Collaborative Similarity Measure for Intra Graph Clustering

Waqas Nawaz, Young-Koo Lee, and Sungyoung Lee

Department of Computer Engineering, Kyung Hee University, Korea
{wicky786,yklee}@khu.ac.kr, sylee@oslab.khu.ac.kr

Abstract. Assorted networks have transpired for analysis and visualization, including social community network, biological network, sensor network and many other information networks. Prior approaches either focus on the topological structure or attribute likeness for graph clustering. A few recent methods constituting both aspects however cannot be scalable with elevated time complexity. In this paper, we have developed an intra-graph clustering strategy using collaborative similarity measure (IGC-CSM) which is comparatively scalable to medium scale graphs. In this approach, first the relationship intensity among vertices is calculated and then forms the clusters using k-Medoid framework. Empirical analysis is based on density and entropy, which depicts the efficiency of IGC-CSM algorithm without compromising on the quality of the clusters.

Keywords: Graph clustering, collaborative similarity, k-Medoid clustering, entropy, density, Jaccard similarity coefficient.

1 Introduction

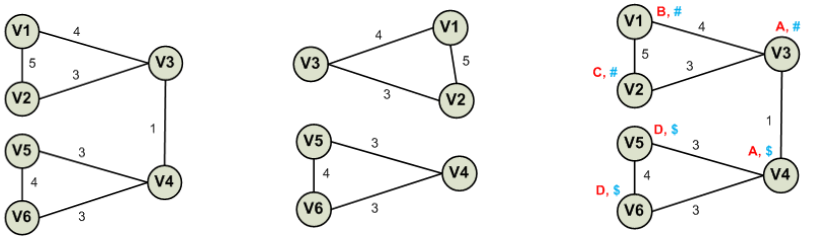
Graph as an expressive data structure is most widely used to model real life objects and their relationships in many application domains like social network, web, sensor network, and telecommunication. Graph clustering is very challenging and interesting research areas. Numerous researchers have focused different aspects of it, discussed in [1], [2], [3], and [4].

The key objective of most of the graph clustering algorithms is to identify strongly connected vertices within a graph with similar neighborhood. The vertices inside the sub-graph are highly cohesive while sparsely connected with other vertices of the graph. All the un-supervised clustering algorithms [5], [6], have the ability to find out natural number of clusters within the whole graph, whereas semi-supervised techniques [7] require extra information regarding the clusters such as number of clusters as an input parameter. We have used term node and vertex interchangeably throughout this paper.

Many researchers utilized the relational context of a network to discern interesting groups of entities, generally known as clusters [1]. In relational aspect, the connectivity among vertices or with similar neighborhood, harmonized topological structure, and characteristic resemblance are playing key role in graph clustering.

H. Yu et al. (Eds.): DASFAA Workshops 2012, LNCS 7240, pp. 204–215, 2012.

Classical applications of graph clustering include community detection in social networks, identification of functional related protein modules in large biological network (protein-protein interaction networks), etc. Numerous prior approaches either focus on the topological structure; attribute likeness or both for graph clustering [10], [11], and [14]. However they compromise either on quality or time.



(a) (b) (c)

Fig. 1. Graph Vertex Structures (a) connected, weighted and contextual graph, weighted graph (b) disconnected, weighted graph (c) connected, weighted, and multi-labeled

In this paper, we introduce the algorithm which belongs to a graph vertex clustering, an un-supervised method, whereas it requires the number of clusters as an input parameter prior to cluster the graph. It also provides an opportunity to control the connectivity or similarity strength among the clustered nodes. The deviation from existing state-of-the-art approaches in terms of the following facets: (a) a new two way node to node similarity measure is utilized which is more powerful because it considers both structural and contextual aspects (b) iterative clustering is applied which has low time complexity without compromising on cluster's quality, (c) also considers three basic scenarios for each vertex in graph as shown in Fig. 1. Key contributions of this work are as follows:

- A new graph vertex clustering strategy is proposed which is simple in nature. It can capture both structural and contextual similarities among nodes in the graph simultaneously.
- Similarity among the vertices is calculated once, which is symmetric and utilized successively.
- A customized k-Medoid framework is adopted for clustering.
- Disconnected graph nodes can be clustered together based on collaborative similarity measure.
- It is easily scalable for small and medium scale graphs because of its simplicity.

