IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 24 September 2023, accepted 27 October 2023, date of publication 1 November 2023, date of current version 6 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3329429

RESEARCH ARTICLE

Partitioning Graph Clustering With User-Specified Density

ROHI TARIQ[1], KITTICHAI LAVANGNANANDA[1], (Senior Member, IEEE), PASCAL BOUVRY[2], AND PORNCHAI MONGKOLNAM[1]

[1]School of Information Technology, King Mongkut's University of Technology Thonburi, Bangkok 10140, Thailand
[2]Department of Computer Science, Faculty of Science, Technology and Medicine, University of Luxembourg, 4365 Esch-sur-Alzette, Luxembourg

Corresponding author: Kittichai Lavangnananda (kittichai.lav@gmail.com)

This work was supported in part by the King Mongkut's University of Technology Thonburi (KMUTT) through the Petchra Pra Jom Klao Scholarship under Grant 49/2562; and in part by the Interdisciplinary Centre for Security, Reliability, and Trust (SnT), Faculty of Science, Technology and Medicine, University of Luxembourg.

ABSTRACT Graph clustering has attracted many interests in recent years, with numerous applications ranging from the clustering of computer networks to the detection of social communities. It presents a challenging NP-class problem, and as a result, numerous algorithms have been developed, each tailored to specific objectives and quality metrics for evaluation. This research commences by categorizing existing graph clustering algorithms based on two distinct perspectives: parameter-free algorithms and user-defined or adjustable parametric algorithms. Quality metrics are further categorized into three distinct groups: internal connectivity, external connectivity, and a combination of both. If a task can be represented by a simple undirected and unweighted graph, from a management and deployment of resources perspective, having clusters of some kind of similar density is advantageous as it allows efficient management. This research introduces a partitioning graph clustering algorithm that allows users to specify the desired density of a cluster by means of 'relative density'. Clustering process involves the determination of all triangles (i.e., smallest cliques) and selecting a clique as an initial cluster. The expansion of a cluster is done by adding adjacent cliques while the required relative density is monitored. Existing metrics are found unsuitable for evaluating the proposed method; therefore, a suitable new metric, the Mean Relative Density Deviation Coefficient (MRDDC), is introduced.

INDEX TERMS Graph clustering, mean relative density deviation coefficient (MDRCC), NP problem, partitioning graph clustering, quality metric, relative density.

I. INTRODUCTION

In the present era, the rapid increase in the volume and variety of data has led to the emergence of advanced data techniques aimed at uncovering the latent patterns and structures in complex networks. Among the many prominent techniques available, graph clustering is a vital domain in data science. It involves the grouping of nodes within a graph based on their connectivity patterns or similarity features to gain insight into the underlying structure of the graph [1], [2].

Graph clustering serves a diverse array of purposes across nearly all domains. For instance, identifying communities with similar interests or behaviors in social network analysis references [2], [3], and [4], optimizing routes and traffic flow within transportation networks, uncovering functional gene or protein modules in biological networks, delineating regions with similar texture or color in image segmentation [4], [5], and providing personalized recommendations through recommendation systems [6], [7]. Numerous techniques have been proposed to address a wide spectrum of objectives. Some notable examples encompass K-means [8], DBSCAN [9], Girvan-Newman [10], Louvain [11], among many others. These algorithms differ in terms of their

The associate editor coordinating the review of this manuscript and approving it for publication was Qilian Liang.

input parameters, which can either be presetted to default parameter values, specified by users, or adjusted by users to fine tune the resultant clusters. Selecting appropriate user inputs for clustering algorithms is a challenging but crucial task, given clustering's subjective and application specific nature [12], [13]. Conversely, limited user inputs and domain specificity may also constrain clustering algorithms' usefulness.

Clustering algorithms commonly employ various parameters including the number of clusters, minimum number of neighboring points, resolution parameters, regularization parameters, and convergence criteria [14]. Distance metrics, such as Euclidean distance, cosine similarity, Jaccard distance, and correlation distance, have been used to measure the similarity or dissimilarity between nodes or clusters [15], [16]. For example, hierarchical clustering algorithms use linkage methods, such as single, complete, or average linkage to calculate the distance between merging clusters [17], [18]. In certain community detection algorithms, the resolution parameter is used to control the number and size of communities [19], [20].

Each clustering algorithm employs one of the three types of input parameters: those with preset default values, user-specified parameters, or adjustable parameters that can be fine-tuned for better clustering results. In more complex algorithms, such as spectral clustering and random walk-based methods, advanced knowledge of linear algebra and graph theory may be required to adjust parameters. The process of tuning parameters often involves iterative cycles of experimentation and evaluation, rendering it computationally intensive, particularly when dealing with large and complex networks. Consequently, there is a trade-off between the potential benefits of parameter tuning and an increase in the algorithmic complexity. In addition to computational challenges, unbalanced clustering is another significant issue that can arise when adjusting the algorithmic parameters. In practice, unbalanced clustering, in terms of density and connectivity can lead to inefficient resource utilization, reduced network performance, and increased maintenance costs [21]. In IoT applications, unbalanced clusters can result in energy wastage, missed events or anomalies, and reduced device lifetime. This ultimately affects the reliability and accuracy of these applications [22], [23].

As mentioned previously, the versatility of graph clustering techniques extends across virtually every domain. Similar to numerous other scientific disciplines, the development of graph clustering algorithms is driven by specific objectives inherent to their respective applications. Nevertheless, several requirements in the field of graph clustering remain unfulfilled. One such requirement pertains to the capability to cluster a graph in which the resultant clusters possess the same or very similar density. This capability is highly advantageous for efficient deployment and management of resources across various applications. For example, in a complex graph that represents large communities, sub communities can be seen as clusters within the overall community. If these sub communities are of equal density, an equal deployment of resources to these communities can be efficient because each sub community is of equal size. The same can be said for a complex graph that represents communication among people (e.g., LINE), where it would be advantageous if the network of communication was split into small units of networks with similar density. This may allow for efficient monitoring of each unit, as each unit is likely to demand similar resources. Similarly, during the recent COVID-19 pandemic, interactions among infected patients in a particular city can be represented by a graph. Dividing them into clusters of similar density allows each cluster to be considered relative to the entire network. It may be possible to have clusters with smaller patients (i.e., nodes) but much higher interactions (i.e., edges). A detailed analysis of each cluster may reveal the information of interest. These examples can be transformed into the ability to cluster a well connected graph where the resultant clusters have the same or very similar density. To date, no clustering algorithm allows users to specify the desired density of any type. Hence, this study attempts to partition the clustering of an undirected, unweighted graph such that user(s) can specify the required density. This density is considered the mean 'relative density' and is described in detail in Section V. It is assumed that the clusters in the graph are not apparent or visible. If such circumstances exist, then the application of a graph clustering algorithm may not be necessary.

The remainder of this paper is organized as follows. Section II briefly describes the fundamental terminologies and definitions related to graphs that are consistently utilized throughout this paper. Section III presents a comprehensive overview of the existing related work on graph clustering algorithms. In addition, they were categorized based on the necessity of input parameters. Section IV presents a comprehensive description of the established clustering quality metrics. These metrics, categorized into internal and external connectivity-based measures, aim to facilitate the selection of appropriate evaluation criteria tailored to specific clustering processes and their applications. Section V describes the proposed two-phase graph-partitioning clustering algorithm with user-specified relative density. Definitions, mathematical equations, pseudo-codes, and examples are provided for clarity of the proposed clustering algorithm. Section VI discusses the partitioning clustering results of the eight real-world networks selected using the proposed clustering algorithm. The results of the three networks have diverse characteristics and are illustrated graphically. This section also presents the evaluation process and identifies the shortcomings of the existing quality metrics. Section VII, therefore, introduced a novel quality metric known as the Mean Relative Density Deviation Coefficient (MRDDC). Section VIII discusses the main findings of the study. The paper is concluded in section IX, and possible directions for further work are suggested in section X.

II. PRELIMINARIES

Theoretically, a graph is typically expressed as $G(V, E)$, where V denotes the set of nodes (vertices) and E is the set of edges. A graph is identified based on the edge type. In its simplest form, if an edge ($e \in E$) has no weight and direction, the graph is referred to as an unweighted and undirected graph; otherwise, it is referred to with respect to the type of edge. Clustering is a field within data science in which all data are grouped together to form several clusters. Good clustering is one in which the intra-similarity of each cluster is high and the inter-similarity between clusters is low. As the name implies, graph clustering is a sub field of clustering that specifically deals with data represented in the form of a graph. The clustering of G is a partition of V into disjoint subsets, where each subset formally represents a cluster. Let $C = \{C_1, C_2, \ldots, C_k\}$ be a collection of k subsets of V, where $|C| = k$, and represents the number of clusters.

$\forall v \in V, \exists! i \in \{1, 2, \ldots, k\} : v \in C_i$. Each node belonged to a single cluster.

OR: $\forall i, j \in \{1, 2, \ldots, k\}$ with $i \neq j$, $C_i \cap C_j = \emptyset$. Clusters are non-overlapping.

Graph clustering can be categorized into three types: partitioning, overlapping, and hierarchical [4]. The definition above belongs to partitioning clustering. In overlapping clustering, a node may belong to more than one cluster, and hence overlapping area(s) is (are) identified, whereas several layers may exist in hierarchical clustering. This work is concerned with partitioning clustering. Henceforth, the terms 'clustering' and 'partitioning clustering' are used interchangeably. For a relatively complex graph (i.e., a reasonably large number of nodes and edges), all three types of clustering may be possible. Also, graph clustering is a combinatorial problem that is known to be NP-hard [3], [24]. Fig. 1 illustrates these three types of clustering and the possibility of different clustering results in an arbitrary simple graph.

