Halim, Z., Sargana, H. M., Aadam, Uzma, and Waqas, M. (2021) Clustering of graphs using pseudo-guided random walk. *Journal of Computational Science*, 51, 101281. (doi: 10.1016/j.jocs.2020.101281)

There may be differences between this version and the published version. You are advised to consult the published version if you wish to cite from it.

http://eprints.gla.ac.uk/306712/

Deposited on 19 October 2023

Enlighten – Research publications by members of the University of Glasgow
http://eprints.gla.ac.uk

Clustering of Graphs Using Pseudo-Guided Random Walk

Abstract Clustering is an unsupervised learning task that models data as coherent groups. Multiple approaches have been proposed in the past to cluster large volumes of data. Graphs provide a logical mapping of many real-world datasets rich enough to reflect various peculiarities of numerous domains. Apart from *k*-means, *k*-medoid, and other well-known clustering algorithms, utilization of random walk-based approaches to cluster data is a prominent area of data mining research. Markov clustering algorithm and limited random walk-based clustering are the prominent techniques that utilize the concept of random walk. The main goal of this work is to address the task of clustering graphs using an efficient random walk-based method. A novel walk approach in a graph is presented here that determines the weight of the edges and the degree of the nodes. This information is utilized by the pseudo-guidance model to guide the random walk procedure. This work introduces the friends-of-friends concept during the random walk process so that the edges' weights are determined utilizing an inclusive criterion. This concept enables a random walk to be initiated from the highest degree node. The random walk continues until the walking agent cannot find any unvisited neighbor(s). The agent walks to its neighbors if it finds a weight of one or more, otherwise the agent's stopping criteria is met. The nodes visited in this walk form a cluster. Once a walk comes to halt, the visited nodes are removed from the original graph and the next walk starts in the remaining graph. This process continues until all nodes of the graph are traversed. The focus of this work remains random walk-based clustering of graphs. The proposed approach is evaluated using 18 real-world benchmark datasets utilizing six cluster validity indices, namely Davies-Bouldin Index (DBI), Dunn Index (DI), Silhouette Coefficient (SC), Calinski-Harabasz Index (CHI), modularity index, and normalized cut. This proposal is compared with seven closely related approaches from the same domain, namely, limited random walk, pairwise clustering, personalized page rank clustering, GAKH (genetic algorithm krill herd) graph clustering, mixing time of random walks, density-based clustering of large probabilistic graphs, and Walktrap. Experiments suggest better performance of this work based on the evaluation metrics.

Keywords Graph clustering, Random walk, Efficient clustering, Community detection

1. Introduction

Graphs provide a fundamental area of research in computing and mathematics having its history spanning over a century [1]. A graph typically consists of nodes representing independent entities and edges indicating the connectivity between them. There are many different types of graph with their specifications and provisions [2]. In the present age of technology, data is a fundamental asset for any organization. To reveal the best results for information processing or to have business insights using unsupervised/semi-supervised data, one has to perform pattern recognition. Two key tasks of pattern recognition include: classification and clustering. Where, classification is a supervised learning task and clustering being unsupervised [3]. The supervised learning utilizes training data for learning and later classifies unknown objects into the predefined categories. Whereas, the process of finding similar groups in the data with respect to some common features is called clustering. This comes under the domain of unsupervised learning. Clustering is different from classification majorly because of the lack of prior knowledge about the classes hidden/available in the data. Graphs can be treated as hierarchical structures connecting nodes through edges. Finding useful patterns or similar communities in a graph data is therefore referred to as graph clustering. Such clustering is to cover only those nodes/edges which have the

maximum relationship within a subgraph and a few inter-cluster connections. Graph-based data or ordinary data represented as a graph provide an essential digital representation for the algorithms to facilitate information processing.

In a transactional database system, the clustering result can be used to collect the relevant queries in a data dictionary for efficient information retrieval in the future. Previously, such work has been done for databases. For instance, Diwen et al. [4] proposed a solution to store the queries in a tree-like structure. Other researchers have proposed graph-based concepts to identify shortest paths. For the domain of bioinformatics, clustering of the graphs is done to identify gene expression data, protein-protein interaction, epidemic expansion, and viral spreading. For example, Enright et al. [5] worked on SIR (susceptible → infective → removed) disease. It is an epidemic disease found in dense population and has several types of infections. Their work uses graph clustering to identify epidemic disease with cluster-like structure. An electroencephalogram (EEG) is a brain signal captured using an EEG cap to observe disorders of brain's electrical pulse. Graph clustering can also be done in such type of data to find neural structures. Clustering the data extracted from social networking websites can help to identify the voting trend in elections, influence, and popularity of a person with maximum node degree. Apart from social networks, clustering can identify the latest trends in the stock market as well.

Previous works on clustering using random walk have overlooked the aspect of "guiding" the walk such that it forms better quality clusters without compromising on the randomness of the process. Past approaches either guide the random walker to an extent that it totally eliminates the underlying randomness of the process or the procedure is completed unguided. All this forces the methods to stuck in a local optimum. Additionally, the previous methods start the clustering from an arbitrary node in the graph. This has the limitation of prolonging the clustering process in reaching a better clustering formation. An initial calculated guess to start the cluster formation will not only result in reducing cluster formation time but will also enable to form clusters with higher cohesion. The current proposal addresses these limitations of the past works. Keeping in view the abovementioned domains where clustering has been utilized previously, this work focus on developing a clustering approach that utilizes graphs for data representation. Since clustering is an unsupervised task, the key challenge here is to produce clusters having strong intra-cluster similarity and higher inter-cluster dissimilarity. For the graph-based clustering approach, this work focuses on the formation of coherent groups through random walk. This is done by introducing the novel friends-of-friends concept as a guiding principal for the random walk to explore neighbours. Although various proposals can be found in the past that utilizes graphs to cluster the data, however, random walk is a technique yet to be fully explored for this task. Finding clusters through random walk has been previously done by Enright et al. [5] as Markov clustering algorithm and through limited random walk in Zhang et al. [6]. This work presents a novel approach in terms of neighbourhood exploration. The randomness in the walk is minimized by guiding the process towards more suited nodes to be explored. However, it is not completely eliminated. The proposed approach assigns higher significance to the graph node having maximum connectivity. This enables the algorithm to start clustering with the most influential node in the community. Since clustering is unsupervised, each clustering algorithm has its pros and cons. Random walk-based clustering can be efficient in terms of randomly moving to its neighbors by examining their importance.

The walking agent would move to its neighbour based on a probability, inflation, and modulation. There can be other factors incorporated into the approach to form better clusters. For instance, the additional feature adopted here is the friends-of-friends concept to grow the clusters using a pseudo-guidance mechanism. At the core of the current proposal is the friends-of-friends concept to guide the random walk process. Traditionally, to form clusters via a random walk, a node is arbitrarily added to the existing group based on it being a neighbor. However, the complete randomness of the process may cause a less suitable node to be added to the cluster. The key intuition here is to not only consider the existing edge (and its weight) between the cluster and the node under consideration for joining the group, but also to identify how many common neighbors exist between the two. Consider the cluster C_i and two nodes n_j and n_{j+1} to be deliberated for inclusion in C_i. If the sum of edge weights between the nodes under consideration is same for n_j and n_{j+1}, however n_j has more common neighbors (friends-of-friends) within C_i than n_{j+1}, then the node n_j is more suitable (based on commonalities) to be added to the cluster C_i in comparison to the node n_{j+1}.

1.1. Key contributions

This work presents a random walk-based clustering methods for the data representable as a graph. The random walk procedure is directed by a pseudo-guidance procedure. The procedure guides the random walker to consider visiting densely connected nodes first. The densely connected portions of the graph are identified based on the number of common neighbors between two nodes instead of relying on their raw degrees. The proposal is evaluated using 18 real-world benchmark datasets utilizing six cluster validity indices, namely Davies-Bouldin Index (DBI), Dunn Index (DI), Silhouette Coefficient (SC), Calinski-Harabasz Index (CHI), modularity index, and normalized cut. It is compared with seven closely related approaches from the same domain, namely, limited random walk, pairwise clustering, personalized page rank clustering, GAKH (genetic algorithm krill herd) graph clustering, mixing time of random walks, density-based clustering of large probabilistic graphs, and Walktrap. The approach proposed in this work can be utilized to identify clusters in diversified fields of study. Random walk-based clustering is a suitable solution for determining the best clusters in the data transformed into a graph. For example, this work can be used to find clusters in the data dictionary of production databases where queries are stored. A cluster can be formed by identifying the queries on the database. This is helpful when next time one tries to write the same query, it would be the part of same database which was loaded previously. In brief, the key contributions of this work include:

- Utilizing graphs to efficiently store and cluster the data
- A "guided" random walk-based approach to extract clusters
- Incorporation of the friends-of-friends concept to guide the clustering process towards better nodes to be explored
- Incorporating the commonality concept based on mutual friends to form clusters instead of relying on the raw node degree
- Avoiding any biases caused by nodes with a higher degree to form clusters

The rest of the paper is organized as follows. Section 2 covers the related work on clustering graphs. The section focuses on the clustering methods utilizing random walk. It also lists some related basic notions. Section 3 presents the proposed solution and explains its working procedure. Section 4 lists the detailed experiments and obtained results. All experiments are performed using benchmark datasets obtained from the UCI machine learning repository[1] and other such venues. Section 5 presents the statistical analysis of the obtained results. Discussion on the obtained results is covered in Section 6. The section also lists a few limitations of this work. Finally, Section 7 concludes this work with a few of the future directions.

2. Related work

Clustering is a key data mining/machine learning task forming bases for many higher-level objectives to be achieved by the modern computing systems. Various algorithms have been proposed for clustering data. Natural clusters are formed by the existence of common characteristics between members of the same group. Considering graphs, the most related links or edges will likely make a cluster of the same type. Various proposals have been presented in the past to cluster data. This section covers clustering approaches for graphs specially those utilizing the random walk. Recently, many approaches have been proposed to implement random walked-based clustering techniques [7,8,9,10]. Diversified works have been done for random walk due to the fact that the walking agent, at times, travels beyond the boundaries of the core neighbourhood. A typical random walk starting from a seed node is more likely to stay in the vicinity of the seed vertex forming a coherent group. Due to this reason, the random walk may work well for a typical clustering problem.

2.1. Clustering using random walk

Harel et al. [11] introduce a random walk-based technique in their work. They propose two operators to change the edge weight in such a way that the strength of inter-cluster edges is reduced and the intra-cluster edges increases. Iteratively, the inter-cluster edge weight decreases approaching to zero to differentiate them with other clusters. Their technique is somewhat similar to the work in Girvan et al. [12] where edge weight between two nodes is utilized for grouping.

Zhou et al. [13] uses random walk between two nodes to group the data represented as a graph. They find the distance d_{ij} between node i and node j as an average distance between two nodes in terms of number of edges connecting them. The nodes in the same cluster will be closely related to each other. The authors define two types of vertices in a cluster known as *global attractor* and *local attractor*. The *global attractors* are defined as the closely related neighbours, i.e., anywhere in the closest domain of vertex i. Whereas, the *local attractors* are defined as the immediate neighbours of vertex i. Using this concept, two types of communities are extracted each containing subgraphs of their own. The work in [14] adopts biased random walker technique. This work finds biasness as the walking agent should visit high degree neighbours frequently. For this, a proximity index is used to find the biased neighbouring node. The approach can be classified as a hierarchical clustering method. Hagen et al. [15] uses random walk to cluster circuits in a Very-Large-Scale integration (VLSI) design problem

[1] https://archive.ics.uci.edu/ml/index.php

through random walk. They utilize the concept of cycles to find clusters. The worst case time complexity of their solution is $O(n^3)$ whereas, the space complexity is $O(n^2)$.

Pons et al. [16] uses a different measuring method for computing distance between nodes through random walk. The distance is calculated between two nodes using a fixed number of steps utilizing the probability matrix. The number of steps is large enough to traverse the substantial portion of the graph. However, for a larger number of steps, the stopping criterion is met. Hu et al. [17] finds the neighbour vertex through a signaling method which resembles diffusion. Initially, on the first move, source vertex transmits unit signal to its neighbours. Next, these neighbours send many unit signals to their neighbours. This process continues until a threshold is reached. The procedure is repeated from the next node by considering the current node as the source. Weinan et al. [18,19] uses Markova chain method to describe a random walk procedure on the subgraph. The vertices of this subgraph represent the clusters in the original graph. This process gives a suitable determining method to extract coherent clusters. Van Dongen et al. [20] also uses the Markov clustering algorithm to find clusters. Their method finds movement of the random walker. The Markov clustering algorithm is the baseline approach that utilizes random walk through a probability matrix for clustering. Probability matrix is obtained by multiplying the adjacency matrix with the inverse degree matrix. The sum of each column of the transition matrix is computed using Eq. (1).

$$P=A\,DG^{-1} \tag{1}$$

Where, P is the transition probability matrix, A is the adjacency matrix, and DG is the degree matrix. Degree matrix is the diagonal matrix with degree of each vertex mentioned at the diagonal. The probability matrix sums-up all columns to one.