The rest of the paper is organized as follows. Section 2 briefly explains the related work. The proposed idea is elaborated in section 3. Subsequently its empirical analysis is carried out and then conclusion and future direction are drawn in section 5 followed by references.

2 Related Works

In traditional relational data clustering, the distance measure is primarily based on attribute similarity, e.g., Euclidian distance among two attribute vectors. On the other hand, in graph clustering the vertex closeness is measured on the basis of (i) connectivity, i.e., possible number of paths between two vertices, (ii) neighborhood similarity, i.e., number of common neighbors between them, or (iii) attribute or contextual similarity, i.e., number of common vertex's features or attributes.

In various real applications, both the vertex properties and the graph topological structure are important. For instance, in a social network, topological structure represents the relationship among different people in a group, while vertex properties describe the role of person. Most of the graph clustering and summarization approaches, mentioned in this and former section, only deal with one aspect and ignore the other.

Numerous existing graph clustering algorithms consider the topological structure of a graph so that every sub-graph attains the cohesive internal structure. It includes clustering based on normalized cut, max flow min-cut problem, structural density, modularity respectively [8], [9], [10]. Recently a new approach is proposed which deals with the attributes of an arbitrary pair of vertices [11], as a result vertices with the similar attribute values are grouped into respective partition or cluster.

A new probabilistic algorithm, i.e. Top Graph Clusters (TopGC) is devised in [12], which finds the best clique like clusters inside the large graphs. It works with both directed and undirected graph and support overlapping based on the given percentage. The space complexity is relatively high, because it is required to maintain the random permutations, hash table, and signatures for clustering. Time and search space complexity has been reduced by introducing the pruning phase. This approach cannot find the definite clusters in a graph which contains the topological as well as contextual information due to the fact that it only considers the structural facet.

Similarly, a three phase Transitive Node Similarity [13] approach has been developed for graph node clustering which consider only single similarity aspect. It is obvious that the search space get reduced incrementally because of the fact that already clustered nodes are not considered in the later iterations. This helps to reduce the space complexity and search space, which also helps to minimize the time complexity. In case of disconnected graph, each isolated component forms a separate cluster.

Conversely in [21], an efficient graph summarization technique based on two database-style operations has been proposed. The first operation, called SNAP, produces a summary graph by grouping nodes based on user-selected node attributes and relationships. Secondly in k-SNAP operation, it further allows users to control the resolutions of summaries.

Recently, a Unified distance measure is introduced in [14], [18] for graph node clustering. It captures both structural and attributes similarities simultaneously which is based on attribute augmented graph, by adding attribute nodes to the original graph and link all the respective nodes. The attribute augmented graph contains more edges in the graph, at least greater than the number of vertices ($> |V|$), which leads to more space complexity of the algorithm. Unified distance measure calculation (involves matrix multiplication) and clustering process is repeated, with iterative weight updates for convergence, which requires more computation time.

All of the methods in this section have the deficiency either with respect to cluster quality or algorithm time complexity. As you have seen none of them is able to fulfill both requirements simultaneously. In subsequent section, we have elaborated new technique which take care both aspects concurrently without any additional dependency.

3 An Intra Graph Clustering (IGC)

We need to construct a graph nodes clustering (i.e. intra graph clustering) method which can handle both structural and contextual similarity (collaborative in nature) in an efficient manner.

3.1 Problem Statement Formulation

An undirected, weighted, multi-labeled graph G = {V, E, W, A}, not necessarily connected, where V and A is the set of all the vertices and attributes respectively, E = $\{(v_i, v_j) \mid v_i, v_j \in V\}$ are the set of undirected links, and two vertices, v_i and v_j, may be connected with single link having cost $w_{ij} > 0$. Each vertex may contains one or more attributes, A = $\{a_1, a_2, \ldots a_m\}$. The set of possible values for any arbitrary attribute a_i is $Dom(a_i) = \{a_{i1}, a_{i2}, \ldots a_{inp}\}$ where $n_p = |Dom(a_i)|$. The total number of links of a vertex v_i is called the degree of v_i and is represented as $deg(v_i)$. If there is a direct link between any two vertices, e.g. v_i and v_j, in the graph, we call them as direct neighbor otherwise indirect neighbors in case of indirect link. Mostly used symbols along their descriptions are given in Table 1.