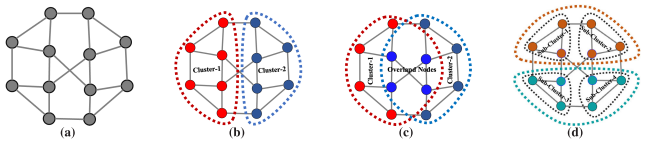

FIGURE 1. (a) Arbitrary simple graph (b) Partitioning clustering (c) Overlapping clustering (d) Hierarchical clustering.

The concept of a cluster's degree encompasses two primary components, internal and external degrees, which can be defined as follows.

1) *Cluster Internal Degree:* The internal degree of a vertex $v \in C_i$, denoted as $int\_deg(v)$, represents the number of edges connecting v to other vertices within C_i. For any vertex $v \in C_i$, $int\_deg(v)$ must be greater than or equal to one ($int\_deg(v) \geq 1$). Therefore, the internal degree of a cluster is calculated as the sum of the internal degrees of all vertices within the cluster, expressed as follows.

$$int_{deg}(C_i) = \sum_{v \in C_i} int_{deg}(v) \tag{1}$$

2) *Cluster External Degree:* Similarly, the external degree of a vertex $v \in$, denoted as $ext\_deg(v)$, is the number of edges connecting v to vertices outside the cluster. For node $v \in C_i$, the external degree $ext\_deg(v) = 0$ indicates that the cluster is highly connected. Hence, the external degree of a cluster C_i is equal to the sum of the out-degrees of its constituent nodes, as follows.

$$ext_{deg}(C_i) = \sum_{v \in C_i} ext_{deg}(v) \tag{2}$$

Therefore, the total degree of a cluster can be defined as the sum of its internal and external degrees, which is formulated as following in (3).

$$tot_{deg}(C_i) = int_{deg}(C_i) + ext_{deg}(C_i) \tag{3}$$

Several traditional methods have been developed for clustering graphs. One common approach is the minimum cut partition, which aims to minimize inter-cluster edges [4]. This approach is similar to graph-partitioning, which is often extended to more than two clusters using iterative bisectioning. Additionally, many cases seek equally sized clusters, resulting in the NP-hard 'minimum bisection' problem [25].

Another popular approach involves identifying patterns within a graph including cliques and motifs. These two terms are sometimes used interchangeably in literature. Cliques are fully connected subgraphs where each node has an edge on every other node; they can be of different sizes (i.e., different numbers of nodes) [26]. Motifs are patterns (or recurring subgraphs(s)) in a graph that occur frequently. Notably, cliques can also be considered motifs; however, not all motifs exhibit clique characteristics. These were used as the basis for clustering graphs [27], [28], [29]. Fig. 2(a)_2(g) illustrates examples of clique and motif structures. Fig 2(a) through (d) illustrate two-node, three-node, four-node, and five-node cliques, respectively. Additionally, Fig. 2 (e), (f), and (g) depict examples of motifs that frequently appear in networks.

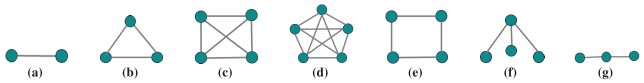

FIGURE 2. Examples of common cliques and motif structures.

In brief, K represents a complete graph and subscript 3 signifies a complete graph comprising three vertices. Consequently, K_3 denotes a clique of size three, specifically a triangle. For brevity, this study adopts the notation K to represent a clique of size three, with the total count of cliques in a graph denoted as $|G(K)|$.

III. CATEGORIZATION OF GRAPH CLUSTERING ALGORITHMS

Numerous taxonomies have emerged in the literature for categorizing graph clustering algorithms that encompass all three fundamental types [30]. Within this field, scholars have proposed classifications based on the exploration of cohesive subgraphs, node clustering techniques, community quality optimization, divisive strategies, and model-based methodologies [19], [31]. Moreover, numerous in-depth surveys have explored the domain of clustering algorithms, including the notable extensive reviews referenced in [4], [32], and [33]. These taxonomies have provided a comprehensive understanding of this field. For instance, as highlighted in [34], clustering algorithms can be broadly categorized into various types such as those based on the center, hierarchical structures, mixture models, graphs, fuzzy logic, and combinatorial search techniques. However, each method relies on a specific set of parameters to detect the clusters in a graph. These parameters can be adjusted, specified by the user, or set to default values by the algorithm, depending on their scope or work.

In this section, a new perspective on categorizing commonly used graph clustering algorithms (GCA) based on the parameters used is proposed. Algorithms are distinguished as those that do not require user-specified parameters (i.e., parameter-free algorithms) and those that rely on user specified or adjustable parameters. This categorization can help users to determine a suitable strategy for clustering and comprehend the underlying structure of the network. Furthermore, this study provides valuable insights into the implementation of various integrated libraries.

A. PARAMETER-FREE ALGORITHMS

Many clustering techniques require pre-existing knowledge or user-defined input parameters, which substantially influence their effectiveness [35], [36]. To address challenge, automatic or parameter-free clustering methodologies have been introduced. In this subsection, we discuss widely used clustering techniques that do not require users to specify parameters but involve technically adjustable parameters such as objective function, convergence, and tolerance. These adjustable parameters have default settings, thereby simplifying the clustering process for the users. Additionally, several libraries provide implementations of these techniques with default settings, thereby simplifying the clustering process and making it more accessible to non-experts.

1) FAST-GREEDY

Fast Greedy (FG) is a modularity optimization approach for agglomerative hierarchical clustering that outperforms other clustering methods in terms of efficiency and scalability, without requiring user-specified parameters [35]. Optimized data structures and strategic optimization shortcuts further enhance their performance [32]. Fast Greedy has been implemented in various libraries, such as the community discovery library (CDlib) in Python, the widely used NetworkX library for complex network analysis, and the igraph software package [37], [38], [39]. These implementations require only the graph structure as an input parameter, making Fast Greedy an accessible and user-friendly option for graph clustering. Its potential applications include analyzing online sales websites and recommender networks, such as Amazon.com.

2) INFO-MAP

Info-Map (IM) is a greedy clustering algorithm that uses information theory principles [40], [41] to identify clusters by analyzing information flows through random walks. It partitions the network into modules, optimizes the transmission rate, and sends signals to a decoder over a channel with limited channel capacity. The clustering quality was measured using the minimum description length, with a shorter length indicating a more compressed cluster structure. Technical customization options require expertise, but libraries such as the community discovery library and igraph software package provide default parameter settings to simplify the algorithm for end_users. Directed networks require random teleportation for ranking, whereas undirected graphs require only one input parameter (graph) and are independent of the teleportation rate [42].

3) LABEL PROPAGATION

The algorithm known as Label Propagation Algorithm (LPA) is a graph clustering technique that achieves clustering by iteratively assigning labels to nodes based on their neighboring nodes until convergence [43]. This approach assumes that nodes within the same cluster are likely to share the same label as most of their neighbors. Although the LPA has the advantage of linear computational complexity, it suffers from randomness. Various extended LPAs have been proposed to improve performance, including memory-based label propagation (MemLPA) [44], modularity, and node importance (LPA-MNI) [45]. However, these extensions do not entirely solve the randomness issue or increase the complexity of the algorithms. LPA is a parameter-free method, but it can be customized using optional parameters such as *max_iter*, which controls the number of iterations that the algorithm will perform before stopping. Parameter *tol* sets the threshold for the change in labels of the nodes between iterations, below which the algorithm is considered to have converged. Several libraries, including CDlib, NetworkX, and igraph, provide easy-to-use interfaces for implementing LPA with default settings [20], [42], [54].

4) WALKTRAP

Walktrap is a hierarchical clustering algorithm that uses random walks over short distances to group nodes within the same cluster [46]. Initially, it starts with a non-clustered partition and then computes the distances between nearby nodes, followed by merging neighboring clusters until

(n-1) iterations have been completed. This technique uses a technical parameter, called, t which determines the number of steps considered in a random walk. The default value for t is 4, but users can set it to any value between 2 and 8. However, it is important to note that (t) is a technical parameter based on the principles of the algorithm, and fine-tuning may not always result in better clustering performance. Most implementations of the algorithm provide a default value for t (e.g., CDlib and igraph), but the optimal value may need to be determined through experimentation or by analyzing the characteristics of the network being clustered.

5) SPINGLASS

The spinglass algorithm is a widely used clustering technique that utilizes the 'Potts model' statistical mechanics [47] to connect nodes with the same spin state and disconnect those with different spin states. The algorithm employs simulated annealing techniques to reduce the resource requirements, as described in [48]. The Spinglass algorithm has several technical and statistical parameters, including the temperature schedule, interaction strength, randomness, initialization, spin update rule, spin glass Hamiltonian formulation, and convergence criteria. These parameters are based on the principles of statistical physics and play a crucial role in optimizing the clustering results. However, it is important to note that these parameters are highly technical and statistical in nature and their optimal values may not be immediately obvious. Several software packages offer implementations of the spinglass clustering algorithm and provide the default settings. Popular examples include NetworkX, igraph, Gephi, and SciPy are Python libraries for scientific computing which include modules for clustering and community detection.