Markov clustering algorithm (MCL) [6] is based on the idea of Markov chain, however, here transition is done with probability matrix only. Graph clustering using random walk is flanked by this algorithm. A walk starts from seed vertex and moves to its neighbour if there is a higher probability between them. Markov clustering algorithm replicates the stream within the cluster. It uses inflation and normalization to increase the stream within one cluster and reduces the flow between different clusters. MCL procedure is time-inhomogeneous for which the transition matrix varies over time. The MCL algorithm starts the random walk from all vertices simultaneously, i.e., there are n agents walking in the graph at the same time. The walk can only continue after all agents have completed a walking step and the resulting probability matrix has been inflated and normalized. Each iteration of the algorithm has two steps, one is expansion and the other one is inflation. Expansion is utilized for computing probability and inflation is used for the weight enhancement between pairs of nodes.

Zhang et al. [6] uses Limited Random Walk (LRW) algorithm utilizing the concepts presented in [20]. Inflation and normalization are applied during each step of the random walk. Applying inflation operator to the matrix causes it to increase the existing small differences. Their work is time-homogeneous Markov chain procedure. They start the walk from a single seed vertex and perform inflation on the probability values of each walking step rather than from many different vertices like MCL. The approach has an advantage that it eliminates the need of multiple walks since a single seed vertex is sufficient to explore the vertices within its

vicinity. This procedure is suitable for local graph clustering problems. However, if there is a need to start multiple walks, these can be computed by the parallel computing paradigm. He et al. [21] extend the work presented in [22] by fusing it with the *k*-means algorithm. Meila et al. and Shi et al. [23] perform experiments to cut the graph and use transition probabilities and the stationary distribution for a walking agent to explore it. Authors link the mathematical transition probabilities with their proposal. Min-cut intuition approach can be used in many kinds of graph like directed, undirected, weighted, and unweighted with an effective maximum flow algorithm.

Auber et al. [24] assume similarity matrix from the vertex set to cluster a graph. They define a discrete formula to turn these values into discrete set. Afterwards, convolution is applied on the discrete set to cluster the graph. This process is iterated to form a cluster hierarchy. Yang et al. [25] perform graph clustering through random walk. They use *k*-step transition through a probability matrix to find a connection between two nodes. According to the *k*-step transition, adjacency matrix is stored to find a cut point in the matrix. This cut is then used to find a point to divide the graph into two sets. The process continues until all elements of a diagonal matrix get traversed. This process usually comes under the domain of top-down approach.

Another approach somewhat similar to a merging point method in space is presented in [26]. The key intuition is to merge two points that are closely related to each other according to their characteristics. This process of merging the clusters continues until all points are traversed or a stopping criteria is met. Such method is generally called pairwise clustering in the literature. Zhang et al. [6] utilizes two approaches for clustering in their work. They use both local and global clustering techniques. For the big graph data and dynamic datasets, global clustering method gets computationally expensive. However, the excessive computation can be reduced by storing the graph in an efficient data structure. The authors in [33] present Mixing Time of Random Walk (MTRW) for clustering. It is a randomized algorithm that extracts clusters from a graph according to a pre-specified metric. It finds the locally optimal solutions. The proposal is distributed and asynchronous. The work in [34] present Density-Based Clustering of Large Probabilistic Graphs (DBCLPG). Their method extract

Table 1
Key features of the proposed work and past methods.

Methods	Time complexity	Space complexity	No. of datasets used to evaluate	Suitable for large graphs	Evaluation metrics utilized	Based on random Walk/Friend-of-Friend
Proposed	$O(cn^2)$	$O(n^2)$	7	√	4	√
(Zhang et al. 2016)-LRW	$O(KJn_c)$	$O(n^2)$	5	√	2	√
(Akbari et al., 2019)-GAKH	$O(cn^2xI)$	$O(In^2)$	10	x	2	X
(Pavan et al. 2007)-PC	$O(n^2\ log(n))$	$O(n^2)$	3	x	2	X
(Avrachenkov et al. 2014)-MTRW	$O(n^3)$	$O(n^2)$	-	-	-	√
(Tabrizi et al. 2013)-PPC	O (*n*)	$O(n^2)$	12	√	1	√
(Halim et al. 2019)-DBCLPG	$O(nxn)$	$O(n^2)$	7	√	3	√
(Pons et al. 2005)-Walktrap	$O(mn^2)$	$O(n^2)$	6	√	4	√
(Papalexakis et al. 2013)-MULTICLUS	$O(k^{2m})$	$O(n^2)$	5	x	2	X

subgraphs $G`$ from an input graph $G$, such that $G` \subseteq G$ and the $G`$ density is above a certain threshold. They exploit the density of the extracted clusters in order to keep them growing.

2.2. Global graph clustering methods

This section focuses on the global graph clustering methods. Previous methods on global clustering techniques work for a few million vertices in dynamic and sparse graphs [27,28]. In global clustering each vertex is assigned its own cluster. Whereas, in a local clustering method cluster assignment is done only for a certain number of nodes. Clustering can be done on all the data points at once or this can be performed iteratively. To cluster the large databases simultaneously, online clustering algorithm scans all or at least some portion of the data at once [29]. This scheme enables to provide a solution for online analytical processing. Toussaint et al. [30] presents an analytical clustering approach to find spatial points using the nearest neighbour concept. For clustering the graph, a distance measure is used to capture the farthest points in the cluster rather than the closest ones. Zanghi et al. [31] work with graphs to present an approach for automated web page classification. They determine the online analytical clustering as k clusters. Various experiments for different values of k are performed to evaluate the classification performance. A tree structure is also introduced to make incremental clusters. The evaluation is performed for a set of documents taken from *Yahoo!*, indicating better classification performance. The work in [32] present a graph clustering method. It is a graph clustering algorithm based on Krill Herd (KH) and Genetic Algorithm (GA). It adopts the cycle and GA operators, using swarm intelligence, and utilizes the krill's movements.

2.3. Hierarchical clustering techniques

Clustering results that provide multi-level clustering are categorized under the hierarchical clustering methods. Such methods are different from the flat clustering approach where there is no level-wise representation of the data. Global clustering does not fulfil the problem of a single level. In hierarchical clustering, each cluster is assigned a level. Therefore, it is convenient to break them into desired number of clusters. This scheme depends on the complete observations of data for flat/hierarchical clustering. Hierarchical clustering is further divided into two types of approaches. One is agglomerative (bottom-up) and the second being divisive (top-down). Morisi et al. [35] use hierarchical clustering method to investigate the approximate computation of the consensus state of a network. In their method clusters are presented in a hierarchal manner. They use spectral graph theory method where a graph is divided into a number of partitions. Each subgraph has its own spectral properties. This facilitates in quick convergence toward the centroid of each subgraph. Guha et al. [36] present an approach named clustering using representatives (CURE). It is a hierarchical clustering method which first partitions the dataset and then partially clusters the data points accordingly. After removing the outliers, the pre-clustered data in each partition is then grouped to produce the final clusters. This clustering algorithm recognizes the arbitrarily shaped clusters and detects outliers. Guha et al. [37] present an approach called robust clustering using link (ROCK) to find the clusters. The ROCK framework exploits the link property when making decisions about the points to be merged into a single cluster. If the link between two points is large, it is probable that these two points belong to the same cluster. Clustering points based on only the closeness between them is not very efficient

because two nearby points may be neighbors. However, even if two points of different clusters are neighbors, it is unlikely that the pair has a large number of common neighbors.

2.4. Key limitations of the previous work

Zhang et al. [6] utilized the limited random walk algorithm for both global and local graph clustering problems. In some cases, domain experts are usually involved in determining the clusters for a given seed node. This is called the local clustering task. For instance, if a user requires to check the closely related friends and family members in a social network then the clustering approach should assign a node to traverse its neighbors only, instead of exploring the complete graph. The proposed approach in this work addresses this issue by limiting the random walk around the seed node. Additionally, the seed node is selected here based on a weighted connection strategy, instead of making a random choice. Tabrizi et al. [38] combined modularity function and random walk to precisely determine the clusters of a graph. They termed the strategy as personalized page rank clustering (PPC). Their method is a top down approach which recursively partitions subgraph until modularity gain is finished. Modularity function is not an appropriate choice to be combined with a random walk as there are other more compatible functions that can work with the random walk. The proposed work here addresses this issue by guiding the random walk utilizing the friends-of-friends concept. This enables to extract logically connected groups. Pavan et al. [26] used hierarchical clustering technique to find clusters from a dataset. The computational time for hierarchical clustering is $O(n^2 \log(n))$. This work addresses the same problem of clustering efficiently by consuming lesser additional resources. Papalexakis et al. [39] finds well defined clusters across all views and addresses the problem of multigraph clustering. They use two approaches to find clusters. One is tensor-based decomposition principle and second is the minimum description length-based technique. However, the presence of additional noise can degrade the performance of their approach by reporting false positive groups. The present proposal addresses this by discarding the singleton clusters formed by the random walk-based solution. Table 1 lists the key features of the closely related methods and the proposed approach. Where n_c is the average cluster size, J denotes the number of vertices that the LRW procedure visits in each iteration and K is the number of iterations for the LRW procedure to converge.

3. Proposed solution

This section presents the proposed solution. The work in this paper addresses graph clustering using random walk-based approach. For this, first, the basic notations are defined and later the proposed strategy is explained. The main goal of this work is to address the task of clustering graphs using an efficient random walk method. For this, a novel walk approach in a graph is presented that determines the weight of the edges and the degree of the nodes. This information is utilized by the pseudo-guidance model to guide the random walk procedure.

3.1. Basic notations

Graph $G(V,E)$ is represented as a combination of nodes N and edges E. Where $V = \{V_1, V_2, V_3, \ldots V_n\}$ is the set of nodes (also called the vertices) and $E=\{E_1, E_2, E_3, \ldots E_n\}$ is the set of edges. Suppose A_g is the adjacency matrix formed from a graph G. The adjacency matrix A_g is the combination of 0s and 1s with nodes represented by the columns and rows. Intersection of a row and a column represents the existence or unavailability of edge between

these nodes. If a cell contains zero, it means there is no edge between the nodes, otherwise an edge exists between those nodes. Suppose D is the diagonal matrix in which the diagonal of a matrix represents the degree of each node in the graph. In a diagonal matrix, all other elements apart from the diagonal elements are zeros. Such matrix is utilized when graph having self-loops are used for mapping a real-world problem.

This work mainly focuses on the clustering of graphs through random walk. It is therefore needed to describe the random walk on a graph here followed by the proposed strategy. Consider a graph with $|N|$ nodes and $|E|$ edges. These edges can either be directed or undirected. Random walk in a graph is a probabilistic process of visiting neighboring nodes. When a walking agent starts its walk from a seed vertex it can jump to any of its neighbor subject to the connectivity through an edge. Walking agent is solely responsible for starting its walk from the seed vertex and then jump to any of its neighbor randomly. The selection of the neighbor depends on a specific criteria, at times, depending on the problem domain.

Consider a graph $G\ (V, E)$ having $|V|$ vertices and $|E|$ edges**.** A walking agent starts a random walk from the seed vertex V_0 and moves to its neighbor on the basis of a probability distribution function. The probability of the neighbor node being visited is usually determined as, $1/deg(v_t)$. Where v is a vertex, V_t is the node at step t, and $deg(v)$ is the degree of the node v. The probability of moving to a node is determined using Eq. (2). The probability is computed using the Markov clustering method.

$$P_t(i) = \text{Prob}(v_t = i) \tag{2}$$

3.2. Proposed strategy

In order to extract coherent clusters from a graph, first the degree of each node is determined. The node having maximum degree is selected as a source node for the random walk to start. This enables to select the most connected node as the starting point to initiate the walk. Considering an analogy of a social network, a well-connected person (assumingly also influential in terms of disseminating information) will have the maximum degree. However, this node's degree is not traditionally computed using the count of edges incident on a vertex. For this, the weight of the edges is first computed based on the friends-of-friends concept. Instead of utilizing the raw edge weight of a graph, the edge weight here is computed as the count of the number of common neighbors of the nodes connected by the edge under consideration. Consider the example graph shown in Fig. 1 having

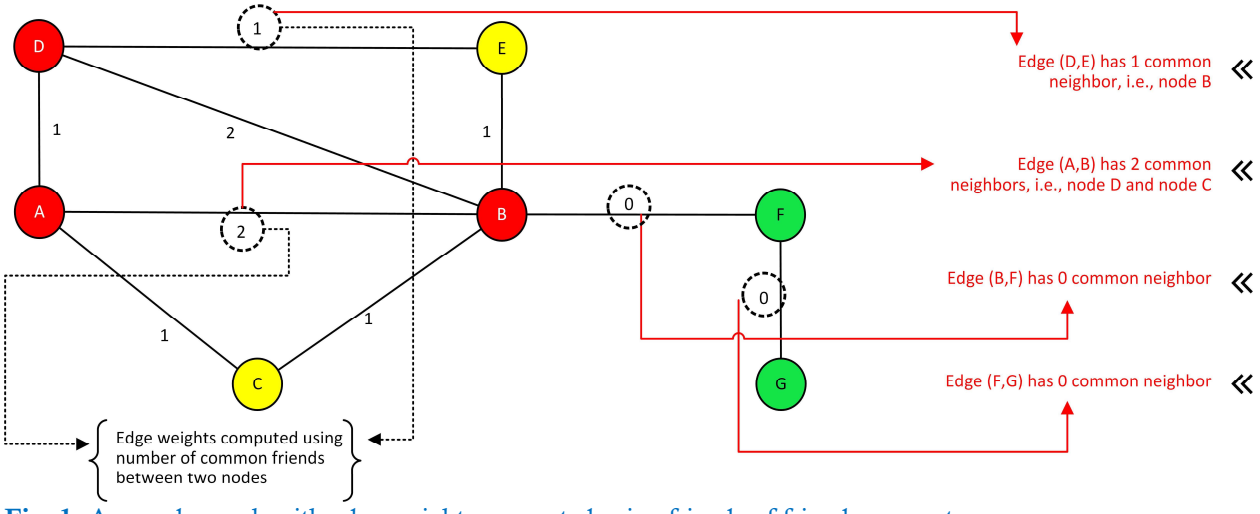

Fig. 1. A sample graph with edge weights computed using friends-of-friends concept.