Intra graph clustering is the process of the relative partitioning the vertices of the graph into **k** disjoint sub-graphs where $G_i = \{V_i, E_i, W_i, A_i\}$ and $V = \bigcup_i^k (V_i)$ and for any $i \neq j$, $V_i \cap V_j = \emptyset$. The ultimate goal is twofold; attain high quality clusters, and time efficiency.

Table 1. Frequently used symbols and their brief description

Symbol	Description
G	A positive weighted, multi-label, and undirected graph
V	Set of the graph vertices
E	Set of edges/links in between nodes
N	No. of nodes inside the graph G
C	Set of the centroids in the graph
M	No. of possible attributes which can be associated with the node in the graph
K	No. of clusters to be found within the graph space
$SIM(X, Y)$	Similarity between two arbitrary sets X and Y, each set contains variable number of elements
$\|?\|$	Magnitude of the inside item
N_a	A neighborhood vector of a vertex a, all the nodes which are directly connected are considered to be in the neighborhood of the node
$deg(v_a)$	Number of links which are incident on the vertex a
w_{ab}	Weight associate on the link among node a and b
$v_a \leftrightarrow v_b$	If there exist a direct link between two vertices v_a and v_b
$v_a \cdots v_b$	If two vertices are indirectly connected with each other through an arbitrary path, when no direct link is present
$\mathrm{CSim}(v_a, v_b)$	Collaborative similarity measure between two vertices in the graph
$\mathcal{F}_{obj}$	Objective function
$c_i, cList[i]$	i^{th} cluster centroid, i^{th} centroid contained in cList

3.2 System Architecture

The proposed system architecture in Fig. 2 consists of two major components. Firstly, it requires estimating the relationship among nodes in the un-clustered graph. It will depict that how much relevance among nodes exists or correlated with each other topologically and contextually or semantically. During the clustering process the topological or contextual information remains unchanged so similarity among nodes is calculated once, in contrast to [14] and [13] which incrementally explore the search space and calculate the similarity as the new node encountered, and utilized multiple times in successive steps.

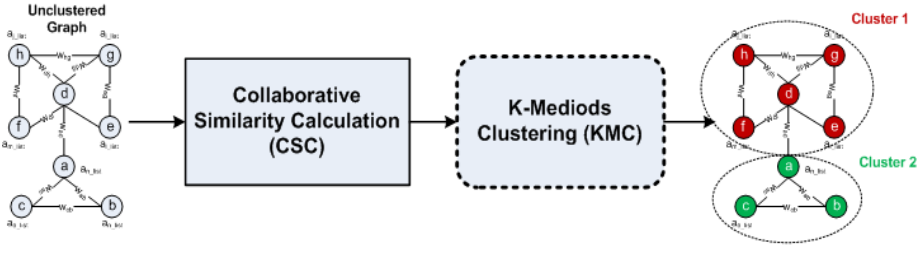


Fig. 2. Proposed Architecture

The similarity measures among nodes along the original graph information are the pre-requisites for the clustering component which is iterative in nature. Additionally, number of clusters k to be found is provided in advance. At the prior stage, we need influence factor α which can control the association among nodes based on topological structure and contextual relevance.

3.3 Collaborative Similarity Measure (CSM)

The similarity measure between graph vertices reflects the strength of their relationship or connectivity. In this approach, we utilize this strength for clustering similar vertices in one cluster and dissimilar to another. We have considered weighted, undirected, and multi-labeled vertex graphs, so relationship among two vertices can be found in the any of the following forms: (1) Directly Connected, (2) In-directly Connect, (3) Disconnected.