6) LEADING EIGENVECTOR

Several solutions have been proposed to identify clusters in graphs with high edge density. One popular method is to maximize the modularity benefit function over all possible network divisions. This approach provides a robust solution to clustering challenges. An alternative method was proposed by Newman [49]. The modularity function is rewritten in matrix form, allowing the optimization task to be expressed as a spectral problem in linear algebra. The Leading Eigenvector Algorithm (LEA) is a popular implementation of this approach. It computes the leading eigenvector of the modularity matrix and partitions the graph into two parts to maximize the modularity improvement. Although the algorithm itself does not have inherent parameters, specific parameters may be applied when it is used for graph clustering via modularity maximization, such as the resolution parameter, stopping criteria, and method for computing the modularity matrix. Many libraries, including NetworkX, igraph, and Gephi, include LEA as part of their default graph clustering routines.

7) GRAPH-BASED K-MEANS

Graph-based k-means clustering is a method proposed to overcome the limitations of traditional k-means when applied to graph-structured data. The approach utilizes the minimum spanning tree of the graph to automatically estimate k and initialize the placement of centroids [50]. This method has been demonstrated to outperform standard k-means on synthetic and real-world datasets. It demonstrates enhanced stability against noise and outliers. Additionally, a theoretical analysis of the proposed algorithm shows that it has a lower bound on the number of iterations required to converge to a solution. Overall, this novel approach to graph clustering provides valuable insight into the theoretical properties of the algorithm and improves upon traditional k-means clustering.

B. PARAMETERIZABLE ALGORITHMS

Clustering is a complex and subjective process, particularly when dealing with real-world networks, and unsupervised clustering approaches may not always meet users' precise requirements or expectations [12]. Studies have demonstrated that even a single user-specified input value can significantly enhance clustering results by aligning them more closely with the specific requirements of the user [33]. Advanced graph clustering techniques in community detection allow users to control certain parameters by specifying or adjusting them. This customization capability directly affects the number and size of the resulting clusters. The following subsection presents a comprehensive overview of the frequently employed graph clustering algorithms that utilize adjustable or user-specified parameters. These algorithms are implemented in renowned libraries such as Scikit-learn, NetworkX, CDlib, and igraph. These libraries offer users the flexibility to choose between default parameter settings or customize them according to their specific requirements, thereby allowing the fine-tuning of the clustering process.

1) GIRVAN-NEWMAN

Girvan and Newman proposed a graph clustering algorithm that uses edge_betweenness to detect clusters in complex networks [10]. The algorithm constructs a full dendrogram by removing edges with the highest betweenness value, providing a hierarchical structure and a way to determine the optimal level-cut through modularity [47]. The Girvan-Newman algorithm is included in several Python libraries, including NetworkX, igraph, and CDlib, allowing users to customize the parameters for optimal clustering results.

The level-cut parameter is critical because it directly affects the number and size of the clusters. In NetworkX, the `community.girvan_newman` function allows users to set the desired number of clusters using argument k, and the algorithm terminates once it reaches this number. By contrast, the clustering method in igraph requires argument n to denote the desired number of clusters, and the algorithm stops accordingly. CDlib sets the level cut to a default value of three, but allows users to adjust this parameter based on

their specific needs. Overall, the Girvan-Newman algorithm is widely used and effective, with customizable parameters available in various libraries to tailor the clustering results to the user requirements.

2) LOUVAIN

The louvain algorithm, which is recognized as a cutting-edge graph clustering method, is commonly referred to as multilevel clustering. It derives its name from the place where it originated. The algorithm optimizes modularity by performing local node movement and network aggregation through 'passes' [11]. Although it has been widely recognized for its speed and efficiency [32], [51], it has been found to have limitations in maximizing modularity, making it difficult to identify small communities [52]. The Louvain algorithm with resolution limit adjustment is available in popular Python network analysis libraries, such as CDlib, igraph, and NetworkX. This requires an input graph and adjustable resolution parameter (γ). Lower values of γ produce fewer and larger clusters, whereas the default value of $\gamma = 1$ retrieves the original modularity definition used by the algorithm.

3) LEIDEN

The Leiden algorithm is an advanced clustering method that builds on the Louvain algorithm [9]. One of the limitations of the Louvain method is the possibility of producing poorly connected clusters, which can be overcome by the Leiden algorithm. This is particularly relevant when dealing with modularity and the constant Potts model (CPM) beyond the resolution limit. The Leiden algorithm integrates a range of previous enhancements and employs a combination of fast local moves, smart local moves, and random neighbor moves. The Leiden algorithm guarantees the identification of partitions in which all clusters are internally connected. It also converges to a stable partition, where subsets of clusters are optimally assigned, thereby providing an upper bound for the quality of an optimal partition. The integration of the Leiden algorithm into popular network analysis libraries in Python, such as igraph and CDlib, enables users to leverage their capabilities. These libraries offer flexibility by allowing users to adjust the resolution parameters of the algorithms according to their specific requirements.

4) FLUID-COMMUNITIES

The fluid community algorithm partitions a graph into communities using virtual fluid [53]. It has been successfully applied to social network analysis, bioinformatics, and recommender systems. The algorithm updates node community affiliations using differential equations that guide the movement of fluids between nodes. These updated rules encourage the fluid to flow towards nodes with similar affiliations and discourage it from flowing towards nodes with dissimilar affiliations. The NetworkX Python library includes the implementation of the fluid community algorithm, which is accessible through the `(algorithms.community.asyn_fluidc.asyn_fluid)`. The function uses a graph and optional estimate of the number of communities (k) to be detected as inputs. It is important to note that the (k) does not guarantee the exact detection of exactly k communities, as the algorithm dynamically determines the number of communities based on the topology of the graph and other function parameters. Overall, the Fluid Communities algorithm provides a powerful and flexible approach to community detection, and its implementation in NetworkX makes it easily accessible to end users.

5) SPECTRAL METHOD

Spectral clustering method is a popular modern clustering method that uses information from the eigenvalues of special matrices constructed from a graph to identify clusters [54], [55]. It outperforms conventional clustering algorithms, such as single linkage or k-means clustering, and is widely used in many graph-based clustering algorithms. Spectral clustering is versatile and can be applied to non-graph data; however, constructing a similarity graph for data points is challenging. Determining an appropriate similarity graph and its parameters, particularly k or ε, is difficult for datasets with clusters of varying densities and arbitrary shapes. Consequently, the clustering outcomes are sensitive to the choice of the similarity graph and its parameters. Several well-known libraries, including Scikit-learn, NetworkX, igraph, and MATLAB, offer spectral clustering implementations that allow users to define the number of clusters (k) as an input parameter.

6) EXPECTATION MAXIMIZATION

The expectation maximization (EM) algorithm, introduced in [56], is a model-based approach for clustering data. It utilizes probability distributions to estimate the parameters of the underlying data-generation process. The EM technique has two main steps: the E-step estimates missing data values, such as cluster assignments for each observation, using the observed data and current parameter estimates, and the M-step updates the parameter estimates based on the complete data, maximizing the expected complete-data log-likelihood. The EM has several parameters, including the initialization method, convergence criteria, and the maximum number of iterations. Numerous community discovery libraries, such as CDlib, offer user-friendly implementations that simplify their usage in various practical applications. Users can easily specify the desired number of clusters (k) as input to the function or method.

Fig. 3 depicts a summary of the categorization of graph clustering algorithms (GCA) with respect to parameter-free and parameterizable.

As discussed in this section, numerous graph clustering algorithms exist, and they were invented for general purposes, as well as with specific objectives in mind. These can be

TABLE 1. Summary of algorithm features, strengths, and limitations.

Algorithm	Feature	Strength	Limitation
Fast-Greedy	Modularity optimization	Efficient and scalable, User-friendly default settings	Requires technical customization
Info-Map	Greedy clustering using information theory principles	Simplified default parameters, No setting is requited for undirected networks	Technical expertise needed for customization
Lable Propagation	Iterative node labeling	Parameter-free method, Customizable with optional parameters	Sensitive to initial conditions and may produce different results for the same graph
Walktrap	Random walks over short distances	Ease of usage due to default parameter settings	Choice of the number of steps (t) may not always lead to improved clustering performance
Spinglass	'Potts model' statistical mechanics	Widely used clustering technique, Provide both default settings and fine-tuning options for optimization	Requires statistical knowledge for tuning
Leading Eigenvector	Modularity maximization as a spectral problem in linear algebra	Utilizes modularity for community detection, Computes principal eigenvector	Good understanding of modularity, optimization and linear algebra are necessary to ensure good performance
Graph-based k-means	Overcomes limitations of traditional k-means on graph-structured data	Outperforms standard k-means on various datasets, Offering enhanced stability and theoretical analysis	May require more computational resources than traditional k-means in some cases
Girvan-Newman	Edge-betweenness	Customizable parameters for adjusting the number of clusters	Level-cut choice impacts the number and size of clusters
Louvain	Modularity optimization through local node movement and network aggregation	Customizable resolution, Fast and efficient, Effective for large networks	Limitation in maximizing modularity making it difficult / inappropriate to identify small communities
Leiden	Improved Louvain	Adjustable resolution to improve results, Effective for resolving poorly connected clusters	Adjusting suitable resolution parameter may be necessary for specific cases
Fluid-Communities	Uses virtual fluid to partition the graph into communities	Providing flexible number of communities based on topology of the graph	Exact number of communities is not guaranteed as determination is dynamically based on the graph topology
Spectral Method	Utilizes eigenvalues of matrices constructed from the graph	Adaptable to non-graph data, Users can specify the number of required clusters (k)	Sensitive to the choice of the similarity graph, Its parameters can greatly affect clustering results
Expectation Maximization	Model-based approach for clustering data	Allows users to specify the required number of clusters (k) as an input	Higher (k) increases computation time and resources considerably