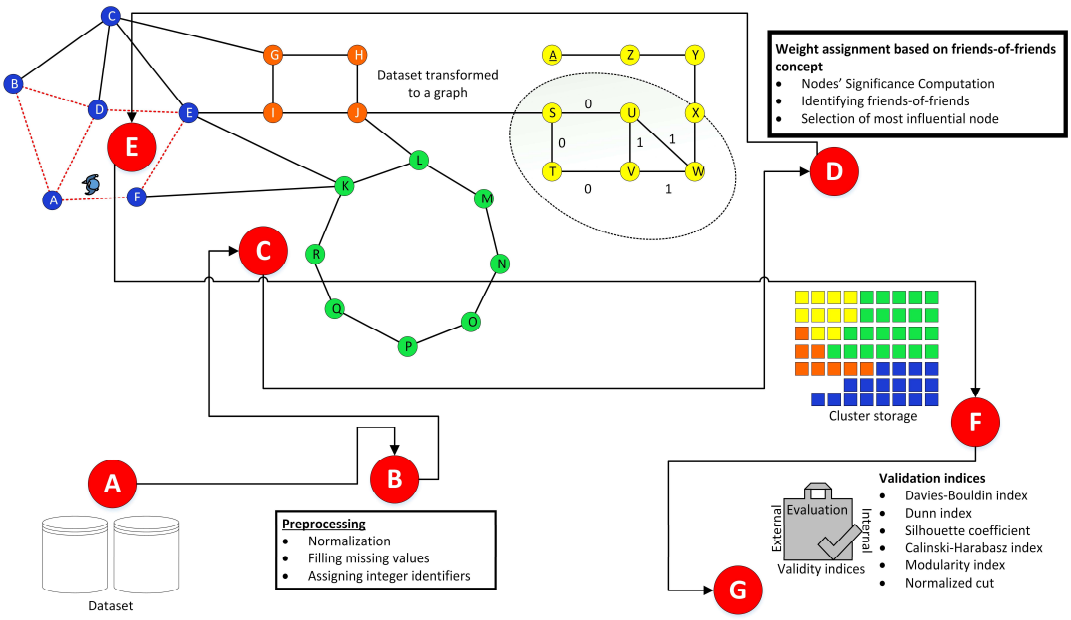

Fig. 2. Overall working of the proposed solution.

seven nodes. For the edge between node A and node B the adjacency matrix-based data structure will have a value of 1 in the corresponding cell, indicating edge existence between the two nodes. However, based on the friends-of-friends concept, the number of common friends of these two nodes is 2, this becomes the edge weight for (A, B). Similarly, there is no common friend of the nodes B and F therefore a weight of 0 is assigned to the edge (B, F). Weights for the rest of the edges can be seen in Fig. 1. Node weight is calculated using these weights on the edges.

This work presents an innovative approach for extracting clusters in a graph based on the intuition of commonalities between nodes. For this, the current proposal first focuses on finding the common neighbors for the nodes incident on an edge. Using this, the weight of the edge is determined between the two nodes. If there is no common neighbor, the weight assigned to this edge is 0. If there is one common neighbor, a weight of one is mentioned over the edge, and this process continues. After determining the edges' weights, the random walk starts from the highest degree node to its neighbors. Where, degree of a node is computed by summing the edge weight incident on the node. An agent walks on the graph edges using a guidance mechanism. The agent is guided using the edge weights. It continues to walk as long as a weight greater than 0 or T_r (a user provided number is available). Using this, the agent moves to the neighbors. The stopping criteria for the agent is met if all non-zero edges are traversed or there is no direct path available for the agent to reach another node from the current position. Once a walk is completed by the agent, the visited nodes are extracted from the original graph and these form a cluster. Afterwards, the agent starts its next iteration. However, for this, the edge and node weights are recomputed first for the remaining graph. This process iterates until all nodes in the original graph are processed. The singleton clusters obtained using this process are termed as noise and are therefore discarded. Fig. 2 visually displays working of the proposed solution. The proposed approach selects the most concentrated node as the seed to start the random walk. However, there is a possibility that a tie may occur between multiple nodes. To address this case, any one of the nodes ranked at the same level is selected as the seed. This is done

```
Procedure Pseudo-Guided Random Walk (D, Tr)
Input: Dataset (D), threshold (Tr)
Output: N Connected clusters

Start
1.      Start reading the dataset D from the text file
2.      nodeIdentifier = 0
3.      foreach node in the dataset do {
4.          Assign a unique identifier nodeIdentifier
5.          Increment nodeIdentifier by 1
6.      }
7.      Convert the processed dataset D to a graph G(V,E)
8.      Store the graph G(V,E) in an adjacency matrix M
9.      foreach row(r) in M do {
10.         foreach column(c) in M do {
11.             if M[r,c] > 0{
12.             M[r,c]=1
13.             }
14.         }
15.     }
16.     M = M x M
17.     foreach row(r) in M do {
18.         nodeWeight[r]= M[r,r]
19.     }
20.     i=0
21.     Sort(nodeWeight[N])
22.     highestDegreeNode=nodeWeight[0]
23.     Do
24.         Start the guided walk from highestDegreeNode
25.         If edgeweight greater than Tr
26.             Move to its neighbor
27.             Add node to the list ClusterListi
28.         Else
29.             Move to another neighbor
30.         End if
31.     While all edges are explored
32.     foreach node ni in ClusterListi{
33.         foreach row(r) in M do {
34.             M[r,ni]=0
35.         }
36.     }
37.     If size of ClusterListi > 1
38.         Export the ClusterListi to the list of allCLusters
39.     End if
40.     i=i+1
41.     sumM=0
42.     foreach row(r) in M do {
43.         foreach column(c) in M do {
44.             sumM += M[r,c]
45.         }
46.     }
47.     If sumM  > 0
48.         Repeat from Step 23
49.     Else
50.         Return allCLusters
51.     End if
End
```

Fig. 3. Pseudocode of the proposed random walk-based clustering approach.

randomly because the current proposal provides priority list to the random walker to choose from instead of hard coding the walk. This can result in different clustering formations when the proposed algorithm is executed multiple times. The same is true for a pure random walk-based clustering solution. However, due to the current proposal's pseudo-guidance mechanism the final clustering quality will be better than a pure random walk. The time complexity of the proposed solution is $O(cn^2)$. Where, c is the number of clusters and n is the number of

nodes. It is worth mentioning that the tighter bound on the random walk is $O(n^3)$ for a graph with n nodes. However, the current proposal does not require the random walk procedure to be exhaustively executed for the complete graph. Instead, it limits the walk based on the values of the threshold T_r and the number of clusters c.

As shown in Fig. 2, the dataset is first converted into a graph-based representation (if it is already not in a graph format). This is followed by the steps of preprocessing, i.e., computing edges' and nodes' weight, and then the guided random walk procedure is adopted for clustering. The process gives clusters as the output by discarding those groups having one node only, i.e., singleton clusters.

The random walking agent (random walker) in this approach starts its walk from the highest weighted node, later it randomly selects any of the neighboring nodes having non-zero edge weight between the current node and the neighbor. Since the edge weights are assigned based on commonalities between the adjacent nodes, this guidance mechanism enables to extract more coherent clusters using the random walk in the graph. Pseudocode of the proposed approach is listed as Fig. 3. The proposed approach receives two parameters, the dataset (D) and the threshold (T_r). Initially, the dataset is read pairwise and the graph representation is formed by creating nodes for the data items and drawing an edge between two adjustment nodes. Next, the graph is stored in an adjacency matrix (M). The adjacency matrix is used here as a data structure to store the graph due to its lower time complexity in retrieving information in comparison to the adjacency list. The matrix M is multiplied by itself to compute the number of common neighbors between any two nodes. The loop from line 9-15 converts every number in the adjacency matrix that is greater than 0 to a 1 and the rest to a 0. This loop is a preprocessing step that is required to obtain the number of common neighbors between any two nodes. The adjacency matrix is multiplied by itself and the resultant matrix contains the number of common neighbors between any two nodes (with an exception of the values at the diagonal). In case, the loop from line 9-15 is omitted the computation of common neighbors via matrix multiplication will give erroneous values. The values at the diagonal of the matrix

Table 2
Summary of the datasets utilized.

Dataset	No. of nodes	No. of edges	Domain	Universal Resource Locator (URL)
Karateclub	34	78	Social network	http://konect.uni-koblenz.de/networks/ucidata-zachary
Mcldata	200	2500	Images	http://mcl.usc.edu/mcl-jcv-dataset/
Dolphins	62	159	Life science	http://konect.uni-koblenz.de/networks/dolphins
Lesmis	77	254	Miscelenious	http://konect.uni-koblenz.de/networks/moreno_lesmis
Facebook combined	4037	87933	Social network	https://snap.stanford.edu/data/egonets-Facebook.html
Moreno health	2539	12969	Social network	http://konect.uni-koblenz.de/networks/moreno_health
Air traffic control	1226	2615	Infrastructure	http://konect.uni-koblenz.de/networks/maayan-faa
Wiki-Vote	7115	103689	Wikipedia who-votes-on-whom network	https://snap.stanford.edu/data/wiki-Vote.html
192bit	14000	154000	Miscellaneous Networks	http://networkrepository.com/192bit.php
176bit	7000	82000	Miscellaneous Networks	http://networkrepository.com/misc.php
Citation	27400	705084	Scientometrics	http://www.sommer.jp/graphs/
Protein Interaction	2361	75740	Bioinformatics	http://vlado.fmf.uni-lj.si/pub/networks/data/bio/Yeast/Yeast.htm
GEOM	7343	11898	Bioinformatics	http://vlado.fmf.uni-lj.si/pub/networks/data/collab/geom.htm
ENRON	36692	183831	Email network	http://snap.stanford.edu/data/email-Enron.html
GRQC CNetwork	5242	14496	Scientometrics	http://snap.stanford.edu/data/ca-GrQc.html
CM Cnetwork	23133	93497	Scientometrics	http://snap.stanford.edu/data/ca-CondMat.html
AP Cnetwork	18772	198110	Scientometrics	http://snap.stanford.edu/data/ca-AstroPh.html
EP Cnetwork	12008	118521	Scientometrics	http://snap.stanford.edu/data/ca-HepPh.html

obtained after multiplication are ignored. An extra row is maintained in the adjacency matrix to store the nodes' weight. This enables to obtain a node's pre-computed weight in O(1) time. The weight of a node is the sum of weights of all the edges incident on it. Next, the clustering procedure is started by placing the random walking agent on the node with the highest weight. The agent randomly picks the next node from the neighbors of the starting node. However, visiting the node is decided based on the value of T_r and the edge weight. The variable T_r is a user provided threshold that determines the minimum number of common connectivity between a cluster and its neighbor to consider it for merging into the cluster. Each cluster grows as long as valid neighbors are available in its vicinity. Line 31 moves the random walking agent arbitrarily to any of its neighbor. Once the cluster stops growing, it is extracted from the original graph and the same process is repeated for the remaining graph.

4. Experiments

This section lists the conducted experiments and obtained results. For experiments, 18 benchmark datasets are utilized. The choice of these datasets is made based on their diversity and utilization in previous such studies. This section first explains the datasets, followed by the evaluation metrics, and a brief about the competing approaches before listing the results. All experiments are performed on Intel Core i7 machine with 3.2 GHz processor and 8 GB RAM.