In case of direct connection, Fig. 3b, two vertices must share a single link in between them. When there is no direct link between two vertices, e.g. vi and vj, however from vertex vi we can reach on the other vertex vj by following an arbitrary links then both vertices are indirectly connected with each other, as shown in Fig. 3c. Thirdly, in absence of direct and indirect link, vertices are stated as disconnected. In this paper we assume that two vertices are connected only with single link.

$$\mathrm{SIM}(X,Y) = \frac{|X \cap Y|}{|X \cup Y|} = {}^{|X \cap Y|}\Big/_{|X| + |Y| - |X \cap Y|} \tag{1}$$

In literature, Jaccard similarity coefficient measure [16], given in Eq. (1), is generally acceptable and most widely used especially in data mining applications [17]. Due to

its simplicity, it has been utilized in many application areas to find out the relevance among the objects. Even though it is application independent, but in order to incorporate this similarity measure inside the proposed approach, we need to re-define the notions of the objects due to the presence of structural behavior. The important structural factor of the graph is the links between the vertices to form an association.

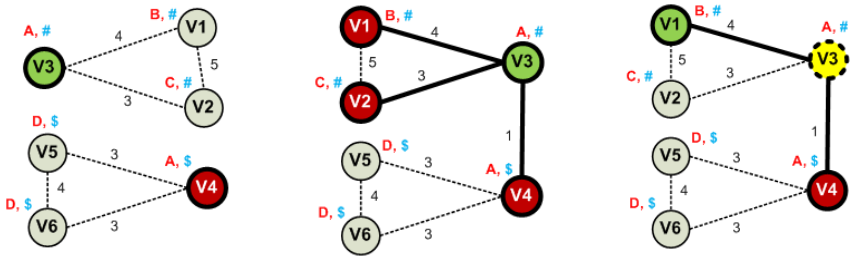


Fig. 3. Scenarios for similarity between source (green) and destination(red) nodes following some intermediate nodes (yellow) (a) No direct path exist (b) Directly connected (c) In-directly connected, shortest path

In Eq. (2), $\mathrm{SIM}(v_a, v_\mathrm{b})^{\mathrm{weighted}}_{\mathrm{struct}}$ represents the similarity between two vertices by exploiting the topological structure of the graph. It can capture all three possible scenarios, e.g. connected, indirectly connected, and disconnected. We need to define another similarity measure which can cover contextual information contained by vertices of the node. Here contextual information or semantics means that node can be participating in multiple contexts. Consider the example of social network, where each user is represented with a vertex and there are some associated contexts, like a person's role can be Teacher, Friend, Researcher, author/co-author of a paper.

$$SIM(v_a, v_b)^{weighted}_{struct} = \begin{cases} \dfrac{|N_a \cap N_b|}{|N_a \cup N_b|} * (w_{ab}), & v_a \leftrightarrow v_b \\ 0, & otherwise \end{cases} \quad (2)$$

Our strategy is equally acceptable for weighted and un-weighted graphs. The weights on edges along the path are considered explicitly which is not the case is in [13]. It will enhance the intensity of the relation among nodes based on the weights associated with edges. If two nodes are sharing edges with high weight will obviously have higher similarity as compared to low weight edge sharing nodes. In the absence of edge weights, constant value is expected to be associated with each link in the graph.

One of the key aspects of this approach is to consider contextual similarity which is defined in Eq. (3) along the structural cohesiveness of the nodes. Its importance is evident from the applications where the nodes emerge in different contexts. For example, in social network, the persons can be represented by nodes and edges among nodes reflect their relationships. Each person can have different roles or contexts like

occupation as student, doctor, engineer, and designer etc. Similarly in co-author network, each author is reflected with a node and if there are two authors of the same paper then a link is attach among those nodes in [14], [18]. Accordingly authors may have contributions in different areas of research at different time intervals; usually they used the labels or attribute associated with the corresponding nodes in the graph to reflect this information.

$$SIM(v_a, v_b)^{weighted}_{context} = \frac{\prod^{M}_{i=1,\, v_a \&\& \, v_b \leftarrow a_i} (w_{a_i})}{\prod^{M}_{j=1,\, v_a || \, v_b \leftarrow a_j} (w_{a_j})}, \quad v_a \leftrightarrow v_b \;||\; v_a \cdots v_b \tag{3}$$

$$\text{Collaborative Similarity} = \text{CSim}(v_a, v_b) = \begin{cases} \alpha * SIM(v_a, v_b)_{struct} + (1-\alpha) * SIM(v_a, v_b)_{context}, & v_a \leftrightarrow v_b \\ \prod_{i=1}^{q} CSim(v_{pi}, v_{pi+1}), & v_a \cdots v_b, \quad v_p \; is \; on \; path \; v_a \; and \; v_b \\ (1-\alpha) * SIM(v_a, v_b)_{context}, & otherwise \end{cases} \tag{4}$$

