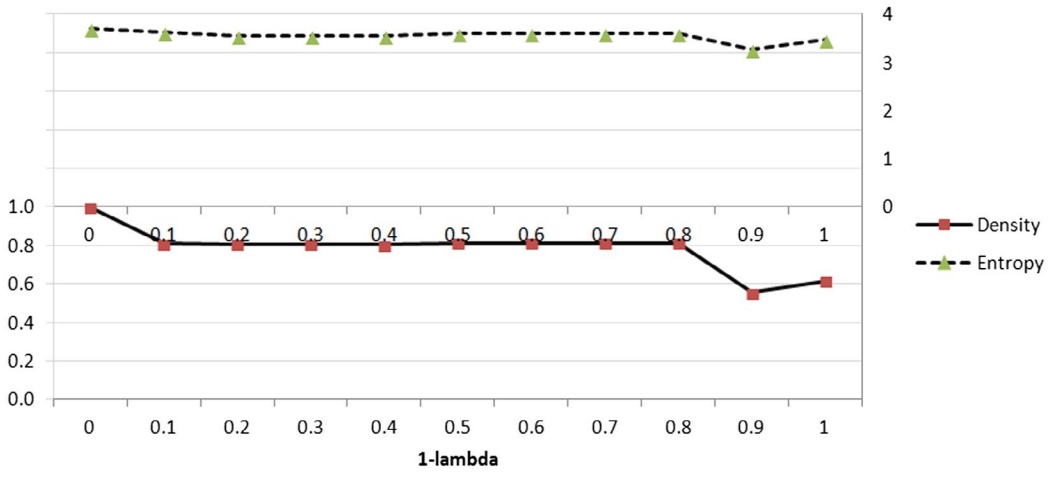


Figure 6. Impact of the balancing factor on density and entropy over the DBLP data-set.

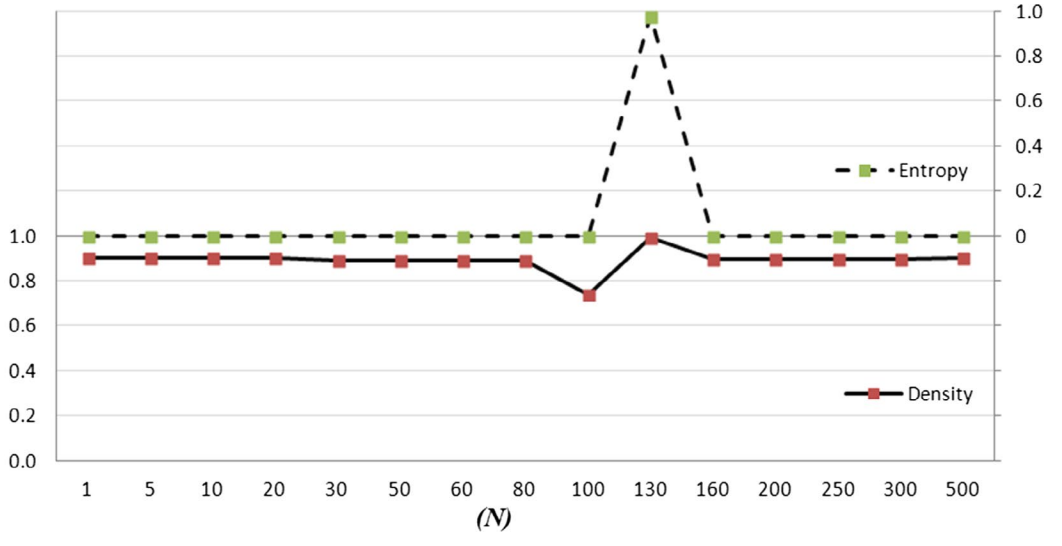


Figure 7. Impact of *N* on density and entropy measures (PBLOG).