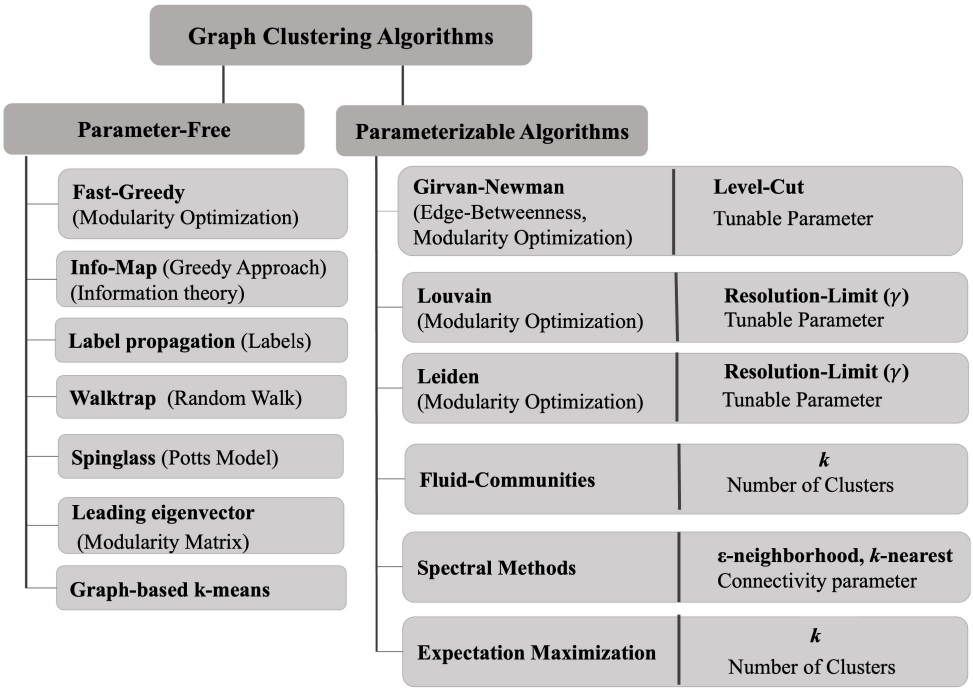

FIGURE 3. Graph clustering algorithm categorization summary.

categorized from a parametric perspective. This section also suggests areas in which the existing algorithms cannot be fulfilled. One such area is the ability to specify parameters that can guide clustering to ensure that the resulting clusters possess equal or similar densities. Understanding the underlying principle of each algorithm is advantageous because it enables the suitable selection of available algorithms. Although a precedent statement is agreed upon among the graph clustering community, it may not always occur in practice. With the public availability of several graph clustering software packages, many scientists from other fields are likely to select parameter-free and rely on default values to provide satisfactory clustering results (which cannot always be guaranteed). This phenomenon also occurs in the application of neural networks. Therefore, because graph clustering is an NP-hard problem, results should not always be taken on face value, and experimentation with several algorithms may be necessary. This Section also paves the way for the need for quality measures(s) for the clustering process, which is discussed in the next section. Table 1 provides a brief summary of feature, strength and limitation of each algorithm.It is, by no means, a comprehensive report with respect to all possible aspects.

IV. CLUSTERING QUALITY METRICS

Numerous clustering algorithms can be used to identify meaningful clusters. However, the discovered clusters may not be of the same quality even if they are meaningful [4]. Therefore, the evaluation of these clusters and the clustering process are necessary and play a significant role. For instance, an algorithm that results in vast diversity in size may not be useful for resource management [57]. It is important to recognize that determining the best cluster or set of clusters is subjective and depends entirely on the cluster definition and quality metrics employed in the evaluation. The evaluation of graph clustering algorithms requires appropriate quality metrics. Researchers have proposed various metrics to assess cluster quality, some of which consider cluster density both internally and externally within a graph [4], [25], [58], [59]. For example, modularity is a common metric for assessing the connections within clusters. These metrics quantitatively measure cluster connectedness from various perspectives including internal and external factors. A list of commonly used quality metrics that consider cluster connectivity can be found in [60]. In this section, we systematically categorize the quality metrics used in graph clustering with respect to the connectivity within the clusters in relation to overall connectivity.

A. INTERNAL CONNECTIVITY CLUSTERING QUALITY METRICS

This subsection elaborates on the clustering quality metrics, which are based on the cluster's internal degree, ($int_{deg}(C_i)$). The efficacy of clustering is determined by evaluating the characteristics of interconnections within clusters, with an emphasis on assessing cluster cohesion.

1) COVERAGE

Coverage (Cov) serves as a quality metric, as described in [9], for assessing the internal density of clusters within a graph [2], [4], [61]. It quantifies the ratio of internal edges across all clusters to the total number of edges in the graph. The coverage metric ranges from 0 to 1, with 0 indicating a graph in which each node is a singleton and disconnected cluster and 1 indicating a graph that is considered a single cluster. The capacity to achieve maximum coverage may be influenced by additional information, such as the number of clusters, and vice versa. Excessive optimization of the coverage metric may lead to a large cluster in which all nodes are assigned to the same cluster, thereby affecting the number of clusters [62]. The mathematical formula for calculating coverage is given by the following 4, where k is the number of clusters, and C represents a set of clusters in a graph:

$$Coverage\ (C) = \frac{\sum_{i=1}^{k} int_{deg}(C_i)}{e} \tag{4}$$

2) INTERNAL DENSITY

Graph density $\delta(G)$, quantifies the level of edge cohesion within a graph. The maximum number of edges in a graph can be calculated using the notion $n(n-1)/2$, where n is the number of nodes in the graph. Hence, the graph density can be expressed as the ratio of $2e$ to $n(n-1)$. A graph is considered complete if $\delta(G) = 1$. Graph density measures, including internal and external cluster densities, can be used to assess the clustering quality. The inter density (ID), denoted by $\delta_{int}(C_i)$ measures the cohesiveness of a cluster as a subgraph within a graph, as described in [63]. ID can be determined using 5 following equation, where C_n represents the number of nodes in a cluster:

$$\delta_{int}(C_i) = \frac{int_{deg}(C_i)}{C_n(C_n-1)/2} \tag{5}$$

Similarly, the external density (ED) of a cluster represented by $\delta_{ext}(C_i)$ is determined by calculating the ratio of edges that link the nodes within cluster C_i to those outside the cluster. The cluster external degree $\delta_{ext}(C_i)$ refers to the number of edges connecting cluster nodes (C_i) outside the cluster $|\{(v_1, v_2) \in E \mid v_1 \in C_i, v_2 \in V \setminus C_i\}|$, and can be determined using 6. where e denotes the total number of edges in the graph.

$$\delta_{ext}(C_i) = \frac{ext_{deg}(C_i)}{C_n(e-C_n)} \tag{6}$$

When considering an arbitrary cluster (C_i), $\delta_{int}(C_i)$ is expected to be significantly larger than $\delta(G)$, whereas $\delta_{ext}(C_i)$ is expected to be significantly smaller than $\delta(G)$. Specifically, a good cluster should satisfy the condition that $\delta_{ext}(C_i) < \delta(G) < \delta_{ext}(C_i)$. Whether explicitly stated, the primary objective of most clustering algorithms is to achieve an optimal balance between maximizing $\delta_{int}(C_i)$ (internal cluster density) and minimizing $\delta_{ext}(C_i)$ (external cluster density), as discussed in [25]. Equation 7 calculates

the Average Internal Edge Density (AIED) for a given number of clusters (k) within a graph.

$$\delta_{int}(G|C_1, C_2, \ldots, C_k) = \frac{1}{k}\sum_{i=1}^{k}(\delta_{int}(C_i)) \tag{7}$$

3) TRIANGLE PARTICIPATION RATIO (TPR)

The triangle participation ratio (TPR) is a quality metric that measures the internal connectivity of cluster nodes by considering the presence of triangular patterns. Cluster TPR denoted by TPR (C_i), is defined as the fraction of nodes within a cluster that participate in at least one triangle: $|\{v_1 : v_1 \in C_i, (v_2, v_3) : v_2, v_3 \in C_i, (v_1, v_2) \in E, (v_1, v_3) \in E, (v_2, v_3) \in E\} \neq \emptyset|$ over C_n [64]. TPR is well suited for community detection when the ground truth is known. A higher TPR value suggested a better community in terms of location prediction. Clique-based clustering, such as the Clique Augmentation Algorithm (CAA), yields a significantly higher TPR than other techniques [27]. The average TPR (ATPR) of the graph is determined by 8 below.

$$TPR(G|C_1, C_2, \ldots, C_k) = \frac{1}{k}\sum_{i=1}^{k} TPR(C_i) \tag{8}$$

4) FRACTION OVER MEDIAN DEGREE (FOMD)

The fraction of nodes with an over-median degree (FOMD) for a cluster is defined as the ratio of nodes within the cluster with an internal degree greater than the median degree ($d_m : |v_1 : v_1 \in C_i, |(v_1, v_2) : v_2 \in C_i| > d_m|$) to the total number of nodes in the cluster. A high FOMD number implies that a considerable fraction of the cluster nodes is densely interconnected, which is often indicative of a high-quality cluster. The average FOMD of a graph can be determined by 9 below.