The semantics or attributes associated with the vertices which define the context of each vertex can also be prioritized by an associated weight w_{a_i} in Eq. (3). At the beginning of the algorithm, these values get initialized and remain fixed throughout all computations.

In order to find the similarity based on the set of shared features, Jaccard similarity coefficient measure gives us required functionality, elaborated in Eq. (1). Conclusively the similarity measure through which we consider three scenarios is given in Eq. (4). Both first and third scenarios are quite simple compared to the case when we have set of possible paths from source vertex to destination instead a direct or no link respectively. In order to extend the similarity for indirect path, we must utilize the desirable property, i.e. longer the path among two arbitrary vertices smaller the value of the similarity measure. Additionally, we should consider another property that the similarity measure value between two vertices along the entire path must be less than the intermediate vertices similarity.

Lemma 1 (**Transitivity):** Let $\mathfrak{p} = \{v_{\mathfrak{p}1}, v_{\mathfrak{p}2}, \ldots v_{\mathfrak{p}q+1}\}$ be a path from source vertex $v_{\mathfrak{p}1}$ to target vertex $v_{\mathfrak{p}q+1}$. Then for all the intermediate vertices i =1, 2…, q

$$\text{sim}(v_a, v_b) = \prod_{i=1}^{q} sim(v_{\mathfrak{p}i}, v_{\mathfrak{p}i+1}) \leq sim(v_{\mathfrak{p}i}, v_{\mathfrak{p}i+1})$$

Proof: It is based on the fact that the similarity value $\text{sim}(v_{\mathfrak{p}i}, v_{\mathfrak{p}i+1})$ lies in the interval [0, 1].

For any pair of vertices, many different paths may exist from initial vertex $v_{\mathfrak{i}}$ to final vertex $v_{\mathfrak{f}}$. The similarity value may vary based on the selected path. In order to avoid this variation and to be consistent, we have adopted shortest path between that pair of

vertices. Conceptually inside the dense community the similarity value is high because of transitivity property. However there is possibility to have smaller value even for shortest path as compared to other path options. This option is expected to have high computational cost. Mostly the shortest path produces larger value [13] unless an intermediate transition may cause significant similarity decrease, as given in Eq. (4).

3.4 Algorithm Details

The proposed method, Collaborative Similarity based Graph Clustering, can operate on undirected, weighted or un-weighted, and multi-attributed graph which requires two parameters as prior knowledge; number of clusters to be found even though it's unsupervised and factor which controls the importance of the topological structure and contextual associations. It does not require altering the structure of the graph. Systematically, it consists of two main components similarity calculation and clustering which is iterative in nature. The pseudo code is omitted due to space limit. For the proof of the concept, we have illustrated the similarity and clustering results in Table (2).

Initialization module considers the issues related to memory allocation, variable management for temporary and permanent storage. Un-clustered graph data is retrieved and stored in the main memory for later processing.

Similarity value estimation is core part of this algorithm because the subsequent processing is entirely dependent on the results of this module. Basically, the similarity value is anticipated among all possible pair of vertices in terms of their topological and behavioral or contextual resemblance, using Eq. (4). At this point, the most important factor which needs to be considered is the symmetry of the values due to undirected graph. For example if there are two vertices $v_a \; and \; v_b$ then $\mathrm{CSim}(v_a, v_b) = \mathrm{CSim}(v_b, v_a)$.

At the last stage, the partitioning of the graph vertices is done by utilizing the pre-calculated similarity values among vertices by employing the inherent features of the k-Medoid clustering infrastructure. In start vertices are randomly selected as the centroids (expected center points) or initial seeds to represent the hidden clusters. Then we associate neighboring vertices to the nearest centroids to make a partition based on their similarity distance. Finally, the quality of each cluster is analyzed based on the density and entropy. The ultimate goal of this iterative process is to approximate the objective function, given in Eq. (5).