As shown in Figure 9 and 10, the best point of the SS-Cluster (for the PBLOG data-set) is achieved at *Max-seeds* = 5 with density value = 0.87 and entropy value = 0.0018. Also the best point of the SS-Cluster (for the DBLP data-set) is achieved at *Max-seeds* = 10 with density value = 0.78 and entropy value = 3.59.

5.5. Running time

N is the most effective factor of time complexity of the proposed algorithm. Figure 11 illustrates the running time of the SS-Cluster with regard to *N* variation over the DBLP data-set.

5.6. Cluster performance comparison

In order to compare the SS-Cluster with other state-of-the-art clustering algorithms, several evaluation tasks have been performed on the PBLOG and DBLP data-sets. Here λ = 0.5 (for SA-Cluster, W-Cluster

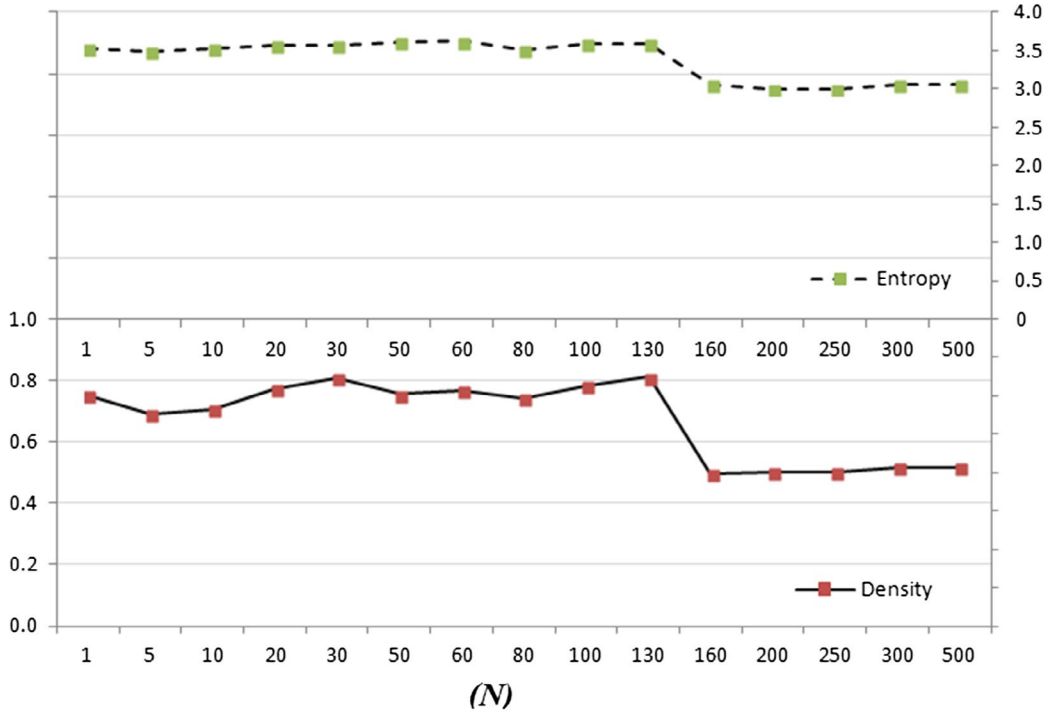


Figure 8. Impact of *N* on density and entropy measures (DBLP).

and SS-Cluster) and $N = 100$ (for SS-Cluster). Note that all algorithms need to determine the number of clusters in advance, whereas the SS-Cluster works with the maximum number of clusters. The exact number of clusters is determined by the *seedFinder* algorithm with help of N^{th} nearest neighbour. Figure 12 illustrates the density and entropy analyses' results over the PBLOG data-set.

On average, the SS-Cluster achieves (+0.2 and + 0.13) | (+0.45 and + 0.62) points improvements than the SA-Cluster | W-Cluster algorithms, respectively, on the density and entropy measures. As Figure 12 shows, it is observed that the SS-Cluster generates the densest as well as the most homogeneous clusters with regard to different number of clusters. Given that the SA-Cluster and W-Cluster algorithms consider both structural and contextual aspects, it implies that the SS-Cluster works better than the SA-Cluster and the W-Cluster on the PBLOG data-set.