$$FOMD(G|C_1, C_2, \ldots, C_k) = \frac{1}{k}\sum_{i=1}^{k} FOMD(C_i) \tag{9}$$

5) AVERAGE EMBEDDEDNESS

Embeddedness is a frequently used metric that provides an understanding of a cluster's tightness and connection to the overall graph. This is achieved by calculating the embeddedness of the individual nodes. The embeddedness of a node in a cluster is defined as the ratio of its internal degree $int\_deg(v)$ to the total degree $deg(v)$ within the graph. This metric measures the degree to which a node's immediate neighbors are members of its own cluster. A cluster with high embeddedness is considered to good cluster, which also indicates well-isolated clusters. Higher embeddedness values ware directly correlated with better detectability for all methods [65]. Equation 10 provides a means to determine the embeddedness of individual nodes within a cluster, denoted as emb(v, C_i). Meanwhile, 11 determines the average embeddedness (AE) of a cluster, and is denoted by $emb(C_i)$. Finally, 12 calculates the average embeddedness (A-Emb) of a set of clusters within the graph.

$$emb(v, C_i) = \frac{int_{deg}(v)}{deg(v)} \tag{10}$$

$$emb(C_i) = \frac{1}{C_n}\sum_{i=1}^{n}\frac{int_{deg}(v)}{deg(v)} \tag{11}$$

$$\text{emb}(G|C_1, C_2, \ldots, C_k) = \frac{1}{k}\sum_{i=1}^{k} emb(v, C_i) \tag{12}$$

B. EXTERNAL CONNECTIVITY CLUSTERING QUALITY METRICS

The external connectivity evaluation of clustering provides a method for determining how well a clustering method can separate data points into distinct clusters. Effective clustering is characterized by a minimal number of edges connecting clusters, signifying successful data point separation and high quality clustering. To assess this aspect, numerous metrics have been proposed to capture the notion of inter-cluster sparsity. These metrics rely on the edges connecting a cluster to other clusters or to different portions of the graph.

1) CUT RATIO

The cut ratio (CR) metric originated in the domain of circuit partitioning [66], and it is well-known that optimizing this metric is a recognized NP-hard problem. Nevertheless, spectral clustering has demonstrated effectiveness in achieving a significant complexity reduction in cut ratio [67]. The cut ratio of a cluster, denoted as $CR(C_i)$, was calculated using the 13. This equation measures the fraction of edges connecting nodes in different clusters, relative to the total number of edges in the graph. The average cut ratio (A-CR) for the clusters in the graph is computed 14 as below.

$$CR(C_i) = \frac{int_{deg}(C_i)}{C_n(n - C_n)} \tag{13}$$

$$CR(G|C_1, C_2, \ldots, C_k) = \frac{1}{k}\sum_{i=1}^{k} CR(C_i) \tag{14}$$

2) EXPANSION

Expansion (Exp) serves as a metric for evaluating the performance of the clustering algorithms. It quantifies the number of external edges of the nodes within a cluster. Expansion is calculated as the average number of edges per node that extends outside the cluster. A low expansion value suggests that the nodes within the cluster are closely connected and deeply integrated within the cluster, indicating a high level of compactness.

Cluster expansion was calculated using the 15. The average expansion (A-Exp) can be determined by using 16, which considers the expansion of a set of clusters.

$$Exp(C_i) = \frac{ext_{deg}(C_i)}{C_n} \tag{15}$$

$$Exp(G|C_1, C_2, \ldots, C_k) = \frac{1}{k}\sum_{i=1}^{k} Exp(C_i) \tag{16}$$

C. INTEGRATED QUALITY METRICS

Clustering quality metrics that consider both internal and external connectivity are commonly employed to evaluate the performance of clustering algorithms. These measures typically examine the degree to which nodes within a cluster are interconnected (internal connectivity), and consider their connections with nodes outside the cluster (external connectivity). In the field of graph clustering evaluation, various metrics that encompass both internal and external connectivity have emerged as widely adopted benchmarks. Below are some common examples of these types of metrics accompanied by a concise overview of widely used measures.

1) CONDUCTANCE

Conductance (Con) is a widely adopted metric for evaluating clustering quality and is, known for its simplicity and effectiveness. It comprehensively accounts for both internal and external connectivity within a cluster with various definitions available in the literature.

The conductance of a cluster is generally computed by considering two cluster parameters: the external degree ($ext_{deg}(C_i)$) and the total degree ($tot_{deg}(C_i)$). These parameters can be defined as either the number of edges ($|E(C_i)|$) or the sum of node degrees within the cluster ($\sum_{v \in C_i} deg(v)$), with the selection depending on the option that yields a lower value. The resulting metric falls within the range of 0 to 1, where 0 is the least favorable value and 1 is the most favorable. The total degree can be measured either by the number of edges $|E(C_i)|$ or by summing the degrees of the nodes within the cluster $\sum_{v \in C_i} deg(v)$.

Three common mathematical formulations for the conductance of a cluster, denoted by $\phi(C_i)$, are presented in 17,18,19 as described in [61] and [62]. To assess the overall quality of the clustering process, the average conductance (A-Cond) across all clusters is frequently employed and can be calculated using the formula referenced in 20.

$$\phi(C_i) = \frac{ext_{deg}(C_i)}{min(tot_{deg}(C_i), tot_{deg}(V \backslash C_i))} \tag{17}$$

$$\phi(C_i) = \frac{ext_{deg}(C_i)}{min(tot_{deg}(C_i), 2e - tot_{deg}(C_i))} \tag{18}$$

$$\phi(C_i) = \frac{ext_{deg}(C_i)}{2(tot_{deg}(C_i))} \tag{19}$$

$$\phi(G|C_1, C_2..C_k) = \frac{1}{k}\sum_{i=1}^{k} \phi(C_i) \tag{20}$$

2) NORMALIZED-CUT

Identifying clusters with few external connections is akin to addressing the maximum-cut minimization problem, which is known for its high computational complexity. However, the minimum-cut approach often results in numerous very small clusters, providing limited insight into the underlying structure of the graph [68]. A graph-theoretic criterion called normalized-cut (NCut) was developed to address this problem and determine the cluster quality. NCut partitions a graph into subgraphs (or clusters) with well connected nodes within each subgraph and weakly connected nodes between subgraphs. NCut is a normalized variant of the cut metric that quantifies the edges between the subgraphs. This metric considers two parameters: the count of external edges and the total number of edges within the graph.

The mathematical representation of the normalized cut for a cluster, denoted as $NCut(C_i)$, is defined mathematically defined in 21. The average value (A-NCut) can be calculated using 22 for a set of clusters.

$$NCut(C_i) = \frac{ext_{deg}(C_i)}{2(tot_{deg}(C_i))} + \frac{ext_{deg}(C_i)}{2(e - int_{deg}(C_i)) + ext_{deg}(C_i)} \tag{21}$$

$$NCut(G|C_1, C_2, \ldots, C_k) = \frac{1}{k}\sum_{i=1}^{k} NCut(C_i) \tag{22}$$

In practice, achieving a value of $NCut(C_i) = 0$ in a connected graph is not possible because there must be at least one inter-cluster edge. Therefore, the goal of clustering is to obtain a solution with a low value of $NCut(C_i)$, where a smaller value indicates a better quality of the clustering solution.

3) FLAKE-ODF

Out degree fraction (ODF) quantifies a cluster's external connectivity by assessing the percentage of nodes within the cluster that exhibit a greater number of edges connected to nodes outside the cluster than to nodes within the cluster, as discussed in [30]. A lower F-ODF value indicates a higher level of cohesion within the cluster, signifying that the nodes within the cluster exhibit stronger connections with each other compared to nodes outside the cluster. To calculate the F-ODF for an individual cluster, we applied 23. In addition, in the case of a graph containing k clusters, the average (A-FODF) of the F-ODF can be computed by employing 24.

$$F\text{-}ODF(C_i) = \frac{|v : v \in C_i, |(v, v_1) \in E : v \in C_i| < \frac{d(v)}{2}|}{C_n} \tag{23}$$

$$F\text{-}ODF(G|C_{(1,)}C_2..C_k) = \frac{1}{k}\sum_{i=1}^{k} F\text{-}ODF(C_i) \tag{24}$$

The assessment of clustering quality emphasizes a significant concern; each metric is focused on limited aspects of clustering outcomes, such as internal or external connectivity, or a combination of both. Consequently, understanding the aspects of clustering would allow appropriate metric(s) to be employed. It must be borne in mind that it may not be possible to obtain clustering results that satisfy all the quality metrics. Among these metrics, conductance and coverage are popular and tend to be selected as quality metrics(s). Nevertheless, using these generic purpose metrics to result in suitably

high quality, but they can be inappropriate with respect to the objective(s) of the clustering or when the objectives(s) is(are) not well understood. The opposite can also be true, for example using 'coverage' to assess the clustering may result in a relatively low value if the objective is to identify dense areas in networks where isolated edges are not of concern. This is particularly true in this study as the existing quality metrics are deemed inappropriate. This aspect is discussed in detail in Section VII.

V. PARTITIONING CLUSTERING WITH USER-SPECIFIED RELATIVE DENSITY

As stated earlier, this study presents the first attempt to partition the clustering of a connected, undirected, and unweighted graph that allows users to specify their desired cluster density. The density of a cluster in this study is taken from the widely recognized concept of ''Relative Density'', which is further elaborated in the subsequent section. It is assumed that the clusters are not apparent or visible at first glance. In such cases, clustering of these type of graphs belong to density based clustering such as ''DBSCAN'' algorithms. Additionally, the requirement for a cluster to have a certain density may not be meaningful. Suitable graphs (networks) for the application of this study are discussed in Section I and some illustrations are presented in Section VI.

The methodology in this study also employs fundamental principles from graph theory, including the utilization of Spanning Trees, to effectively decompose a connected graph. These concepts allow the identification of cliques of size three within the main graph. Subsequently, the algorithm expands from a clique while considering neighboring cliques to determine the desired level of density. To provide a comprehensive of the research methodology being proposed, this section is organized as follows: first, it provides an explanation of the terminology and symbols employed in this study, followed by an in-depth examination of the primary algorithms that have been put into practice. In addition, it includes a systematic demonstration of the algorithmic approach, along with an instance featuring a randomly connected network.