4.1. Datasets

The experiments here are performed on 18 benchmark graph datasets. These are taken from various sources keeping domain and size diversity in view. These include: *Karateclub, Mcldata, Dolphins, Lesmis, Facebook combined, Moreno health, Air traffic control, Wiki-Vote, 192bit, 176bit, Citation, Protein Interaction, GEOM, ENRON, GRQC, CNetwork, CM Cnetwork, AP Cnetwork,* and *EP Cnetwork*. The size of these datasets ranges from 34 nodes to 36692 nodes. Additionally, these datasets have a minimum of 78 and a maximum of 705084 edges. The size of a graph is dependent on both number of nodes and the number of edges. For a complete analysis either the number of nodes or the number of edges cannot be considered in isolation. For example, if a graph has say 1000000 nodes with 0% density, i.e., no edges, considering such graphs for simulations will be inappropriate. Similarly, for a graph having 100% density, but with too few nodes, say 10 will also be unsuitable. Therefore, the choice of datasets for simulations in this work is made based on both the number of nodes and the number of connecting edges in the graph. Table 2 lists the summary of these datasets. The *karateclub* is a social network of a university karate club which has 34 nodes and 78 links. The *karateclub* network is divided into two groups by Zachary et al. [40]. The *mcldata* represent 24 source videos having resolution 1920×1080 and 51 H.264/AVC encoded clips for each source sequence. The dataset consists of 200 nodes and 2500 edges between them. The *protein Interaction* dataset is about the predictions of cellular localization sites of proteins with 2361 nodes and 75740 edges. It has eight features per instance. The *dolphins* data is a directed social network of bottlenose dolphins. The nodes represent bottlenose dolphins living in a community of Doubtful Sound, a fjord in New Zealand. An edge indicates their frequent association. The dataset has 62 nodes and 159 edges. For this dataset, the dolphins were observed between 1994 and 2001. The *lesmis* dataset represent a co-appearance network (*igraph*

object[2]) of characters in the novel Les Miserables (written by a French writer Victor Hugo). Its vertices are the novel characters and an edge indicates that the two characters appear together in the same chapter at least once. Vertex attributes for this graph are unique identifier, a vertex number between 1 and 77, and label, i.e., the character's name. The dataset *facebook combined* is extracted from actual users of the facebook. The dataset has 87933 edges between 4038 nodes. The links in this data do not carry any weight. The *air traffic control* data is constructed from the USA's FAA (Federal Aviation Administration) National Flight Data Center (NFDC), consisting of preferred routes. Nodes in this network represent airports or service centers and links are created from strings of preferred routes recommended by the NFDC. This network has 1226 vertices and 2615 edges. The *moreno health* dataset is a directed network created from a survey data. Each survey participant listed her/his five best female and five male friends. A node in the data represents a student and an edge between two students shows that the left student (node) chose the right student (node) as a friend. Higher edge weights indicate more interactions. The *wiki-vote* dataset is extracted from Wikipedia using its dump page edit history by extracting all administrator elections and vote history data. The dataset has 7115 nodes and 103689 edges. The 192bit and 176bit are taken from the interactive data and network data repository having 14000 and 7000 nodes, respectively. The *citation* data contains 27400 nodes and 705084 edges representing the number of citations from the scientific papers represented as nodes. The *GEOM* data is a collaboration network with 7343 nodes representing different authors and 11898 edges, which represents the joint work of multiple authors. The dataset *enron* is an email network having 36692 nodes and 183831 edges. The *GRQC CNetwork* (General Relativity and Quantum Cosmology collaboration network) data comprises of 5242 nodes and 14496 edges. The nodes in this dataset represent the authors and the edge between two authors represents collaborating authors. The *CM Cnetwork*, *AP Cnetwork*, and *EP Cnetwork* datasets represents collaboration network is from the e-print arXiv and covers scientific collaborations between authors papers submitted to condense matter physics, astro physics, and high energy physics-phenomenology categories, respectively.

4.2. Other competing approaches

The random walk-based clustering solution presented in this work is compared with seven closely related state-of-the-art methods. These include: Walktrap [16], Pairwise Clustering (PC) [26], Personalized page rank clustering (PPC) [38], Mixing Time of Random Walk (MTRW) [33], Limited Random Walk (LRW) [6], Genetic Algorithm Krill Herd for graph clustering (GAKH) [32], and Density-Based Clustering of Large Probabilistic Graphs (DBCLPG) [34]. The choice of these methods is made based on close relevance mainly decided by the utility of graphs, random-walk, or friend-of-friend concept. Table 3 lists a summary of the competing methods with respect of various relevance factors. As shown in the table, all the competing methods addresses graph clustering and 57% of these, i.e., 4 are the graph clustering methods that utilize random-walk. In terms of recency, these seven methods can be divided into three brackets; latest, established, and classic. It can be seen from the table that 28% of the competing methods are from the latest bracket, 28% are from the

[2] http://igraph.org/r/doc/

Table 3
Summary of the competing methods.

Competing method	Year	Citations (April 2020)	Utilizes graphs?	Utilizes random walk?	Utilizes friend-of-friend concept?
Walktrap [16]	2005	1510	√	√	x
Pairwise clustering (PC) [26]	2007	464	√	x	x
Personalized page rank clustering (PPC) [38]	2013	38	√	√	x
Mixing Time of Random Walk (MTRW) [33]	2014	11	√	√	x
Limited random walk (LRW) [6]	2016	12	√	√	x
Genetic algorithm krill herd for graph clustering (GAKH) [32]	2019	.	√	x	x
Density-Based Clustering of Large Probabilistic Graphs (DBCLPG) [34]	2019	6	√	x	√

latest group, and 42% of the comparison methods are the classic ones. Following is a brief description of these seven clustering methods.

Limited Random Walk (LRW): The limited random walk [6] utilizes inflation and normalization applied at each step to cluster a graph. Applying inflation operator to the matrix causes it to increase the existing small differences. The approach is time-homogeneous Markov chain procedure. The procedure uses both local and global clustering techniques.

Pairwise Clustering (PC): Pairwise clustering [26] is based on merging two points that are closely related to each other according to their characteristics. This process of merging the clusters continues until all points are traversed or a stopping criteria is met.

Personalized Page rank Clustering (PPC): Personalized page rank clustering [38] combines modularity function and random walk to precisely determine the clusters of a graph. The method is a top down approach which recursively partitions subgraph until modularity gain is finished.

Genetic Algorithm Krill Herd for graph clustering (GAKH): The work in [32] present a graph clustering algorithm based on KH and GA. It adopts the cycle and GA operators, using swarm intelligence, and utilizes the krill's movements.

Mixing Time of Random Walk (MTRW): MTRW [33] is a randomized algorithm that extracts clusters from a graph according to a specified metric. It finds the locally optimal solutions. The proposal is distributed and asynchronous.

Density-Based Clustering of Large Probabilistic Graphs (DBCLPG): The DBCLPG method [34] extracts subgraphs G` from an input graph G, such that G` ⊆ G and the density of G` is above a certain threshold. They exploit the density of the extracted clusters in order to keep them growing.

Walktrap: The work in [16] is a random-walk-based clustering technique that computes distance between two nodes using a fixed number of steps utilizing the probability matrix. The method walktrap captures community structure in a network which can be used in an agglomerative clustering algorithm.

4.3. Evaluation metrics

The clustering results of the competing approaches are evaluated using six clustering evaluation metrics, namely: Davies Bouldin index (DBI), Dunn index (DI), Calinski-Harabasz index (CHI), silhouette coefficient (SC), modularity index, and normalized cut.

Davies-Bouldin Index (DBI): DBI validates the inter-cluster and intra-cluster similarity of the nodes in the formed clusters. The index validates the clusters based on dimensions inherent to the dataset. Mathematically, DBI is computed using Eq. (3). Where, n represents the number of clusters. The variable σ_i represents the average distance of all the nodes in the cluster to the center of the cluster c_i. The value of $d(c_i, c_j)$ is the distance between centroids of the two clusters c_i and c_j. The DBI ranges in $[0, \infty]$. The value of DBI closer to 0 indicates better clustering formation.

$$DBI = \frac{1}{n}\sum_{i=1}^{n} \max_{i \neq j} \left(\frac{\sigma_i + \sigma_j}{d(c_i, c_j)} \right) \tag{3}$$

Dunn Index (DI): DI is another evaluation metric for the validity of clustering results. It checks how dense the cluster is with-in and how well it is separated from other clusters. Eq. (4) shows the mathematical formulation of DI. Where $d(i,j)$ denotes the distance between two clusters, while $d'(k)$ represents the distance between the nodes within the cluster. For a given clustering formation, higher values of DI indicate better clustering.

$$DI = \min_{1 \leq i \leq n} \left\{ \min_{1 \leq j \leq n, i \neq j} \left\{ \frac{d(i,j)}{\max_{1 \leq k \leq n} d'(k)} \right\} \right\} \tag{4}$$

Calinski-Harabasz Index (CHI): Calinski-Harabasz index measures the clustering quality by gauging the separation between formed groups. Eq. (5) lists the formula to compute CHI.

$$CHI = \frac{S_b/(k-1)}{S_w/(n-\mathrm{k})} \tag{5}$$

Where, S_b is the squared sum of inter-cluster distance, S_w is the squared sum of intra-cluster distance, k is the number of clusters, and n is the number of objects (nodes in this case). Higher CHI values indicate better clustering.

Silhouette Coefficient (SC): This clustering validation metric (Eq. (6)) checks nodes' similarity with all other nodes in its cluster and it also checks dissimilarity of the node from the members of other clusters. The value of SC is in the range [-1, 1]. The clustering formations having higher SC value have

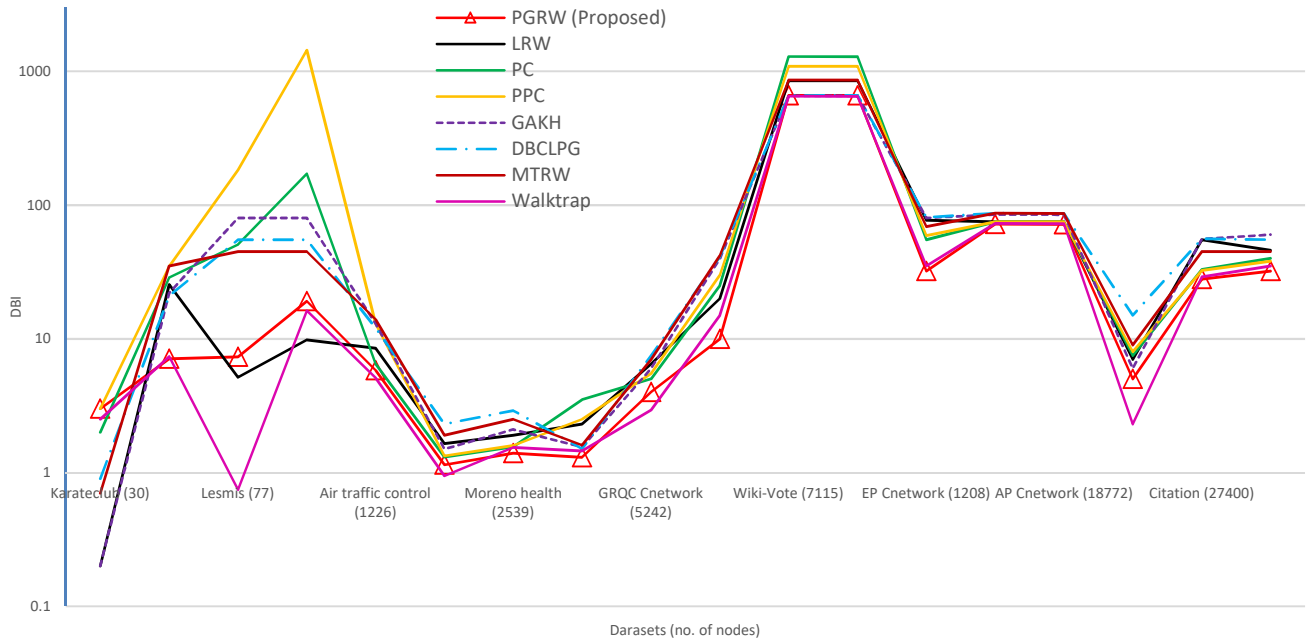

Fig. 4. Results of the eight competing methods using DBI.

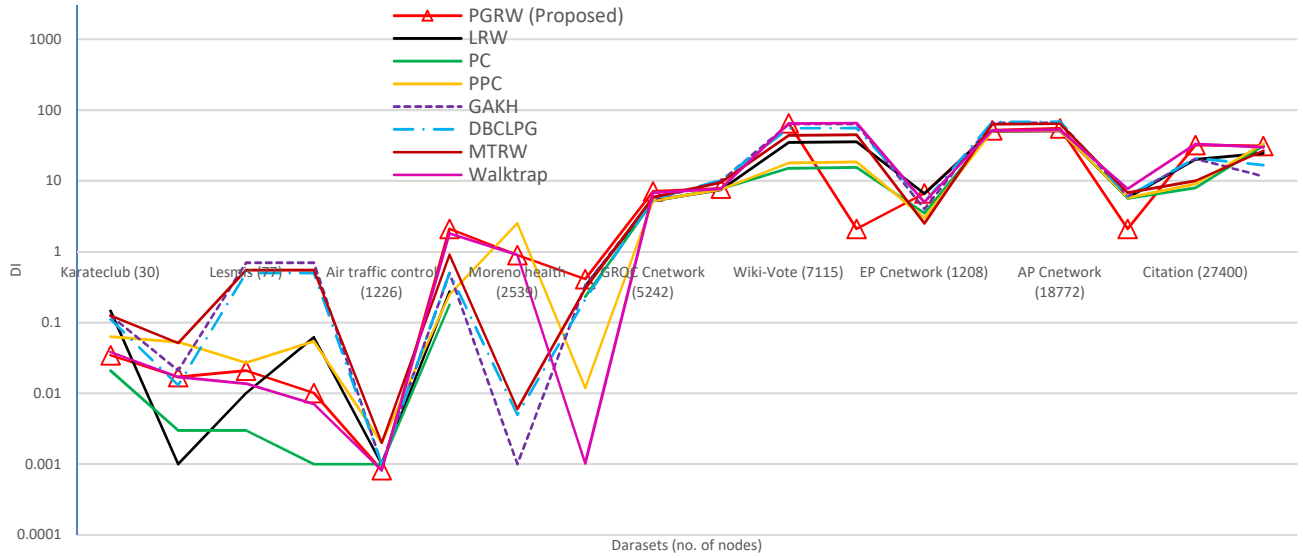

Fig. 5. Results of the eight competing methods using DI.

more cohesion that those having lower SC values. Mathematically, silhouette coefficient can be written as follows.

$$SC = \frac{1}{N}\sum_{i=1}^{N}\{\frac{b(i)-a(i)}{\max\{a(i),b(i)\}}\} \tag{6}$$

Where, *b(i)* is the average dissimilarity of node *i* with all the other nodes in the other clusters, i.e., inter-cluster dissimilarity. The value *a(i)* is the average dissimilarity of node *i* from the other nodes within the cluster, i.e., intra-cluster dissimilarity.