Also, on average, the SS-Cluster achieves (+0.11 and + 0.71) | (+0.47 and + 0.00) points improvements than S-Cluster | KSNAP algorithms, respectively, on density and entropy measures. Given that the S-Cluster algorithm considers only structural aspects, it is observed that the proposed algorithm generates the densest as well as the most homogeneous clusters for all cases except the number of cluster equals 3. The KSNAP algorithm considers only the contextual aspect. The clusters generated by both KSNAL and SS-Cluster algorithms have the same homogeneity while clusters generated by the SS-Cluster algorithms are denser than clusters generated by the KSNAP algorithm. Generally, the results illustrate the superiorities of the SS-Cluster than other three well-known and popular algorithms in terms of density and entropy measures over the PBLOG data-set. Figure 13 demonstrates the density and entropy analyses' results over the DBLP data-set.

On average, the SS-Cluster achieves (+0.01) | (+0.01) | (+0.28) points improvements than SA-Cluster | W-Cluster | S-Cluster algorithms, on density measure. As Figure 13(a) shows, it is observed that the SS-Cluster generates the densest clusters with regard to different number of clusters. As Figure 13(b) shows, the SS-Cluster achieves (+0.34) points improvements than the SA-Cluster algorithm, on entropy

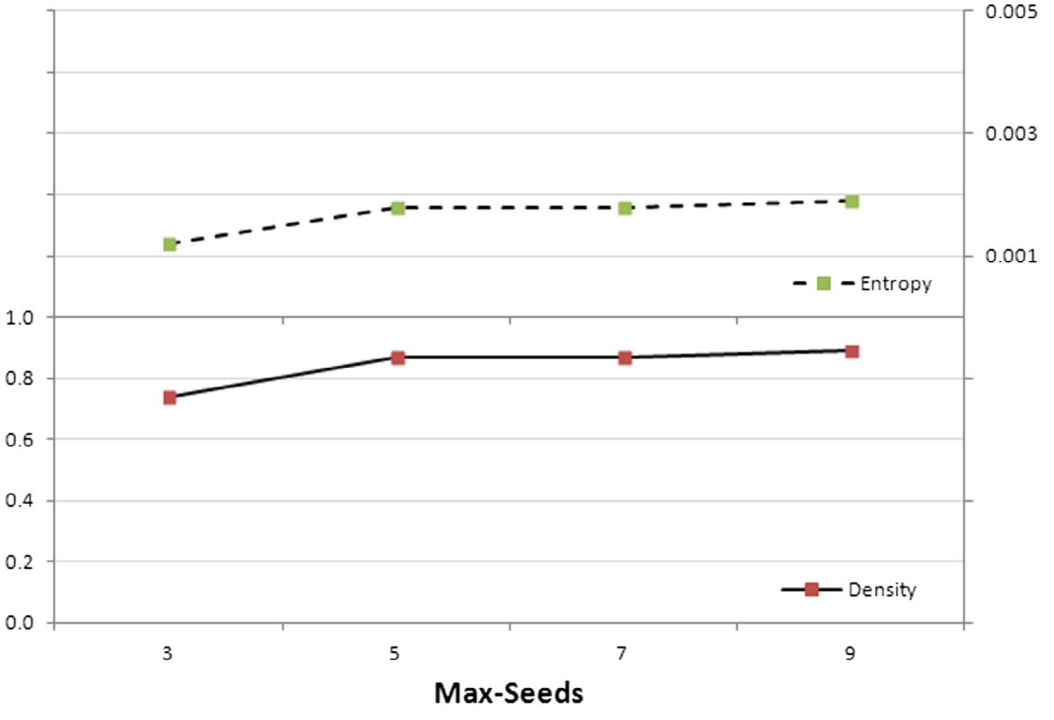


Figure 9. Impact of the Max-seed on density and entropy measures (PBLOG).