The crucial part in this algorithm is to accurately measure the similarity value which can reflect the true relationship among vertices in the graph. Normally in a graph structure from one vertex to another there might be various paths available. In the heart of similarity calculation, the shortest path strategy has been incorporated to get rid of diverse paths dilemma.

As you can observe, the relationship among all possible pairs of vertices is estimated, so in accordance with this if we have V the total number of vertices in the graph then the time complexity for estimating the collaborative similarity (which consider both aspects simultaneously) among each pair of vertices will be O(|V|2). On the other hand, similar task, i.e. similarity estimation, in SA approach [14] costs us O (|V|3) due to matrix multiplication. However, shortest path calculation is done efficiently in order of O (|E| + |V| log |V|) using the Fibonacci heap [22].

Table 2. (**a**) Collaborative Similarity among vertices given in Fig. 3 using Eq. (4), (**b**) Clustering results on the graph (in Fig. 3) by varying number of clusters (K), quality of each measure is calculated using Density and Entropy

CSim(v_a, v_b)	**vertex**	V1	V2	V3	V4	V5	V6
	V1	1	2.67	1.17	0.20	0.18	0.18
	V2	2.67	1	0.92	0.15	0.14	0.14
	V3	1.17	0.92	1	0.17	0.15	0.15
	V4	0.2	0.15	0.17	1	0.92	0.92
	V5	0.18	0.14	0.15	0.92	1	2.5
	V6	0.18	0.14	0.15	0.92	2.5	1

(a)

K	Clustered Vertices	Density	Entropy
2	{V1,V2,V3},{V4,V5,V6}	0.42	0.133
3	{V1,V3},{V2},{V4,V5,V6}	0.28	0.084
4	{V5},{V6},{V4},{V1,V2,V3}	0.21	0.084

(b)

Where $\mathcal{F}_{\text{obj}}$ an objective function which depends on two sub functions $\mathrm{D}(\ldots)$, $\mathrm{E}(\ldots)$ as density and entropy respectively. The detail description for each function can be obtained from [14]. A well know k-Medoid clustering strategy is adopted which have the tendency to explore the best centroid candidates iteratively. At the beginning of the cluster phase, the centroids picked randomly, and in subsequent iterations the summated distance is computed using similarity measures and compared to select vertex with maximum value as new centroid. The termination criteria for clustering process is the decline of the objective function described in Eq. (5) or small improvement in the function value over subsequent iterations.

$$\mathcal{F}_{obj} = \max_{\mathrm{k}}\left(\left(\alpha * D\left(\{V_q\}_{q=1}^{k}\right)\right) - \left((1-\alpha) * E\left(\{V_q\}_{q=1}^{k}\right)\right)\right) \tag{5}$$

In our clustering scheme, the objective function $\mathcal{F}_{\text{obj}}$ needs to be maximized for clusters quality enhancement. High density corresponds to tight connection among vertices and low entropy ensures that most of the vertices in the cluster have similar contextual aspects or labels. In this respect higher objective function value always leads us towards better performance. If there is no improvement with respect to objective function in consecutive iterations it is supposed to be converged or terminated.

Influence factor α (alpha) is the controlling parameter which is exploited in Eq. (4) and Eq. (5) to balance the impact of both connectivity and semantics. Its value range is from 0 to 1. When alpha is 0 then vertices having similar attributes get clustered in one region irrespective of their interconnection and associated weights. However, value 1 has opposite impact to group densely connected regions of vertices instead their context. We have given an equal importance to both factors by taking its value 0.5. In this paper, its value remains fixed throughout the clustering process.