Modularity index: Modularity index finds the modular strength of a cluster in a network. It is computed using Eq. (7). Higher modularity value indicates better clustering value.

$$Q = \sum_{i=1}^{k}(e_{ii} - a_i^2) \quad (7)$$

Where, a_i is the percentage of edges with at least one end in module I, $a_i = |\{(u,v) : u \in Vi, (u,v) \in E\}| / |E|$. The term e_{ii} show percentage of edges in module i, $e_{ii} = |\{(u,v) : u \in V_i, v \in V_i, (u,v) \in E\}| / |E|$.

Normalized cut: It is another metric to find the partitions of graphs with minimum connections between two clusters. The optimal bi-partitioning of a graph is the one that returns the minimum cut value. It is computed using Eq. (8).

$$Ncut(A,B) = \frac{cut(A,B)}{asso(A,V)} + \frac{cut(A,B)}{asso(B,V)} \quad (8)$$

Where, $cut(A,B) = \sum_{u \in A, v \in B} w(u,v)$ and $asso(X,V) = \sum_{u \in X, t \in V} w(u,t)$ is the total connection from nodes in X to all nodes in the graph.

4.4. Comparison

This section presents the comparison of the proposed approach, this point onwards referred to as Pseudo-Guided Random Walk (PGRW). The comparison here is made with seven baseline/state-of-the-art related clustering methods using the 18 benchmark datasets. Seven standard evaluation metrics are used here, namely, DBI, DI, CHI, SC, modularity index, normalized cut, and execution time. The obtained results are grouped here metric wise.

4.4.1. Comparison based on DBI

The DBI is a stranded clustering validity index to gauge clustering quality. For a clustering formation produced by a method, the DBI represents a numeric value, where the lesser value of this index represents better clustering formation. The DBI computed for the eight computing methods is shown in Fig. 4. It can be seen from the results that the proposed methods perform better than others on nine (out of 18) datasets achieving minimum DBI value. The datasets for which PGRW performed better include, *dolphins, moreno health, facebook combined, 176bit, EP Cnetwork, 192bit, AP Cnetwork, citation,* and *ENRON*. The clustering method Walktrap has been the second best by achieving minimum DBI value over 7 out of the 18 datasets. The method LRW performs better on two datasets, i.e., *karateclub* and *mcldata*. The DBCLPG clustering method has performed the least based on the DBI by achieving the maximum DBI value (indicating inappropriate clustering formation) over eight datasets.

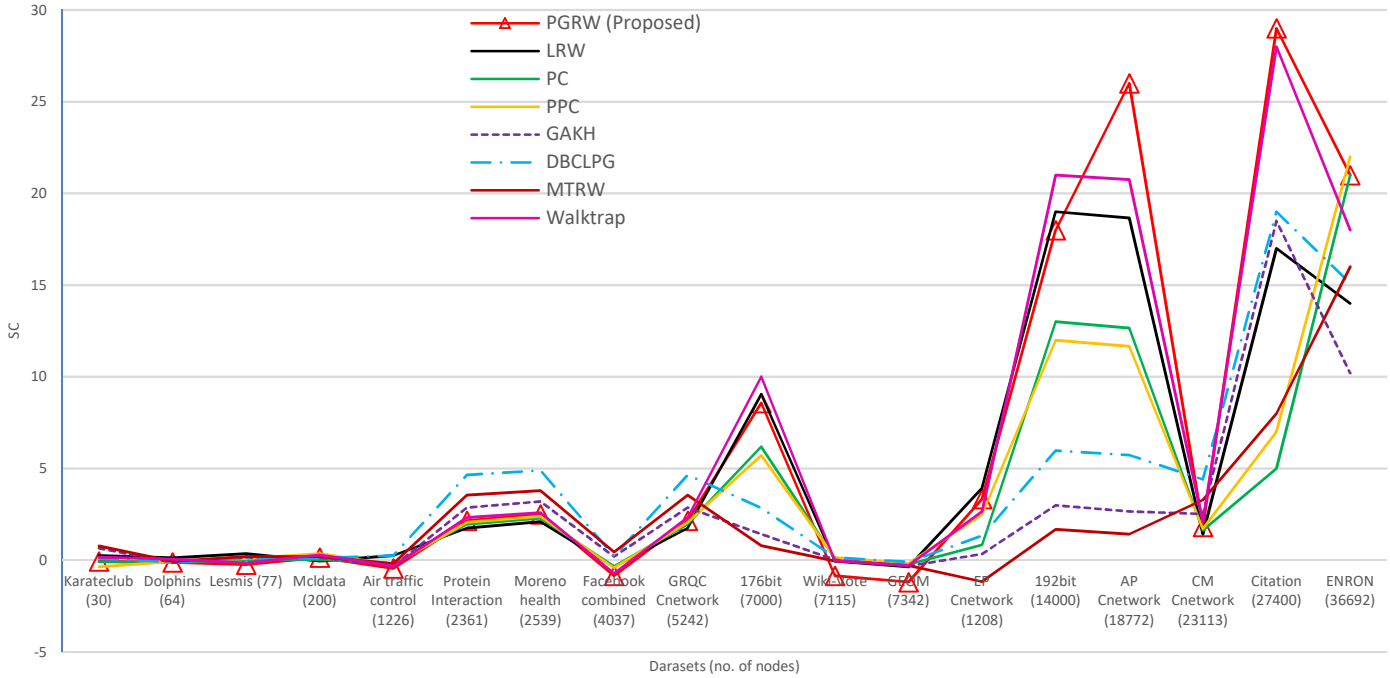

Fig. 6. Results of the eight competing methods using SC.

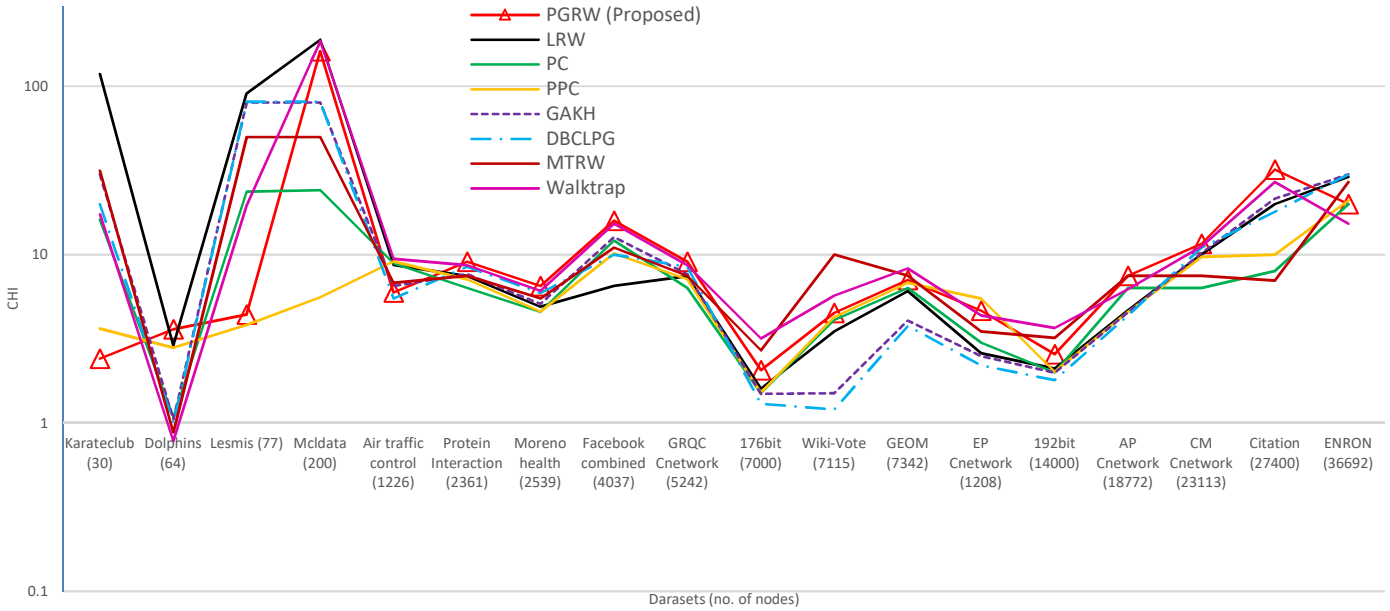

Fig. 7. Results of the eight competing methods using CHI.

4.4.2. Comparison based on DI

The DI is the second cluster validity index used here for comparing performance of the eight clustering methods. The DI computes a numeric value for a given cluster formation, where a higher DI value indicates better clustering as compared to a clustering formation for which the computed DI is low. The DI values obtained for the eight competing methods is shown in Fig. 5. Based on this metric, the proposed approach has performed

better on *protein interaction, facebook combined, GRQC Cnetwork, wiki-Vote,* and *EP Cnetwork* datasets. As shown in the figure, PPC, DBCLPG, and Walktrap perform equally and achieves second best DI value (indicating appropriate clustering formation) on three datasets each. Whereas, the clustering method GAKH performed best on two datasets (i.e., *lesmis* and *mcldata*) and MTRW has better performance only on the *air traffic control* dataset.

4.4.3. Comparison based on SC

The SC is a standard validity index used widely to evaluate clustering quality. The value of SC ranges between -1 to 1, where the SC value close to 1 indicates more cohesive clustering. An experiment has been performed that computes the SC value of each of the eight competing methods over the 18 benchmark datasets. The results of this are shown in Fig. 6. The figure shows that the proposed approach performs better only on *AP Cnetwork* and *citation* datasets based on the SC metric. Based on the results in Fig. 6 the DBCLPG method performs better than others by achieving maximum SC value over six datasets, namely, *air traffic control*, *protein Interaction, moreno health, GRQC Cnetwork, GEOM,* and *CM Cnetwork*. Performance of LRW and PPC is the second best as they obtain higher SC values on three datasets each. Whereas, the performance of the proposed method, i.e., PGRW, and two other approaches is ranked 3rd based on SC as they obtain better SC value than others on two datasets each. Clustering is an unsupervised learning task and gauging the quality of results in the absence of the ground truth is challenging. Performance evaluation using one metric may differ than the other in such scenario. Therefore, in all such cases, multiple metrics are utilized to make an informed quantitative decision about the performance.

4.4.4. Comparison based on CHI

The CHI is the fourth cluster validity index used in this work for evaluating the clustering quality obtained by the proposed approach and the seven state-of-the-art methods used for comparison. Like other validity indices, the CHI is also a numeric value where a higher CHI value indicates better clustering formation in comparison to a lower value. An experiment has been performed that computes the CHI value of each of the eight competing methods over the 18 benchmark datasets. The results of this are shown in Fig. 7. The results suggest that PGRW perform better than others on the eight datasets. These datasets include *dolphins, protein interaction, moreno health, facebook combined, GRQC Cnetwork, AP Cnetwork, CM Cnetwork,* and *citation*. Using CHI as a metric, the Walktrap clustering method performs the second best by achieving higher CHI values than others on three datasets. LRW has performed better than others in three cases. The clustering methods, PPC, GAKH, DBCLPG, and MTRW perform better than other competing approaches in one case each.