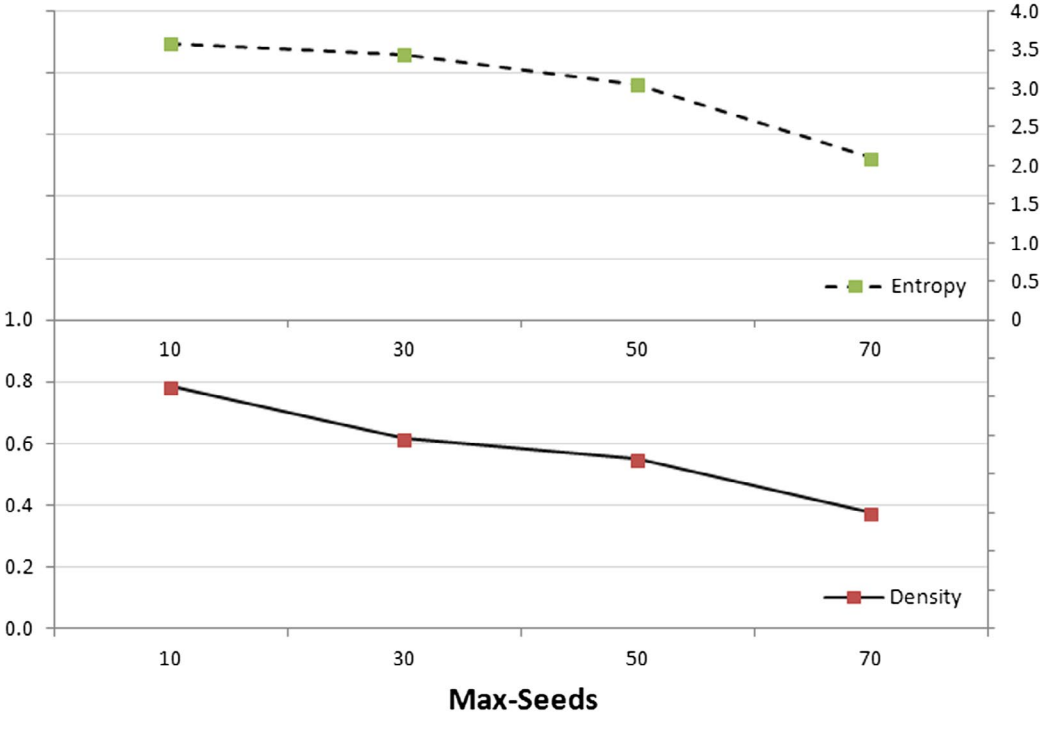


Figure 10. Impact of Max-seeds on density and entropy measures (DBLP).