A. DEFINITIONS AND NOTATIONS

To aid in the comprehension of the proposed partitioning clustering approach, the following notation is used:

1) SPANNING TREE

A spanning tree is often used as a starting point for analyzing the sub-components of a network. Consider a graph $G = (V, E)$, and let $T(V, E')$ be a spanning tree of G, where $E' \subseteq E$. Notably, $T(V, E')$ is acyclic and has the same number of nodes as G. Fig. 4(a) depicts an arbitrary simple graph, whereas Fig. 4(b) illustrates a possible spanning tree of the graph. It is important to note that several spanning trees mare possible for a connected graph.

FIGURE 4. (a) Arbitrary graph (b) Possible spanning tree.

2) FUNDAMENTAL CYCLE

The addition of an edge from the graph $G = (V, E)$ to the spanning tree $T(V, E')$ may or may not result in a cycle, known as a fundamental cycle. The formation of a cycle depends solely on the graph topology. The procedure for obtaining a fundamental cycle associated with a spanning tree $T(V, E')$ is illustrated in Figs. 5 (a)_(c).

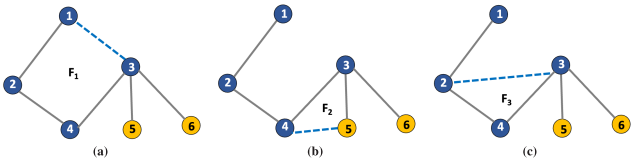

FIGURE 5. Three fundamental cycles: (a) F_1 = (1,2,3,4), (b) F_2 = (3,4,5), and (c) F_3 = (2,3,4) from the Spanning Tree in Fig. 4 (b).

3) CLIQUE DEGREE

The computation of a clique degree, denoted as K_{deg}, involves the summation of node degrees within a clique (K), with the exclusion of duplicate edges that might lead to double counting. The clique degree is determined by using 25:

$$K_{deg} = \sum_{i=1}^{3} deg(v_i) - 3 \tag{25}$$

4) AVERAGE DEGREE CLIQUE

The formation and density of partitioned clusters may be influenced by several factors, among which the degree of cliques plays a significant role. This paper proposes the use of cliques with an average degree as an initial reference point for both cluster initialization and expansion clustering. The following mathematical expression represents the identification of cliques with an average degree.

$$A_{deg}(K) = \frac{1}{|G(K)|} \sum_{K_i}^{K_n} K_{deg} \tag{26}$$

where $|G(K)| = (K_i \ldots K_n)$.

Intuitively, a clique with an exact degree of $A_{deg}(K)$ may not be present. In such instances, the algorithm selects the clique that has the closest value to $A_{deg}(K)$ to ensure the optimal initialization of the clustering process. This approach is further discussed in subsection II, in which the identification of cliques from a spanning tree is elaborated.

5) RELATIVE DENSITY

The notion of relative density has been extensively studied from two perspectives: identifying dense clusters and assessing the effectiveness of graph clustering methods, as discussed in [69]. The first notable application of relative density is the identification of clusters with various goals, as referenced in [2] and [25]. Nonetheless, in the study presented by [70], it was proposed that density and cut-size based metrics are appropriate choices primarily for networks characterized by densely connected nodes or cliques. Their performance is less optimal when dealing with star-like structures or graphs that exhibit a sparsity. The second application involves employing density estimates derived from minimum spanning trees, which aids in the detection of clusters within dense regions in multi-scale datasets [71]. The algorithm outlined in [72], known as Identifying Diverse Density Clusters (IDDC), is a technique based on the relative density. It identifies clusters of different densities and effectively manages outliers. Additional studies that utilized the relative density also include [73] and [74]. However, it should be noted that relative density favors introverted clusters, which are subgraphs with minimal connections to the rest of the graph. This characteristic may limit the effectiveness of relative density as a fitness measure for community-detection algorithms in highly complex networks. The relative density is adopted as a metric for the user-defined density specification, where the relative density of a cluster (C_i), denoted as $\delta_r(C_i)$, is determined as the ratio of its internal edges to its total degree or edges (cluster total degree), as described by 27 below.

$$\delta_r(C_i) = \frac{\sum_{v \in C_i} int_{deg}(v)}{tot_{deg}(C_i)} \tag{27}$$

where $tot_{deg}(C_i)$ is the total degree of cluster C_i referred to (3). This total degree can alternatively be expressed as:

$$tot_{deg}(C_i) = \sum_{v \in C_i} int_{deg}(v) + \sum_{v \in C_i} ext_{deg}(v) \tag{28}$$

Relative density parameter ranged from 0 to 1. An entire graph with no intrinsic substructure has a relative density of 1, whereas a single node as a stand-alone node has a relative density of 0.

6) USER-SPECIFIED RELATIVE DENSITY

Referring to 27, the user-specified input parameter, denoted as $U(\delta_r)$ is employed through the concept of relative density in this study. This indicates that the equation considers a user_Specified parameter to assess the relative density of a given cluster within the context of the entire network.

B. PROPOSED APPROACH TO PARTITIONING CLUSTERING

The proposed clustering approach involved a two-phase strategy. In the first phase, all cliques in a graph are identified, and in the second phase, the graph is partitioned based on the user-specified density parameter. The processes for identifying cliques and partitioning clusters are described in the subsequent section.

1) DETERMINATION OF CLIQUES IN A GRAPH

To initiate the clustering process, the first step is to determine all cliques in the graph. The simplest way to determine all cliques (i.e., triangles) within a graph is to determine all possible cliques connected to each node and have a list of cliques encountered so far. This process is iterative, and several cliques may be unnecessary determined more than once as each node is processed (hence the need for the encountered cliques). Determination of all cliques in this work takes the advantage from the method to decompose all substructures (cycles) hidden in the composite graph. While not all substructures are cliques, it still a more efficient method in general. Also note that, even if there exists a way to calculate number of cliques within a graph, this information is not sufficient as position of each clique within the graph must be known to facilitate the clustering. This involves constructing a spanning tree and identifying the fundamental cycles, followed by finding all remaining cycles in the graph. The desired output of this phase is cliques (triangles), consisting of three edges that form a triangular structure. Algorithm 1 presents step-by-step for identifying cliques and their respective degrees. Following the identification of all cliques in the initial phase, they were utilized in the second phase of the partitioning clustering process.

2) DETERMINATION PARTITIONING CLUSTERS

After identifying all cliques in the first phase of the clustering process, the partitioning clustering process commences with the following steps: As it is important to note that, this work adopts an expansion approach from a single clique, where the relative density δ_r of the cluster is continuously monitored until reaching the desired value of $U(\delta_r)$, or the closest value to it. Once a cluster has been determined, the process is repeated to form new clusters using the remaining cliques and edges. This iterative process continues until all the cliques and edges are considered.

In the second phase of the clustering process, the initial consideration is to determine the most appropriate clique for starting the clustering process. Four potential scenarios were examined to select a clique as the starting point: the lowest total degree, average total degree, highest total degree, and random choice of the clique. However, extensive experimentation has revealed that prioritizing cliques with the highest or lowest total degree can lead to uneven cluster sizes despite having almost the same relative density δ_r. This observation merits further investigation, particularly in cases in which clusters of varying sizes with identical $U(\delta_r)$ values are desired. For example, situations in which the order of clusters from the highest number of nodes with lower densities to the lowest number of nodes with higher densities is of interest. However, such investigations were beyond the scope of this study. Therefore, to initialize the clustering process and cluster expansion, a clique with an

Algorithm 1 Pseudo Code for Cliques Identification

Require: Undirected and Unweighted Connected Graph.
Ensure: List of Cliques (Triangles) with respective degrees.

```
Finding a spanning Tree from the graph;
Number of Fundamental cycles (F_n) = 0;
Number of Cliques (C_n) = 0;
repeat
    Add an edge that is in the graph but not in the Spanning Tree
    if cycle occurs (i.e., Fundamental Cycle is found) then
        Record this as a Fundamental Cycle;
        Increment the value of (F_n) by 1;
        if this cycle is a new desired clique (i.e., a triangle) then
            Record this cycle as a Clique;
            Determine the total degree of this clique;
            Increment the value of (C_n) by 1;
        end if
    end if
until (no more edges in a graph to consider)
Number of combinations (C) = 2;
repeat
    Perform "Exclusive OR" to all Fundamental Cycles of combination (C) (to determine remaining cliques);
    if (The "Exclusive OR" operation results in a Clique) then
        Record this Clique;
        Determine the total degree of this Clique;
        Increment the value of (C_n) by 1;
    end if
    Increment the value of (C) by 1;
until (C = F_n)
List the Cliques in accordance with their respective degrees (i.e., all the possible cliques are determined);
```

average (or closest to average) total degree was selected as the initial cluster seed. If multiple cliques exist, one is randomly selected, although there are typically few cliques from which to choose. In addition to selecting cliques with average total degrees for the clustering process initialization, prioritizing cliques for cluster expansion is critical. The random selection of adjacent cliques can lead to linear-shaped clusters, which are undesirable. Therefore, the selection of the next clique or edge during expansion is based on the degree of nodes shared with the existing cluster, which is set as follows:

1) Priority_1: Cliques with all three nodes in common with the existing cluster have the highest priority.
2) Priority_2: Cliques with two nodes in common with the existing cluster have the second-highest priority.
3) Priority_3: Cliques with one node in common with the existing cluster have the third-highest priority.
4) Priority_4: Edges adjacent to the existing cluster are considered based on the average total degree of nodes that do not belong to the existing cluster if no adjacent cliques exist.