4.4.5. Comparison based on modularity index

Modularity index is used to find the modular strength of a cluster in a network. It is utilized here as the fifth cluster validity index. The modularity index values computed for the eight computing methods are shown in Fig. 8. The results suggest that the proposed approach performs better than others on eight datasets, namely, *protein interaction, moreno health, facebook combined, GRQC Cnetwork, wiki-Vote, EP Cnetwork, CM Cnetwork,* and

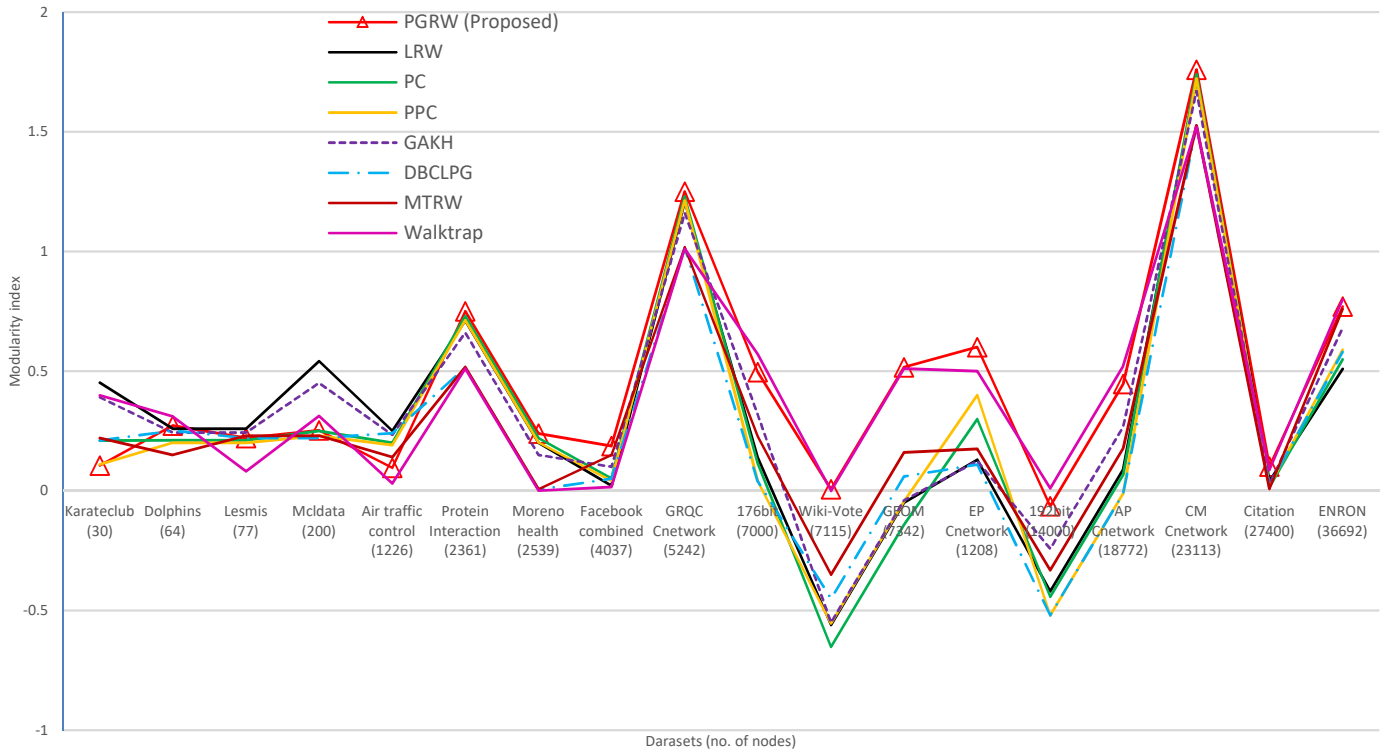

Fig. 8. Results of the eight competing methods using modularity index.

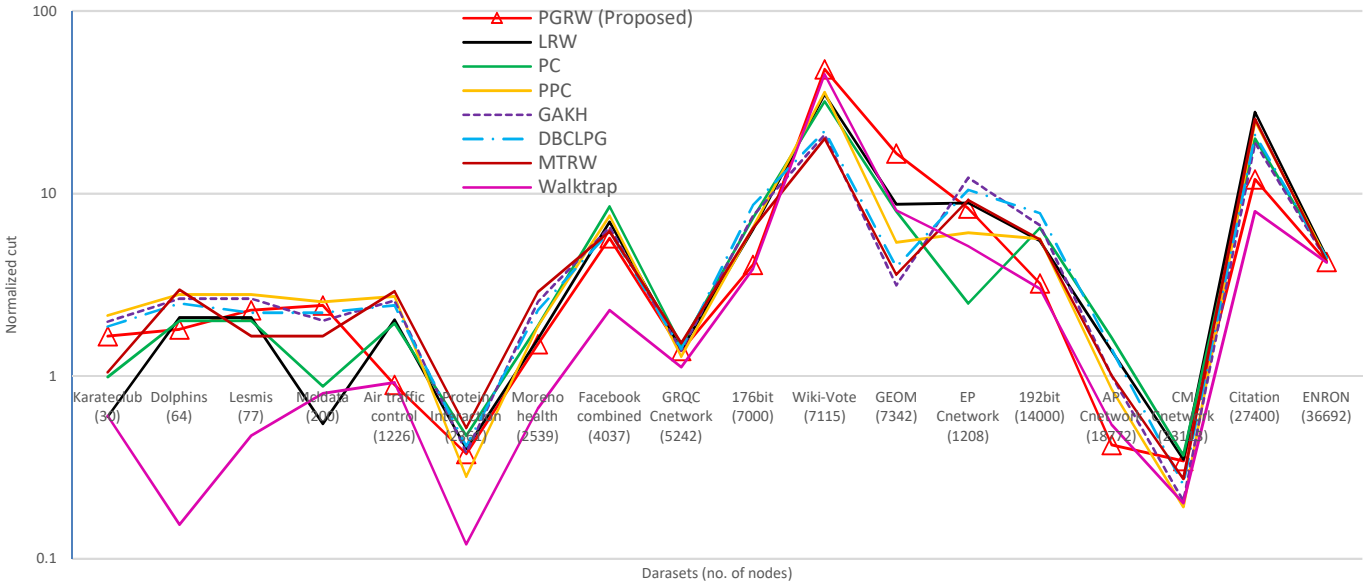

Fig. 9. Results of the eight competing methods using normalized cut.

citation. The Walktrap and LRW are ranked 2nd and 3rd respectively by performing better than others on 6 and 4 dataset, respectively.

4.4.6. Comparison based on normalized cut

The normalized cut is a standard metric majorly used in graph theory to find the partitions of graphs with minimum connections between two clusters. An experiment has been performed where each of the competing methods is executed on the 18 benchmark datasets and normalized cut of the resulting clustering formations is computed accordingly. The normalized cut is represented by a numeric value where the lesser value of this metric indicates better clustering formation. The results of this experiment are shown in Fig. 9. These results suggest that the Walktrap method performs better in case of normalized cut by achieving better value of this metric on nine datasets, i.e., *dolphins, lesmis, protein interaction, moreno health, facebook combined, GRQC Cnetwork, 176bit, 192bit, citation,* and *ENRON.* Whereas, PGRW and LRW are ranked second here as they perform better than other clustering approaches in two cases each. The DBCLPG is the least performing clustering method based on the normalized cut as it could not get the minimum value in comparison to others on any of the 18 datasets.

4.4.7. Comparison based on execution time

A comparison of the eight competing methods is also performed based on execution time. The eight clustering methods are executed on all 18 benchmark datasets and the execution time of each is recorded. Although the asymptotic time complexity of the competing methods is listed in Table 1, however, the actual execution may vary majorly depending on the dataset represented as a graph, its number of nodes and the number of edges. It is worth mentioning here that two of the competing methods, i.e., LRW and PPC are parallel computing-based clustering approaches and they will have a clear advantage in execution time over all other methods. For the sake of completeness, execution time of these two methods is also reported. Fig. 10 shows the execution time of the eight competing methods. The proposed method performs better than others (in terms of time) on *karateclub*,

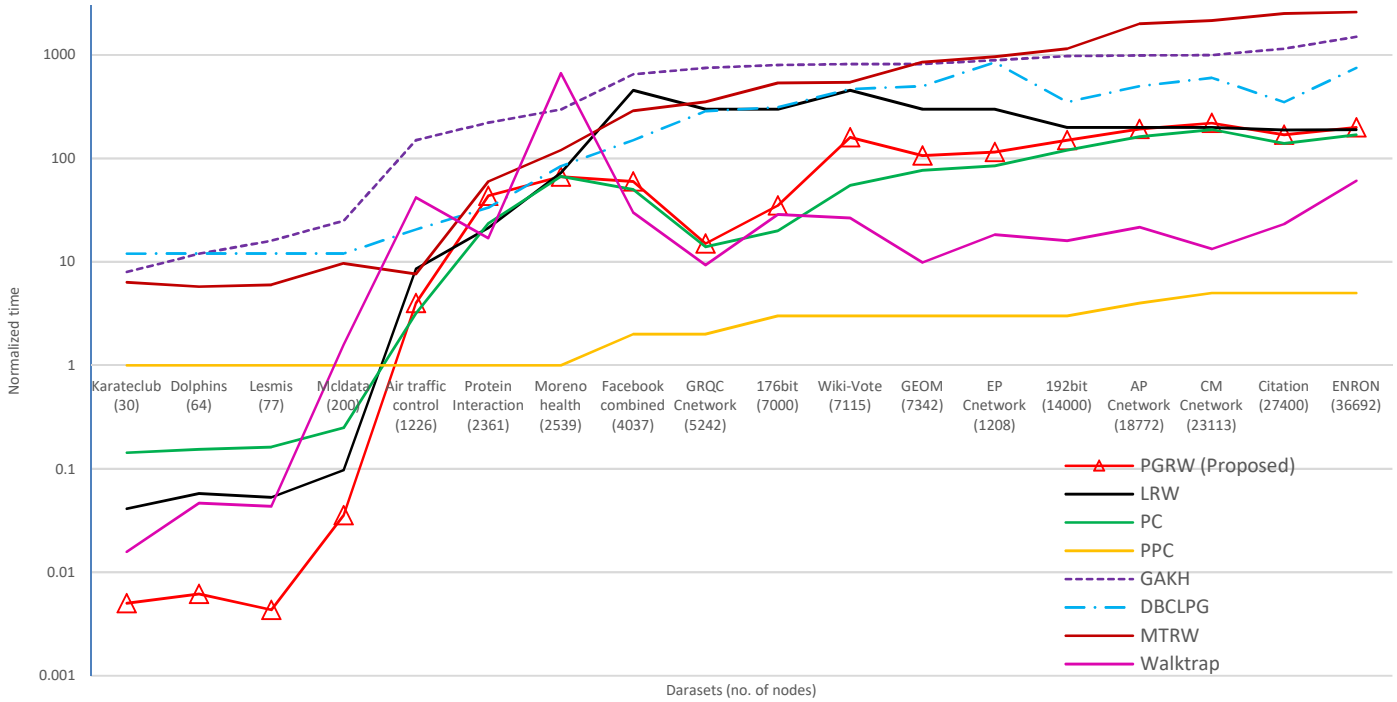

Fig. 10. Results of the eight competing methods using execution time.

dolphins, *lesmis*, and *mcldata*. On the remaining datasets, either LRW or PPC consumes minimum execution time. These two are the parallel programming-based methods and comparing their execution time with the other six methods will not be rational, therefore the remaining results are explained after excluding LRW and PPC. After this normalization, the Walktrap method consumes less time than other clustering methods considered in this work on 11 datasets. Whereas, the proposed method consumes minimum time on five datasets and is ranked second. However, if the average of the time consumed by all the competing methods over 18 datasets is computed, the difference between Walktrap and PGRW is about 1.25 seconds only. The difference between the average time (using the 18 datasets) consumed by six other methods and PGRW and Walktrap is more than 13 seconds.

4.8. Performance comparison using other weight assignment methods

In addition to the evaluation of the proposed technique based on the notion of common neighbors, an experiment is performed using three weight assignment methods. These include, friends of friend (*FoF*) method, Jaccard index (*JI*), and Pearson correlation coefficient (*r*) [41]. The common neighbor notion also called the *FoF* is already explained in the preceding section. The Jaccard's index in Eq. (9) computes dissimilarity between sample sets. The value of *JI* is obtained by dividing the difference of the sizes of the union and the intersection of two sets by the size of the union. Its value ranges between 0 and 1.

$$JI\ (A,B) = \frac{|A \cap B|}{|A \cup B|} \tag{9}$$

The *r* (Eq. (10)) is a measure of the linear correlation between two variables X and Y. The value of *r* ranges between +1 and −1, where 1 is total positive linear correlation, 0 is non-linear correlation, and −1 is a total negative linear correlation.

$$r = \frac{\sum XY - n\overline{XY}}{\sqrt{\sum X^2 - n\overline{X}^2}\sqrt{\sum Y^2 - n\overline{Y}^2}} \tag{10}$$

Fig. 11 shows the result of the this experiment by plotting values of DBI, DI, CHI, and SC for the air traffic control and yeast datasets using *FaF*, *JI*, and *r* to assign weights. It can be seen that *FoF* performs better than the other two indices. However, the performance difference between *JI* and *FoF* is marginal and JI shows the second best results. The proposed approach actually guides the random walk in forming appropriate clusters. Therefore,

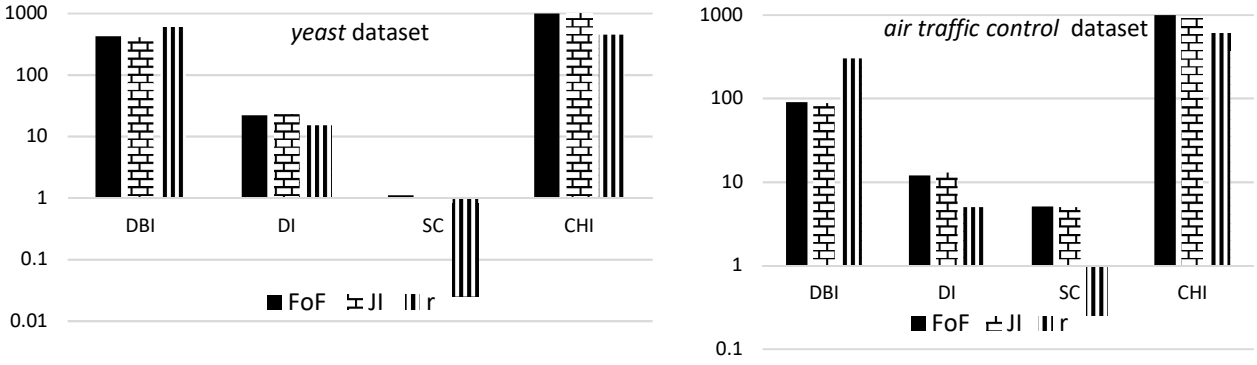

Fig. 11. Performance comparison using other weight assignment methods.