4 Empirical Analyses

The experiments were carried out on single 32-bit machine having 2.40GHz Intel dual core processor with 4GB main memory, and windows 7 as an operating system. The proposed method and SA-Cluster methods have been implemented using open source

matrix manipulation library JAMA[1] in Java. The comparison has been carried out among the following methods:

- ***IGC-CSM.*** The proposed efficient algorithm which considers both contextual and structural similarity without compromising its effectiveness.
- ***SA-Cluster.*** [14] It also considers both aspects but lack in scalability to huge graph due to iterative time consuming random walk strategy.
- ***S-Cluster.*** It is focused on topological structure of the graph.
- ***W-Cluster.*** It has combined both aspects, i.e. structural and attribute, however is based on random walk strategy which requires more computation [18].
- ***K-SNAP.*** It's a top down methodology which is concerned about attribute similarity [21].

4.1 Datasets

We have analyzed the proposed strategy on real and synthetic datasets. The political blogs network[2] dataset contains 1490 web blogs on United States politics with 19,090 hyperlinks between these blogs. This dataset contains one attribute associated with the web-blogs which has two possible values (in other words the domain of this attribute)

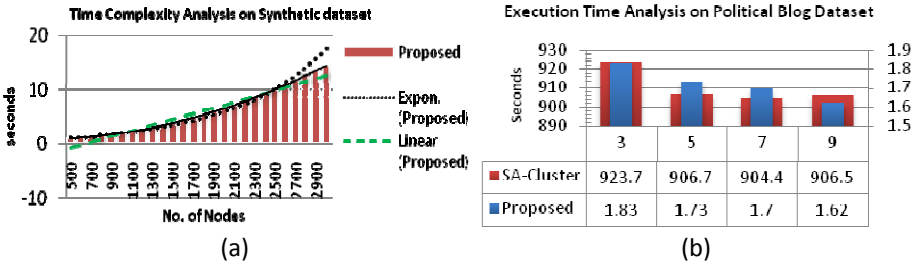


Fig. 4. (a) Graph size vs. time, (b) No. of Clusters vs. Time (seconds)

as either liberal or conservative. The real-life datasets are most of the times limited to small variation in attributes, links, and associated weights.

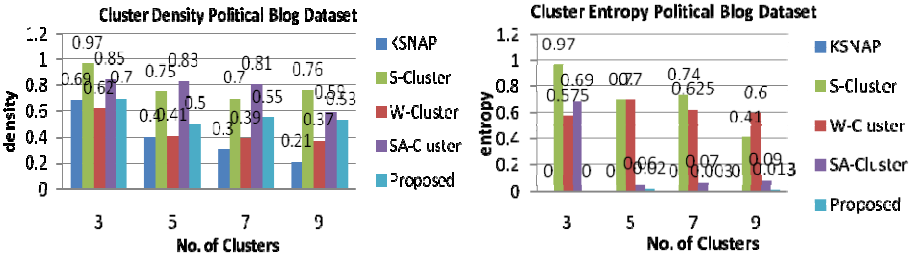
In all experiments α value set to 0.5. Our proposed method has the time complexity quadratic in nature which is acceptable for medium scale graphs as analyzed with respect to the graph size and number of clusters in Fig. 4. In order to develop a time efficient algorithm, it should not compromise on quality or effectiveness.

4.2 Cluster Evaluation Criteria

The structural and contextual quality of the clusters has been analyzed in terms of Density and Entropy respectively [14]. Combined effect can also be analyzed using F-measure [20], however due to space limit it has been omitted.

[1] http://math.nist.gov/javanumerics/jama
[2] http://www-personal.umich.edu/mejn/netdata

Strong connection among vertices can be easily analyzed by utilizing the density function. It is the ratio between number of edges present in a cluster and total number of edges in the whole graph. The ratios get accumulated for all clusters to evaluate the overall impact. Its values lie in the interval of [0, 1].One of the key aspects to measure the quality of the cluster is to determine the relevancy among vertices based upon their attributed nature.



(a) (b)

Fig. 5. (a) Number of Clusters vs. Density (**b**) Number of Clusters vs. Entropy

Fig. 5 (a) and (b) shows the competitive results against density and entropy respectively. Our algorithm maintains the quality in terms of density and entropy by varying number of clusters. It is obvious from Fig. 5 (b) that overall entropy difference is large which shows good clustering tendency.