By following this prioritization strategy, cluster expansion is carried out effectively, while avoiding the formation of linear clusters. Extensive experimentation has demonstrated that this approach leads to tightly knitted and highly embedded clusters of nearly identical size. However, it may not always be possible to identify a cluster with the exact user_specified density ($U(\delta_r)$) during the expansion process. In such a situation, the strategy is to accept an existing cluster with a relative density close to $U(\delta_r)$. Specifically, the most recent addition is accepted if it brings the relative density of the existing cluster, $\delta_r(C_i)$ closer to $U(\delta_r)$. Algorithm 2 illustrates the determination of partitioning clusters in the second phase.

As partitioning clustering in this study assumes that the network is static, it is assumed that the computation time is not a constraint. This assumption may not be possible if the network is dynamic; for example, communication among moving vehicles, where it is mobile ad_hoc in nature. A basic analysis suggests that the number of cliques ($|G(K)|$) is likely greater than the number of fundamental cycles (F_n). In practice, the availability of open-source software, such as NetworkX, makes it possible to determine the substructures. In this study, this implies the determination of all the triangles (i.e., cliques). The worst case in partitioning clustering in this study is when the network is fully connected (as this possesses the largest number of cliques) and has a density of 1 (i.e., the entire graph is a large cluster). The computation time is then dominated by the expansion of a one larger (whole graph) cluster. This has a time complexity of $O(|G(K)|)$, where $|G(K)| = K_n$ denotes the number of cliques in the network.

3) AN ILLUSTRATION ON A RANDOM GRAPH

To illustrate the partitioning clustering process proposed in this study, a simple random graph featuring 12 vertices ($v_0, v_1, \ldots, v_{11}$) and 19 edges was generated, as depicted in Fig. 6(a). In this demonstration, it was presumed that all cliques were identified, and the target $U(\delta_r)$ was set to 0.45.

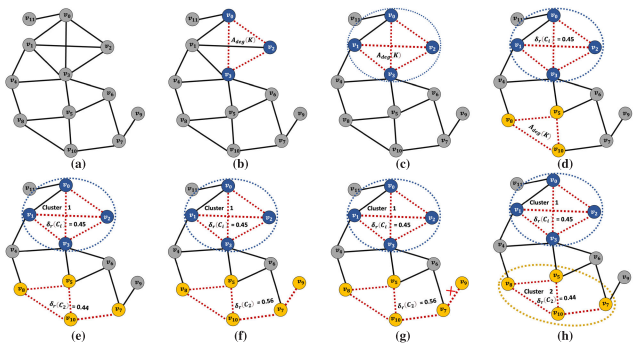

FIGURE 6. An illustration of partitioning graph clustering on a simple arbitrary graph.

Algorithm 2 Graph Partitioning Clustering Algorithm

Require: List of cliques with their respective degrees, Undirected and Unweighted Connected Graph
Ensure: Collection of partitioning clusters with equal or close to $U(\delta_r)$

Set Collection of Partitioning clusters to empty
while not all cliques and edges are considered **do**
  A new cluster is set to an average degree clique
  Calculate δ_r of the cluster: $\delta_r(C_i)$
  while there exist *clique(s)* adjacent to the cluster **do**
    Find all the adjacent *clique(s)* with an average degree to the cluster
    Consider the cliques with common nodes in descending order to the cluster
    for each clique **do**
      temp_cluster ← (cluster with the addition of the clique)
      Calculate δ_r of temp_cluster: $\delta_r(\text{temp}_{C_i})$
      if $\delta_r(\text{temp}_{C_i}) = U(\delta_r)$ **then**
        cluster ← temp_cluster
        Mark this clique as considered
        // an ideal cluster is found
        goto 99
      else if $\delta_r(\text{temp}_{C_i}) < U(\delta_r)$ **then**
        cluster ← temp_cluster - (a cluster is expanded)
        Mark this clique as considered
      else if $\delta_r(\text{temp}_{C_i}) > U(\delta_r)$ **then**
        $\text{Diff\_temp}_{C_i}\_U(\delta_r) \leftarrow |\delta_r(\text{temp}_{C_i}) - U(\delta_r)|$
        $\text{Diff\_}C_i\_U(\delta_r) \leftarrow |\delta_r(C_i) - U(\delta_r)|$
        if $\text{Diff\_temp}_{C_i}\_U(\delta_r) < \text{Diff\_}C_i\_U(\delta_r)$ **then**
          cluster ← temp_cluster – (a cluster is expanded); – Higher but closer to $U(\delta_r)$
          Mark this clique or edge as considered
          goto 99 (Proceed to the next cluster)
        else
          $\text{Diff\_temp}_{C_i}\_U(\delta_r) \geq \text{Diff\_}C_i\_U(\delta_r)$
          // This clique or edge is rejected;
          cluster ← cluster without the addition of this clique or edge – Lower but closer to $U(\delta_r)$
          goto 99 (Proceed to the next cluster)
        end if
      end if
    end for
  end while
  while there exist edge(s) adjacent to cluster **do**
    Carry out a similar expansion process of all clusters with cliques until no more edges to consider
  end while
  99 – A partitioning cluster with δ_r closest to $U(\delta_r)$ is now determined
  Add cluster to the Collection of partitioning Clusters
end while

The clustering process begins by selecting a clique with an average degree (v_0, v_1, v_2) as the initial cluster, as illustrated in Fig. 6(b). The cluster then expanded with the inclusion of the most suitable adjacent cliques (v_1, v_2, v_3), as shown in Fig. 6(c). As the resulting cluster $(C_1) = (v_0, v_1, v_2, v_3)$ had a relative density $\delta_r(C_1)$ of exactly 0.45, it was determined to be a cluster.

Next, a new cluster is initiated by selecting a clique with average degrees (v_5, v_8, v_{10}) from the remaining cliques, as shown in Fig. 6(d). Once all the available cliques have been considered for inclusion in this cluster, the clustering process considers adjacent edges based on their average degree in relation to the cluster for potential inclusion (i.e., edges (v_7, v_9) in Figs. 6(e) and (f)). Note that edges (v_7, v_9) are considered for inclusion but are rejected because this makes $U(\delta_r)$ 0.56. Therefore, the partitioning clustering process resulted in two clusters with relative densities of 0.45 and 0.44, respectively, as illustrated in Fig. 6(h).

VI. PARTITIONING CLUSTERING RESULTS

From the perspective of using networks to validate the proposed algorithms, there are two main approaches: using real networks available from various public websites, and generating synthetic networks with the required characteristics. Both approaches were used in this work. However, in this section, real-world networks are presented in the evaluation and synthetic results are omitted. In addition to the fact that real-world networks exist, they also allow comparisons for future work from the same and different perspectives. These network datasets were obtained in Graph Modeling Language (GML) format from the Network Repository (https://networkrepository.com) and transformed into graph structures using NetworkX 3.0. These were selected to reflect the variety of structures, average degrees, and other characteristics necessary for ensuring the validity of the proposed algorithm. Table 2 summarizes their statistical characteristics. The minimum degree (Min_{deg}) identifies the least connected node. The maximum degree (Max_{deg}) is the most highly connected, the average degree (Avg_{deg}) of a graph is the mean number of edges connected to each node within a graph, the standard deviation (σ) quantifies variability in connectivity relative to the mean.

A. EXPERIMENTAL SETTING

According to Section II, the user-specified density $U(\delta_r)$ can be set between 0 and 1. To validate the proposed method, $U(\delta_r)$ values of 0.4 and 0.6 are selected for demonstration as they are not too sparse or too dense (i.e., one below and another above 0.5). The clustering results obtained from the eight real-world networks are summarized in Tables 3 and 4. To visually demonstrate the resultant clusters generated by the proposed approach within a reasonable space, three networks, namely Zachary's Karate Club (ZKC),

TABLE 2. Key characteristics of eight selected real-world networks.

No	Dataset	$\delta(G)$	V	E	Min_{deg}	Max_{deg}	Avg_{deg}	(σ)	$\|G(K)\|$
1	**Zachary's Karate Club (ZKC)**	0.139	34	78	1	17	4.5	3.8	45
2	**Aves-Weaver Social (AWS)**	0.176	42	152	2	27	7.2	4.9	287
3	**Dolphins-interaction (DI)**	0.084	62	159	1	12	5.1	2.9	95
4	**Les Misérables (LM)**	0.086	77	254	1	36	6.6	6.0	467
5	**Political Books (PB)**	0.080	105	441	2	25	8.4	5.4	560
6	**American College Football (ACF)**	0.093	115	613	7	12	10.8	0.8	810
7	**Facebook-pages-Food (FPB)**	0.010	620	2102	1	134	6.8	9.4	2935
8	**US-Grid Power (USGP)**	0.005	4941	6594	1	19	2.7	1.7	651