Table 4
Null hypothesis with their alternatives.

Null hypothesis	Alternate hypothesis
H_{10}: The proposed method, PGRW, does not perform better on the six cluster validity indices.	H_{1A}: The proposed method, PGRW, does perform better on the six cluster validity indices.

Table 5
Ranking of the competing approaches.

Datasets	PGRW	LRW	PC	PPC	GAKH	DBCLPG	MTRW	Walktrap
Karateclub	0	0.666667	0	0.166667	0	0	0.166667	0
Dolphins	0.333333	0.166667	0	0.166667	0	0	0.166667	0.166666667
Lesmis	0	0.5	0	0.166667	0.166667	0	0	0.166666667
Mcldata	0	0.5	0	0.333333	0.166667	0	0	0
Air traffic control	0	0.333333	0	0.166667	0	0	0.166667	0.333333333
Protein Interaction	0.5	0	0	0	0	0.166667	0.166667	0.166666667
Moreno health	0.5	0	0	0	0	0.166667	0.166667	0.166666667
Facebook combined	0.666667	0	0.166667	0	0	0	0.166667	0
GRQC Cnetwork	0.5	0	0	0	0	0.166667	0.166667	0.166666667
176bit	0.166667	0	0	0	0	0.333333	0	0.5
Wiki-Vote	0.5	0	0	0.166667	0	0	0.166667	0.166666667
GEOM	0.333333	0	0	0	0	0.166667	0.166667	0.5
EP Cnetwork	0.333333	0.333333	0	0.166667	0.166667	0	0	0.5
192bit	0.166667	0	0	0	0	0.333333	0	0.166666667
AP Cnetwork	0.5	0	0	0	0	0.166667	0	0.166666667
CM Cnetwork	0.333333	0	0	0	0	0	0	0.333333333
Citation	0.666667	0.166667	0	0	0	0	0	0.166666667
ENRON	0.166667	0.166667	0	0.166667	0.166667	0.166667	0	0.166666667
Average	0.314815	0.157407	0.009259	0.083333	0.037037	0.092593	0.083333	0.212962963
Std. dev.	0.227869	0.217474	0.039284	0.103058	0.071299	0.117465	0.085749	0.159713342

Table 6
Paired sample *t*-test results.

	Paired differences					T	df	Sig. (2-tailed)
	Mean	Std. deviation	Std. error mean	95% confidence interval of the difference				
				Lower	Upper			
Pairs								
PGRW-LRW	0.1574074	0.0103956	0.0024503	0.1522373	0.1625775	1.4995695	17	0.1520677
PGRW-PC	0.3055556	0.1885857	0.0444501	0.2117659	0.3993452	4.852507	17	0.0001494
PGRW-PPC	0.2314815	0.1248116	0.0294184	0.1694087	0.2935543	2.9676999	17	0.0086284
PGRW-GAKH	0.2777778	0.1565707	0.0369041	0.1999102	0.3556453	3.9392934	17	0.0010576
PGRW-DBCLPG	0.222222	0.110404	0.0260225	0.1673148	0.2222222	2.7301305	17	0.0142488
PGRW-MTRW	0.231481	0.14212	0.033498	0.1608006	0.3021623	3.1314864	17	0.0060798
PGRW-Walktrap	0.101852	0.068156	0.016065	0.0679557	0.135748	1.1149124	17	0.2804019

there can be instances where *JI's* performance is better than *FoF* if the pseudo-guided random walk is executed multiple times. The reason for *JI* and *FoF* performing so closely is the fact that both actually computes cohesions between two nodes of a graph to estimate similarity.

5. Statistical significance

In the statistical analysis, the term significant means something quite different. It is important to see the significant difference between the proposed clustering approach and the seven other methods. Therefore, the paired sample *t*-test is used here to see if the outcome of the proposed solution is statistically significant as compared to the seven related state-of-art algorithms. For this, first, the null hypothesis (H_{10}) and its alternative (H_{1A}) hypothesis is defined in Table 4. The performance evaluation of the proposed work with competing algorithms is based on the six cluster validity metrics. Therefore, the statistical test is performed here based on these. The level of significance α, a probability to reject the null hypothesis, is set to 5%. Where, 95% confidence level (1-α) refers to the probability of accepting the null hypothesis. The degree of freedom df presents the total number of datasets, i.e., 18. The probability value (i.e., *p*-value) determines the evidence to reject the null hypothesis. A small *p*-value show more evidence in favor of the alternative hypothesis.

This section statistically investigates if the proposed solution does not perform better using the six cluster validity indices, i.e., DBI, DI, SC, CHI, modularity index, and normalized cut. For this, *t*-statistic is computed for each pair. Therefore, first, a score is assigned to each approach for all datasets using the six cluster validity indices. The score shows that an approach performs better on how many metrics out of the six cluster validity indices for a given dataset. For example, if an approach performs better for three validity indices on a given dataset, then the score will be 3 out of 6. Table 5 present all assigned scores. The result of the paired sample *t*-test is shown in Table 6 using the data mentioned in Table 5. Table 6 describe the significant difference between the PGRW and PC [$t(17) = 4.852507$, $p < 0.05$)], PPC[$t(17) = 2.9676999$, $p < 0.05$)],GAKH[$t(17) = 3.9392934$, $p < 0.05$)], DBCLPG [$t(17) = 2.7301305$, $p < 0.05$)] , and MTRW [$t\ (17) = 3.1314864$, $p < 0.05$)]. Based on the obtained *p*-value, the null hypothesis is rejected in favor of the alternative hypothesis. However, there is no significant difference between PGRW and LRW [$t\ (17) = 1.4995695$, $p = 0.1520677$)], Walktrap [$t\ (17) = 1.1149124$, $p = 0.28040188$). These methods perform, very close to each other.

6. Discussion

Many real-world problems generate data that has intrinsic relationship between their entities. Such data can be represented as a graph [42, 43]. This is because graphs provide a rich mapping of many real-world problems for data presentation to the modern computing devices. It consists of nodes and edges with their domain specific interpretation. For instance, nodes in a social network can represent persons, in a communication networks these can be computers or a server, in biological experiment nodes may represent proteins. Finding communities in such large graphs is an active area of research. This task is commonly known as clustering.

6.1. Brief synopsis

This work has addressed the problem of clustering graphs using a random walk-based approach. The random walk here was directed by a pseudo-guidance mechanism towards better clustering formation. The unsupervised nature of the clustering task makes the job challenging. The goal here was to form communities in a graph based

on commonalities, instead of utilizing the simple link-based mechanism. Additionally, the relationship between two nodes in a graph is not always reciprocal. For example, in a social network, if a celebrity is followed by a fan then it does not mean that the fan is also being followed by the celebrity. This example is also relevant for a citation graph. Therefore, additional parameters linking nodes together are always required. However, this interpretation is usually domain specific. The solution presented in this work (PGRW) utilized the friends-of-friends concept to assign weights to the graph's edges. Later, the edges' weights were utilized to find node weight. The node having maximum weight was marked as the seed node to grow the clusters. The cluster growth was managed using random walk. However, this walk was guided towards the nodes where a non-zero edge strength was available. Random walk has produced good results in studies like [44] and [16]. Guiding the random walk procedure here, further improved the results. The proposed approach in this work was compared with seven graph clustering methods, namely, limited random walk (LRW), pairwise clustering (PC), personalized page rank clustering (PPC), GAKH (genetic algorithm krill herd) graph clustering, mixing time of random walks, density-based clustering of large probabilistic graphs, and Walktrap. The comparison here was based on six cluster validity and other indices. These included: Davies-Bouldin index (DBI), Dunn index (DI), Silhouette coefficient (SC), Calinski-Harabasz index (CHI), modularity index, and normalized cut. Further, the experiments were performed using 18 benchmark datasets, namely, *karateclub, dolphins, lesmis, mcldata, air traffic control, Protein interaction, moreno health, facebook combined, GRQC Cnetwork, 176bit, wiki-Vote, GEOM, EP Cnetwork, 192bit, AP Cnetwork, CM Cnetwork, citation,* and *ENRON*. Utilization of six validity indices, execution time, and 18 datasets helped in averaging out the results and reach generalized findings.

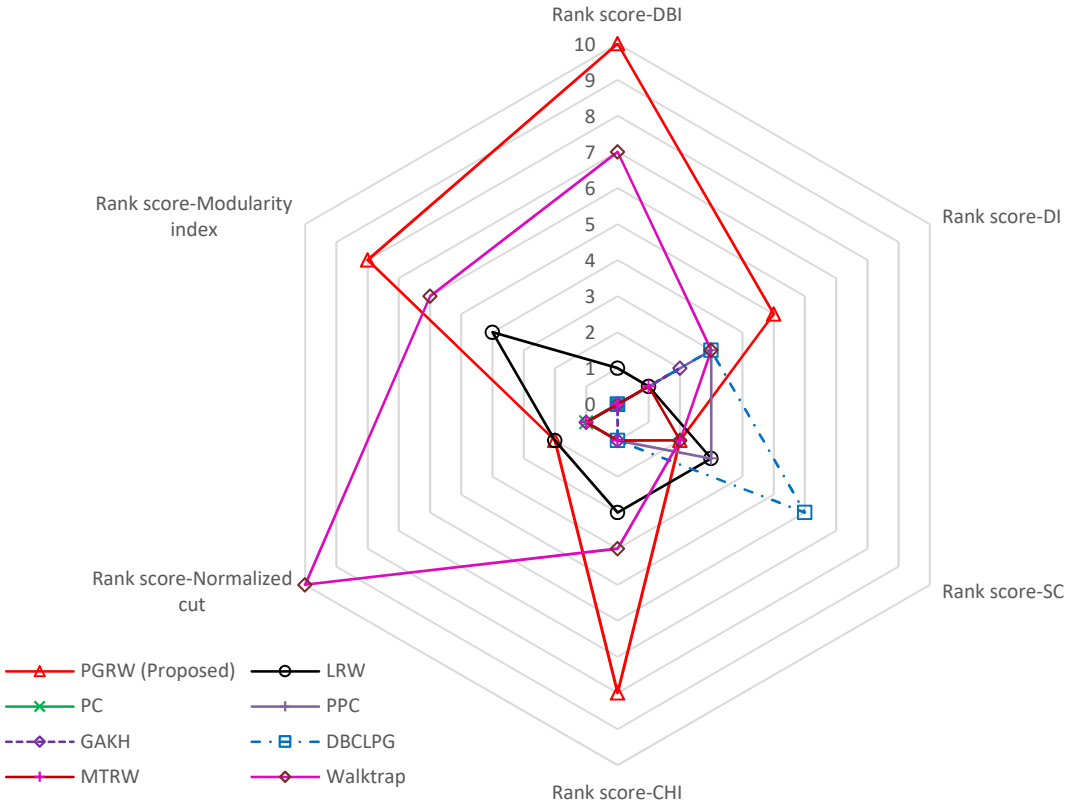

Fig. 12. Ranking scores of the eight competing methods.

6.2. Results analysis

This proposal was compared with seven related state-of-the-art clustering approaches using 18 benchmark datasets based on time and six cluster validity indices. The proposed solution performed better than the other clustering methods on 55.55% of the datasets based on DBI as the validity index. For the same metric, the second best performer was Walktrap having better performance in 38% instances. Using the DI as a metric, the proposed solution performed better than others in the majority of the cases, i.e., 27.33% instances. The second best performance has been observed for two methods, namely, PPC and Walktrap having performed better in 16.66% cases each. The results obtained for SC as a cluster validity index show somewhat different ranking in comparison to the ones obtained using DBI and DI. Here, DBCLPG has been the best performer achieving the optimum clustering formation in 33.33% cases. This is followed by LRW and PPC methods in the second place. The PGRW and Walktrap methods which got 1st or 2nd slot based on DBI and DI, share third position based on SC having better performance in 11.11% instances each.