5 Conclusions and Future Directions

In this paper, we study the problem of graph clustering which enables us to efficiently partition the vertices based on homogeneous characteristics in terms of context and topology. We have carefully designed the collaborative similarity measure to reflect the relational model among pair of vertices. Subsequently, well known k-Medoid clustering framework is adopted for iterative evolution of all the clusters. The quality estimation of each cluster is concurrently done based on density, and entropy measures. Experiments on synthetic and real datasets depict the time efficiency of the proposed methodology while sustaining the cluster quality.

Our idea works in polynomial time (i.e. quadratic in nature) however it's still hard to scale up for very huge datasets. The automatic adaptation of the tuning parameter, alpha, should also require extensive analysis which is beyond the scope of this paper.

References

1. Cook, D.J., Holder, L.B.: Mining Graph Data. Wiley and Sons (2006)
2. Drineas, P., Frieze, A., Kannan, R., Vempala, S., Vinay, V.: Clustering Large Graphs via the Singular Value Decomposition. Machine Learning 56(1-3) (2004)

3. Dongen, S.: Graph Clustering by Flow Simulation, Ph.D. thesis, University of Utrecht (2000)
4. Flake, G.W., Tarjan, R.E., Tsioutsiouliklis, K.: Graph Clustering and Minimum Cut Trees. Journal of Internet Mathematics 1(4), 385–408 (2003)
5. Newman, M.: Detecting Community Structure in Networks. The European Physics Journal B 38, 321–330 (2004)
6. Huang, X., Lai, W.: Clustering Graphs for Visualization via Node Similarities. Journal of Visual Languages and Computing 17, 225–253 (2006)
7. Anand, R., Reddy, C.K.: Graph-Based Clustering with Constraints. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part II. LNCS, vol. 6635, pp. 51–62. Springer, Heidelberg (2011)
8. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Analysis and Machine Intelligence 22(8), 888–905 (2000)
9. Xu, X., Yuruk, N., Feng, Z., Schweiger, T.A.J.: Scan: a structural clustering algorithm for networks. In: Proc. 2007 Int. Conf. Knowledge Discovery and Data Mining (KDD 2007), San Jose, CA, pp. 824–833 (August 2007)
10. Newman, M.E.J., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E 69, 026113 (2004)
11. Tian, Y., Hankins, R.A., Patel, J.M.: Efficient aggregation for graph summarization. In: Proc. 2008 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD 2008), Vancouver, Canada, pp. 567–580 (June 2008)
12. Marcopol, K., et al.: Scalable Discovery of Best Clusters on Large Graphs. Proceedings of VLDB Endowment 3(1) (2010)
13. Tiakas, E., et al.: Graph Node Clustering via Transitive Node Similarity. In: 14th Panhellenic Conference on Informatics (2010)
14. Zhou, Y., et al.: Graph Clustering Based on Structural/Attribute Similarities. In: Proceedings of VLDB Endowment, France (2009)
15. Larsen, B., Aone, C.: Fast and effective text mining using linear-time document clustering. In: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 16–22 (1999)
16. Jaccard, P.: Etude Comparative de la Distribution Floraledansune Portion des Alpes et des Jura. Sociètè Vaudoise des Sciences Naturelles 37, 547–579 (1901)
17. Everitt, B.: Cluster Analysis, 3rd edn. Edward Arnold, London (1993)
18. Cheng, H., et al.: Clustering Large Attributed Graphs: A Balance between Structural and Attribute Similarities. ACM Transaction on Knowledge Discovery from Data 5(2) (February 2011)
19. Fredman, M., Tarjan, R.: Fibonacci Heaps and their Uses in Improved Network Optimization Algorithms. Journal of ACM 34, 596–615 (1987)
20. Witsenburg, T., et al.: Improving the Accuracy of Similarity Measures by Using Link Information. In: International Symposium on Methodologies for Intelligent Systems, 9th edn., Poland (2011)
21. Tian, Y., Hankins, R.A., Patel, J.M.: Efficient aggregationfor graph summarization. In: Proc. 2008 ACM-SIGMOD Int.Conf. Management of Data (SIGMOD 2008), Vancouver, Canada, pp. 567–580 (June 2008)
22. Fredman, M., Tarjan, R.: Fibonacci Heaps and their Uses in ImprovedNetwork Optimization Algorithms. Journal of the ACM 34, 596–615 (1987)