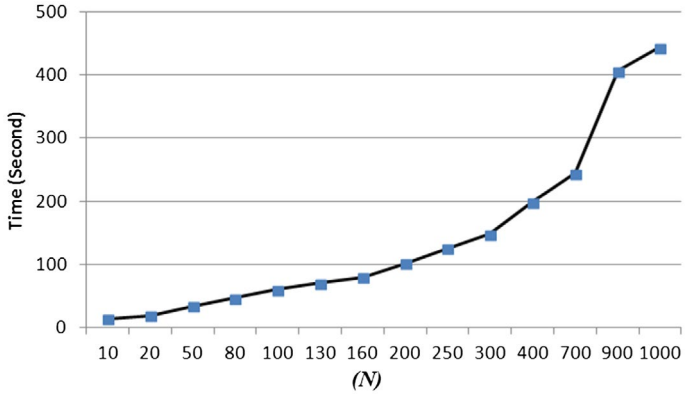
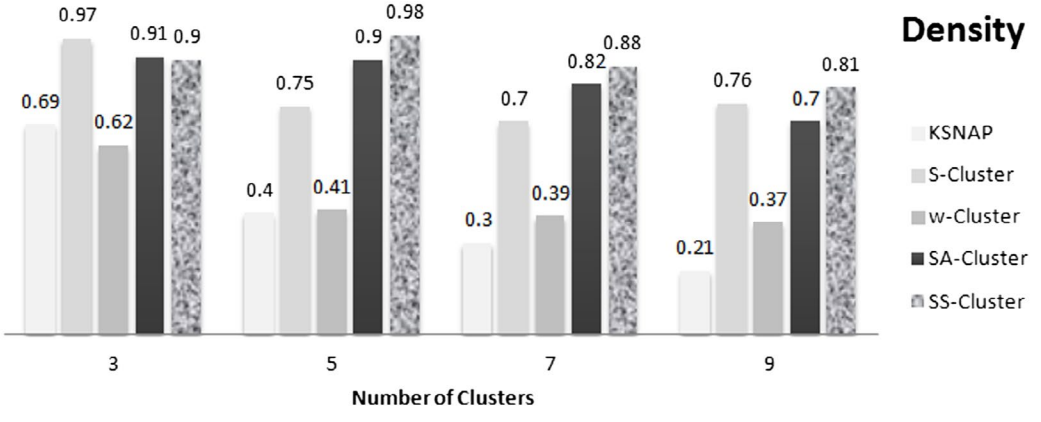
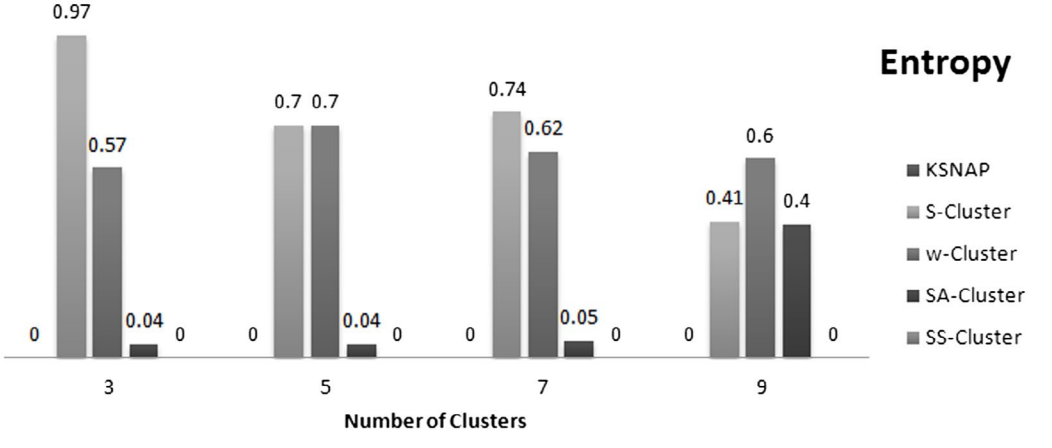


Figure 11. Time complexity.