While using CHI as a cluster validity index, the PGRW has performed better than others in majority cases, i.e., in 44.44% instances. Whereas, the Walktrap has performed better in 22.22% cases thus achieving second slot. The clustering method LRW is the third best based on CHI by producing optimum cluster formation in 16.66% instances. Normalized cut is a graph specific quality measure and based on it the Walktrap has performed better on the majority of the datasets, i.e., in 55.55% cases. Whereas, PGRW has been in the second place with better performance on 11.11% datasets. Finally, based on modularity index, the proposed approach again performed better in 44.44% cases and this achieves first rank. Likewise, the previous indices, Walktrap has been the second best performer here by achieving better results than others in 33.33% cases. Fig. 12 lists a ranking score achieved by each of the competing methods on the six cluster validity indices over the 18 datasets. A higher rank score indicates better overall performance. It can be seen that the proposed PGRW clustering method performance is better in the majority of the cases. The method Walktrap has been the second best and its performance is very close to the proposed method.

6.3. Synthesis

The proposed approach addressed the task of clustering graphs keeping in view the time and space constraints. The current proposal utilized an adjacency matrix to store the graph. This causes the space complexity of PGRW to be $O(n^2)$. Where, n is the number of nodes in the graph. However, internally each cell of the adjacency matrix is of a structure type to store clustering related information about each node. This has a constant time effect on the storage requirement. Therefore, the space requirement of PGRW remains $O(n^2)$. The overall execution time of the proposed approach is dominated by the $O(cn^2)$ bound. Where, n is the number of nodes in the graph and c is the number of clusters. The PGRW approach initiated a walk from the highest degree node. The path of the walking agent was a pure random process limited to the nodes connected via edges having weight greater than zero. This process has a certain probability that once a walk has been initiated, it may have different results if repeated. This will cause slightly different cluster formation if the process is repeated. This is a limitation of the current proposal. However, it is also true for other random walk-based clustering methods. Fig. 12 shows the

execution time of the three competing approaches for various numbers of nodes. The execution time of PGRW is slightly at a higher side, however, PGRW produces 46% better clusters consuming 5% addition time. Experiments performed on larger datasets suggest that the performance of the proposed approach is comparable to other methods (excluding those written using parallel programming paradigm). However, it must be mentioned here that the performance of the proposed work is not only dependent on the nodes in the input graph, but it varies due to the graph density. For example, for a graph with 1000 nodes and edge connection density around 60% the proposed approach may take less execution time than for a graph with 600 nodes and edge connection density around 95%. This is because the random walking agent jumps from one node to the other using the edges resulting in increased execution time. The closest competitor to the proposed method has been the Walktrap methods and the same has been confirmed through the *t*-test.

7. Conclusion

This work proposed a novel random walk-based approach to cluster the data represented as a graph. Extracting clusters from big graphs is an active area of research. The proposed solution addressed this problem by presenting a pseudo-guided random walk procedure to find strongly connected groups. The solution presented here is named Pseudo-Guided Random Walk (PGRW). Initially, the input dataset was transformed into a graph where nodes represented the objects and edges represented the link between them. The PGRW started with finding the significance of connections between nodes by identifying their common friends in the network. Instead of the raw edge weight mentioned in the datasets, the count of such common friends, called the friends-of-friends, was then placed as an edge weight between two nodes. This enabled to identify strongly connected nodes based on commonalities. These weights were then utilized to find the most influential node in the graph. The pseudo-guided random walk started from the node having maximum weight. The walking agent randomly picked any of its neighbours to be visited, however, this choice was limited to only those nodes where the edge weigh based on friends-of-friends concept was greater than or equal to a threshold. The traversed neighbours in this manner were then added to the same cluster. This process stopped when the agent could not find any node to visit. The visited nodes were then extracted from the original graph and the same process was repeated for the rest of the network. The proposed approach was compared with seven closely related graph clustering methods from the same domain, namely, limited random walk, pairwise clustering, personalized page rank clustering, GAKH (genetic algorithm krill herd) graph clustering, mixing time of random walks, density-based clustering of large probabilistic graphs, and Walktrap. The comparison was made using six cluster validity and other indices; Davies-Bouldin index (DBI), Dunn index (DI), Silhouette coefficient (SC), Calinski-Harabasz index (CHI), modularity index, normalized cut, and execution time. For experiments, 18 real-world benchmark datasets were utilized. Results suggested a better performance of the proposed approach in the majority of the cases.

The random walk-based approaches have shown promising results in clustering complex large networks. This work can be extended in multiple ways in the future. Parallel computing paradigm is an option where multiple walking agents can be initiated simultaneously. This shall produce results quickly. The current proposal has assigned edge weights based on the friends-of-friends concept which again utilizes common friends using connectivity. In the future, these common friends can be identified using various attributes of the graph other

than the connectivity aspact. There are many optimization approaches available, like genetic algorithms, ant colony algorithm, and evolution strategy. In the future, it would be interesting to see how these techniques optimizes the clusters extracted from a graph using pseudo-guided random walk.

Acknowledgment

(Blinded as per journal policy)

References

[1] H.L. Bodlaender, A partial k-arboretum of graphs with bounded treewidth. Theoretical computer science, 209(1), 1998, pp. 1-45.
[2] G. Engels, A. Schürr, Encapsulated hierarchical graphs, graph types, and meta types. Electronic Notes in Theoretical Computer Science, 2, 1995, pp. 101-109.
[3] R.O. Duda, P.E. Hart, D.G. Stork, Pattern classification. John Wiley and Sons, 2012.
[4] A.A. Diwan, S. Rane, S. Seshadri, S. Sudarshan, Clustering techniques for minimizing external path length. In VLDB, Vol. 96, 1996, pp. 3-6.
[5] A.J. Enright, S. Van Dongen, C.A. Ouzounis, An efficient algorithm for large-scale detection of protein families. Nucleic acids research, Vol. 30, No. 7, 2002, pp. 1575-1584.
[6] H. Zhang, J. Raitoharju, S. Kiranyaz, M. Gabbouj, Limited random walk algorithm for big graph data clustering. Journal of Big Data, Vol. 3, No. 1, 2016, pp. 26.
[7] L. Yen, D. Vanvyve, F. Wouters, F. Fouss, M. Verleysen, M. Saerens, Clustering using a random walk based distance measure. In Proceedings of the 13th Symposium on Artificial Neural Networks, 2005, pp. 317–324.
[8] K. Avrachenkov, V. Dobrynin, D. Nemirovsky, S.K. Pham, E. Smirnova, Pagerank based clustering of hypertext document collections. In Proceedings of the 31st annual international ACM SIGIR conference on Research and Development in Information Retrieval, 2008, pp. 873-874.
[9] S. Kale, C. Seshadhri, Combinatorial approximation algorithms for maxcut using random walks. 11th Annual Conference on Innovation and Technology in Computer Science Education, 2011, pp. 367–388.
[10] Z. Halim, Uzma, Optimizing the minimum spanning tree-based extracted clusters using evolution strategy. Cluster Computing, 21(1), 2018, pp. 377–391.
[11] D. Harel, Y. Koren, On clustering using random walks. In FSTTCS, 2001, pp. 18-41.
[12] M. Girvan, M.E. Newman, Community structure in social and biological networks. Proceedings of the national academy of sciences, Vol. 99 No. 12, 2002, pp. 7821-7826.
[13] H. Zhou, Distance, dissimilarity index, and network community structure. Physical review e, Vol. 67, No. 6, 2003, pp 061901.
[14] H. Zhou, R. Lipowsky, Network brownian motion: A new method to measure vertex-vertex proximity and to identify communities and subcommunities. In Lecture Notes in Computer Science, Vol. 3038, 2004, pp. 1062–1069.
[15] L. Hagen, A.B. Kahng, A new approach to effective circuit clustering. In Proceedings of the 1992 IEEE/ACM international conference on Computer-aided design, 1992, pp. 422-427.
[16] P. Pons, M. Latapy, Computing communities in large networks using random walks. In ISCIS Vol. 3733, 2005, pp. 284-293.
[17] Y. Hu, M. Li, P. Zhang, Y. Fan, Z. Di, Community detection by signaling on complex networks. Physical Review E, Vol. 78 No. 1, 2008, pp. 016115.
[18] E. Weinan, T. Li, E. Vanden-Eijnden, Optimal partition and effective dynamics of complex networks. Proceedings of the National Academy of Sciences, Vol. 105, No. 23, 2008, pp. 7907-7912.
[19] Z. Halim, M. Waqas, A.R. Baig, A. Rashid, Efficient Clustering of Large Uncertain Graphs Using Neighborhood Information, International Journal of Approximate Reasoning, Vol. 90, 2017, pp. 274-291.
[20] S.M. Van Dongen, Graph clustering by flow simulation, Ph.D. Thesis, Dutch National Research Institute for Mathematics and Computer Science, University of Utrecht, Netherlands, 2000.
[21] X. He, H. Zha, C.H. Ding, H.D. Simon, Web document clustering using hyperlink structures. Computational Statistics and Data Analysis, Vol. 41, No. 1, 2002, pp. 19-45.
[22] J. Shi, J. Malik, Normalized cuts and image segmentation. IEEE Transactions on pattern analysis and machine intelligence, Vol. 22, No. 8, 2000, pp. 888-905.
[23] M. Meila, J. Shi, Learning segmentation by random walks. In Advances in neural information processing systems, 12, 2001, pp. 873-879.
[24] D. Auber, M. Delest, Y. Chiricota, Strahler based graph clustering using convolution. In Information Visualisation, 2004. IV 2004. Proceedings. Eighth International Conference on Information Visualization, pp. 44-51. 2004.
[25] B. Yang, J. Liu, An efficient probabilistic approach to network community mining. Rough Sets and Knowledge Technology, Vol 1, No. 1, 2007, pp. 267-275.
[26] M. Pavan, M. Pelillo, Dominant sets and pairwise clustering. IEEE transactions on pattern analysis and machine intelligence, Vol. 29, No. 1, 2007, pp. 167-172.
[27] J. Hopcroft, O. Khan, B. Kulis, B. Selman, Natural communities in large linked networks. In Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, 2003, pp. 541-546.
[28] M.E. Newman, Fast algorithm for detecting community structure in networks. Physical Review E, Vol. 69, No. 6, 2004, pp. 066133.
[29] P.S. Bradley, U.M. Fayyad, C. Reina, Scaling Clustering Algorithms to Large Databases, In KDD, 1998, pp. 9-15.
[30] G.T. Toussaint, Proximity graphs for nearest neighbor decision rules: recent progress. In Interface 2002, 34th Symposium on Computing and Statistics, 2002, pp. 1-6
[31] H. Zanghi, C. Ambroise, V. Miele, Fast online graph clustering via Erdos–Rényi mixture. Pattern recognition, Vol. 41, No. 12, 2008, pp. 3592-3599.

[32] M. Akbari, H. Izadkhah, GAKH: A new evolutionary algorithm for graph clustering problem. In 2019 4th International Conference on Pattern Recognition and Image Analysis (IPRIA), 2019, pp. 159-162.
[33] K. Avrachenkov, M. El Chamie, G. Neglia, Graph clustering based on mixing time of random walks. In IEEE International Conference on Communications, 2014, pp. 4089-4094.
[34] Z. Halim, J.H. Khattak, Density-based clustering of big probabilistic graphs. Evolving Systems, vol. 10, no. 3, 2019, pp.333-350.
[35] R. Morisi, G. Gnecco, A. Bemporad, A hierarchical consensus method for the approximation of the consensus state, based on clustering and spectral graph theory. Engineering Applications of Artificial Intelligence, 56, 2016, pp. 157-174.
[36] S. Guha, R. Rastogi, K. Shim, CURE: an efficient clustering algorithm for large databases. In ACM Sigmod Record, Vol. 27, No. 2, 1998, pp. 73-84.
[37] S. Guha, R. Rastogi, K. Shim, ROCK: A robust clustering algorithm for categorical attributes. Information systems, Vol. 25 No. 5, 2000, pp. 345-366.
[38] S.A. Tabrizi, A. Shakery, M. Asadpour, M., Abbasi, M.A. Tavallaie, Personalized pagerank clustering: A graph clustering algorithm based on random walks. Physica A: Statistical Mechanics and its Applications, Vol. 392, No. 22, 2013, pp. 5772-5785.
[39] E.E. Papalexakis, L. Akoglu, D. Ience, Do more views of a graph help? Community Detection and Clustering in Multi-graphs. In 16th international conference on Information fusion (FUSION), 2013 pp. 899-905.
[40] W.W. Zachary, An information flow model for conflict and fission in small groups. Journal of anthropological research, Vol. 33, No. 4, 1977, pp. 452-473.
[41] S. Daminelli, J.M. Thomas, C. Durán, C.V. Cannistraci, Common neighbours and the local-community-paradigm for topological link prediction in bipartite networks. New Journal of Physics, Vol. 17, No. 11, 2015, pp. 113037.
[42] Z. Halim, O. Ali, and G. Khan, On the Efficient Representation of Datasets as Graphs to Mine Maximal Frequent Itemsets. IEEE Transactions on Knowledge and Data Engineering, pp. 1-18, 2019.
[43] S. Iqbal, Z. Halim, Orienting Conflicted Graph Edges Using Genetic Algorithms to Discover Pathways in Protein-Protein Interaction Networks. IEEE/ACM Transactions on Computational Biology and Bioinformatics, pp. 1-16, 2020.
[44] C. Sherlock, A.H. Thiery, G.O. Roberts, J.S. Rosenthal, On the efficiency of pseudo-marginal random walk Metropolis algorithms. The Annals of Statistics, Vol. 43, No. 1, 2015, pp. 238-275.