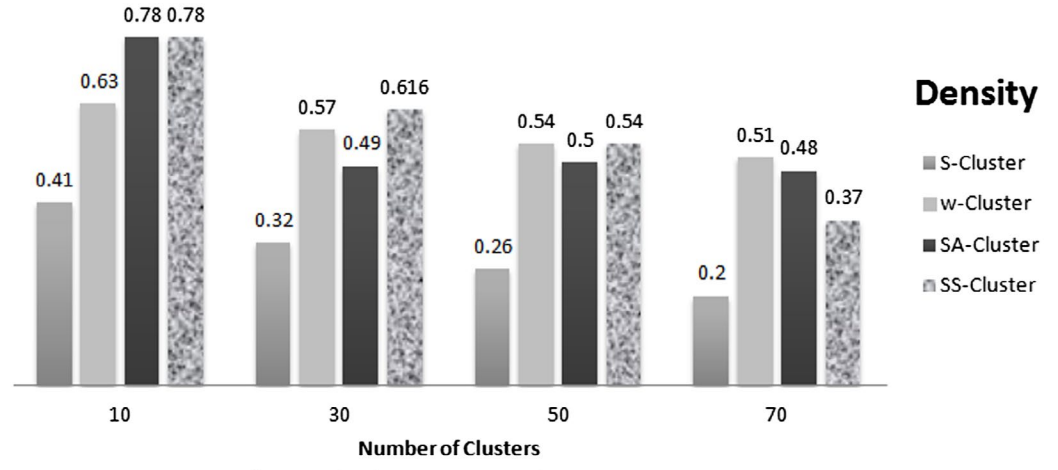


(a) Density analysis on PBLOG dataset

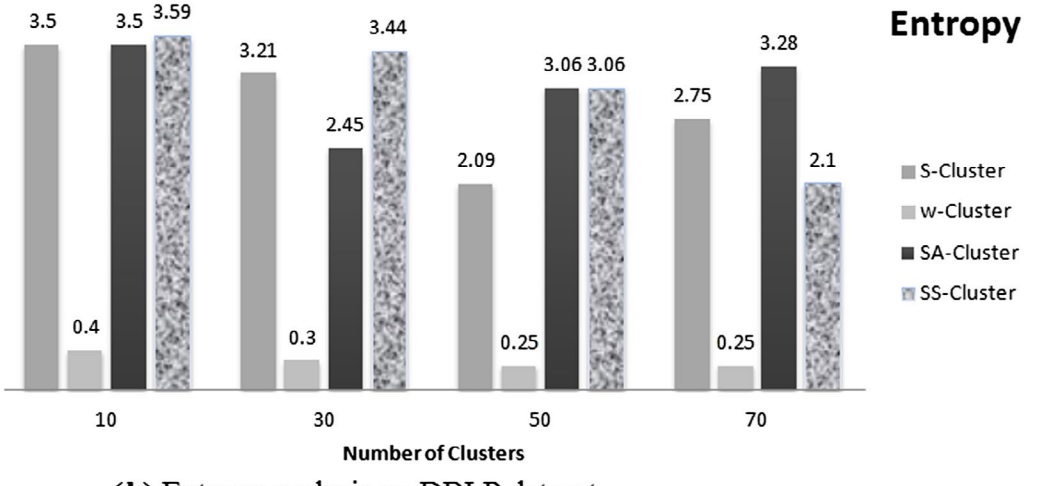


(b) Entropy analysis on PBLOG dataset

Figure 12. Density and entropy analyses on the PBLOG data-set.



(a) Density analysis on DBLP dataset



(b) Entropy analysis on DBLP dataset

Figure 13. Density and entropy analyses on the DBLP data-set.

measure, whereas the S-Cluster is better than the SS-Cluster (-0.17 points improvements) in terms of the entropy measure. The improvement is negligible.

6. Conclusion

In this study, an effective graph clustering method is presented, which is aimed at finding high density clusters with homogenous nodes in terms of node attributes. The structural and contextual aspects of the attributed graphs are modelled by signal similarity and weighted Jaccard similarity, respectively, and subsequently integrated into a spatial integrated platform. In this space, our method tries to maximise intra-cluster similarity and minimise inter-cluster similarity. The quality of the clustering results is evaluated by density and entropy metrics. Our method achieves a better performance gain over available algorithms. The experimental study illustrates competitive results in terms of cluster quality. Also, our method works in polynomial time and it is easy to scalable for mid-range and wide-range networks.

Notes

1. Http://www-personal.umich.edu/~mejn/netdata/
2. Refer to [10] for more details.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Bothorel, C., Cruz, J. D., Magnani, M., & Micenkova, B. (2015). Clustering attributed graphs: Models, measures and methods. *Network Science, 3*(03), 408–444.

Cazabet, R., Takeda, H., Hamasaki, M., & Amblard, F. (2012). Using dynamic community detection to identify trends in user-generated content. *Social Network Analysis and Mining, 2*(4), 361–371.

Cheng, H., Zhou, Y., & Yu, J. X. (2011). Clustering large attributed graphs: A balance between structural and attribute similarities. *ACM Transactions on Knowledge Discovery from Data (TKDD), 5*(2), 12.

Dourisboure, Y., Geraci, F., & Pellegrini, M. (2007). *Extraction and classification of dense communities in the web.* Paper presented at the Proceedings of the 16th international conference on World Wide Web.

Farzi, S., & Dastjerdi, A. B. (2010). Leaf constrained minimal spanning trees solved by modified quantum-behaved particle swarm optimization. *Artificial Intelligence Review, 34*(1), 1–17.

Guimera, R., & Amaral, L. A. N. (2005). Functional cartography of complex metabolic networks. *Nature, 433*(7028), 895–900.

Hu, Y., Li, M., Zhang, P., Fan, Y., & Di, Z. (2008). Community detection by signaling on complex networks. *Physical Review E, 78*(1), 016115.

Ioffe, S. (2010). *Improved consistent sampling, weighted minhash and l1 sketching.* Paper presented at the Data Mining (ICDM), 2010 IEEE 10th International Conference on.

Kaufman, L., & Rousseeuw, P. (1987). *Clustering by means of medoids*. North-Holland.

Kianian, S., Khayyambashi, M. R., & Movahhedinia, N. (2016). Semantic community detection using label propagation algorithm. *Journal of Information Science, 42*(2), 166–178.

Konstantinidis, K., Papadopoulos, S., & Kompatsiaris, Y. (2017). Exploring Twitter communication dynamics with evolving community analysis. *PeerJ Computer Science, 3*, 107–115.

Li, Y., Jia, C., & Yu, J. (2015). A parameter-free community detection method based on centrality and dispersion of nodes in complex networks. *Physica A: Statistical Mechanics and its Applications, 438*, 321–334.

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

Newman, M. E. (2003). The structure and function of complex networks. *SIAM Review, 45*(2), 167–256.

Newman, M. E., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E, 69*(2), 113–121.

Popescu, M., Keller, J. M., & Mitchell, J. A. (2006). Fuzzy measures on the gene ontology for gene product similarity. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 3*(3), 263–274.

Ravasz, E., Somera, A. L., Mongru, D. A., Oltvai, Z. N., & Barabási, A.-L. (2002). Hierarchical organization of modularity in metabolic networks. *Science, 297*(5586), 1551–1555.

Ruan, Y., Fuhry, D., & Parthasarathy, S. (2013). *Efficient community detection in large networks using content and links.* Paper presented at the Proceedings of the 22nd international conference on World Wide Web.

Shi, J., & Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence, 22*(8), 888–905.

Tian, Y., Hankins, R. A., & Patel, J. M. (2008). *Efficient aggregation for graph summarization.* Paper presented at the Proceedings of the 2008 ACM SIGMOD international conference on Management of data.

Wilkinson, D. M., & Huberman, B. A. (2004). A method for finding communities of related genes. *Proceedings of the National Academy of Sciences, 101*(suppl 1), 5241–5248.

Xu, X., Yuruk, N., Feng, Z., & Schweiger, T. A. (2007). *Scan: A structural clustering algorithm for networks.* Paper presented at the Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining.

Xu, Z., Ke, Y., Wang, Y., Cheng, H., & Cheng, J. (2012). *A model-based approach to attributed graph clustering.* Paper presented at the Proceedings of the 2012 ACM SIGMOD international conference on management of data.

Zhou, Y., Cheng, H., & Yu, J. X. (2010). *Clustering large attributed graphs: An efficient incremental approach.* Paper presented at the Data Mining (ICDM), 2010 IEEE 10th International Conference on.