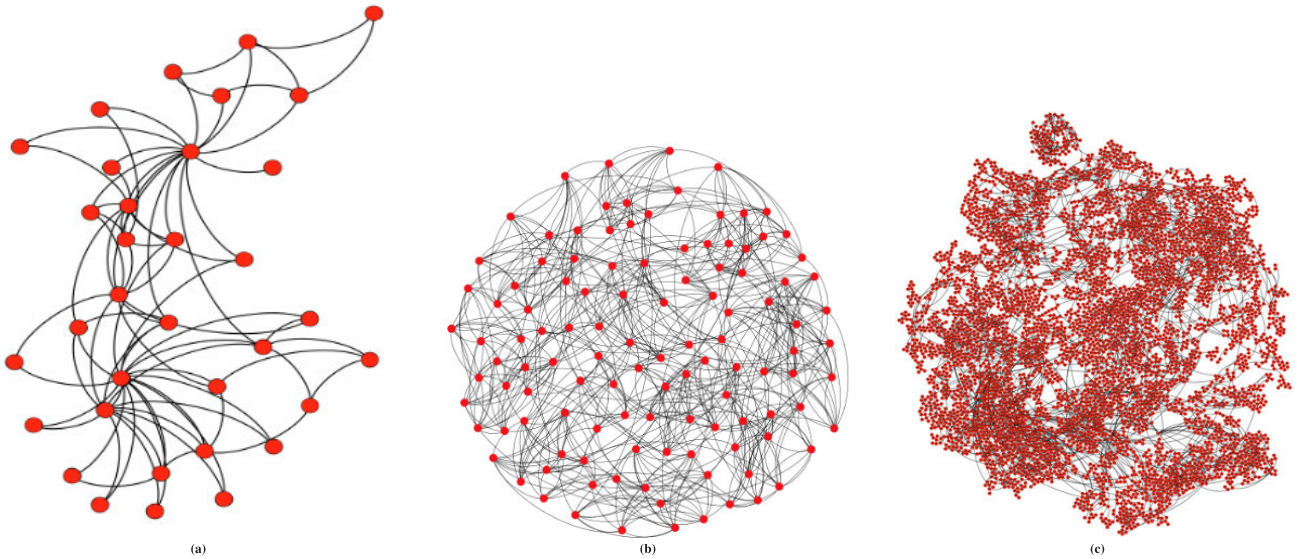

FIGURE 7. The real structure of the selected networks (a) Zachary's karate club (b) American college football and (c) US grid power.

American College Football Network (ACF), and US-Grid Power Network (USGP), were selected. These networks were chosen based on their diverse properties and structures (e.g., number of nodes, number of edges, Avg_{deg} etc). Figs. 7(a)-(c) illustrate the actual connectivity structures of the selected networks.

1) PARTITIONING CLUSTERING RESULTS WITH USER-SPECIFIED DENSITY OF 0.4

Table 3 presents the key findings of the partitioning clustering outcomes obtained from the eight real-world graphs with a user_specified density parameter $U(\delta_r)$ of 0.4. The average relative density (denoted as $Avg_{\delta_r}(C)$ is shown for each dataset, rather than showing δ_r of each cluster for brevity.

TABLE 3. Partitioning clustering results with a $U(\delta_r)$ of 0.4.

No.	Dataset	(C_k)	$Avg_{\delta_r}(C)$	unclustered Nodes
1	**ZKC**	4	0.40	9
2	**AWS**	5	0.39	8
3	**DI**	5	0.40	19
4	**LM**	3	0.40	25
5	**PB**	7	0.35	14
6	**ACF**	12	0.37	13
7	**FBPF**	20	0.39	209
8	**USGP**	184	0.40	2880

The findings in Table 3 reveal that, across all networks, $Avg_{\delta_r}(C)$ is equal to or nearly identical to the user-specified relative density for a given $U(\delta_r)$ of 0.4. However, the number of clusters (C_k) varies among datasets owing to various factors, such as the network size and structure and $U(\delta_r)$ among others. Fig. 8 graphically depicts the clustering outcomes of the three selected networks. Clusters are color-coded, whereas unclustered nodes are shown in grey, following a consistent color scheme used throughout the study.

Referring to the results in Table 3 and Fig. 8, the number of unclustered nodes varied depending on the structural characteristics of the network. Notbaly, the number of unclustered nodes in the US-Grid Power results was relatively high (2, 880 nodes out of all 4, 941 nodes). This is because the average degree of the nodes is considerably small, ($Avg_{(deg)}$ = 2.7), resulting in a relatively low number of cliques. The density used in this study is the 'relative density'; hence, the total number of external degrees of a clique (or any cluster considered) may be relatively small or close to the total number of internal degrees. Another factor is that this work focuses on partitioning clustering. If overlapping clustering is the objective, it is anticipated that the number of unclustered nodes in cases such as the US-Grid Power network will

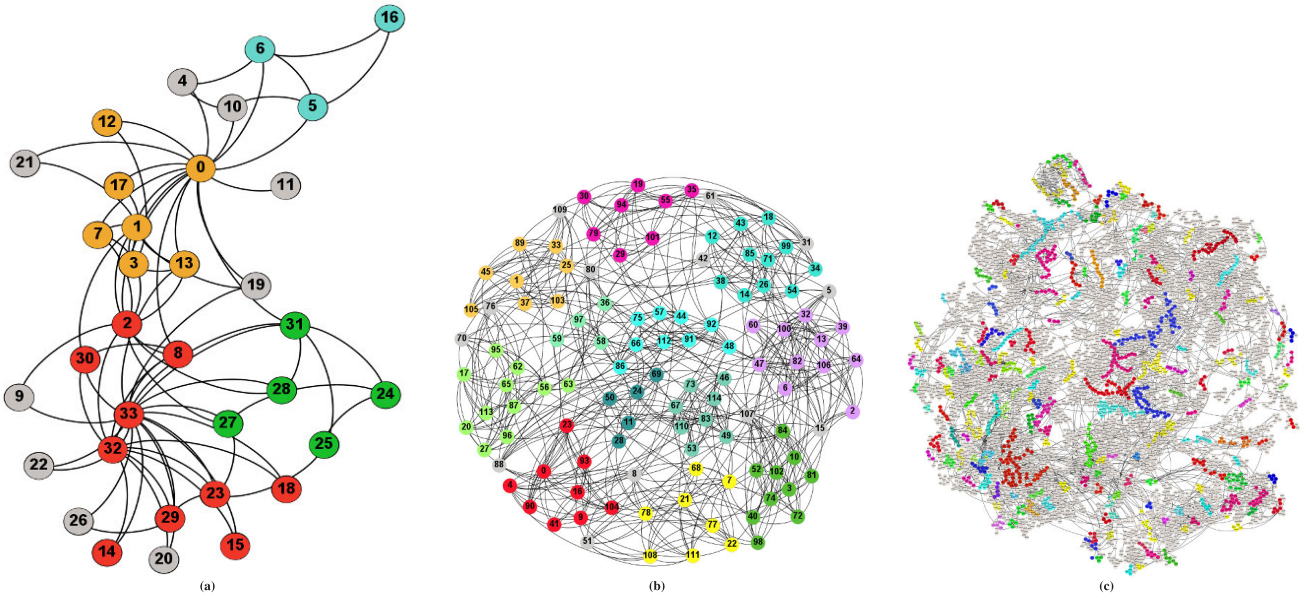

FIGURE 8. Graphically representation of partitioning clustering results of three selected networks at $U(\delta_r)$ of 0.4. (a) Zachary's Karate Network, (b) American college football network, (c) US-grid power network.

be much less. Partitioning clustering of Facebook-Pages-Food also resulted in relatively high unclustered nodes (approximately 10% of the total edges). This pattern is likely due to the relatively high number of total cliques ($|G(K)|$), in conjunction with the relatively low $Avg_{(deg)}$, when compared to the total number of edges in the graph. The Zachary Karate Club network results represents an example of a relatively small network with good spread results. The results for American college football also spread relatively well because of the well-balanced ratio between the total number of nodes (115) and edges (613), as well as the fact that $Avg_{(deg)}$ is relatively high (10.8).

2) PARTITIONING CLUSTERING RESULTS WITH USER-SPECIFIED DENSITY OF 0.6

Table 4 presents the key findings of the partitioning clustering outcomes obtained from all eight real-world networks with a user-specified relative density $U(\delta_r)$ value of 0.6. Again, $Avg_{\delta_r}(C)$ was used to denote the average relative density of the set of resulting clusters for each dataset.

TABLE 4. Partitioning clustering results with a $U(\delta_r)$ of 0.6.

No.	Dataset	(C_k)	$Avg_{\delta_r}(C)$	unclustered Nodes
1	**ZKC**	3	0.60	5
2	**AWS**	5	0.51	5
3	**DI**	3	0.55	17
4	**LM**	3	0.60	20
5	**PB**	6	0.45	15
6	**ACFA**	4	0.57	2
7	**FBPF**	16	0.58	236
8	**USGP**	84	0.59	1478

Overall, the results for $U(\delta_r)$ of 0.6 exhibit consistency with those of 0.4 (i.e., a smaller number of clusters and a smaller number of unclustered nodes as $U(\delta_r)$ is higher). This trend is illustrated in Fig. 9 for the three selected datasets. The Political Books network, however, yielded less favorable results for both $U(\delta_r)$ values of 0.4 and 0.6, with $Avg_{(\delta_r)}(C)$ values of 0.35 and 0.45, respectively, compared to the ideal values of 0.4 and 0.6. The observed results are most likely influenced by the significant variation ($\sigma = 5.4$) in the degree of nodes within the graph, with its lowest degree at 2 and highest at 25, a relatively high $Avg_{(deg)}$ of 8.4 and a high number of $|G(K)|$.

In comparison with Fig. 8 which depict $U(\delta_r)$ of 0.4, the results for both Zachary's Karate Club and the American College Football networks align with the expectations. However, the results for the US-Grid Power network display a noticeable variation in the number and size of clusters. These variations can be attributed to the distinctive characteristics of the US-Grid Power network, including the ratio of total nodes to total edges and the relatively low Avg_{deg}, in conjunction with the chosen value of $U(\delta_r)$.

The results presented in Tables 2 (for $U(\delta_r)$ of 0.4) and 3 (for $U(\delta_r)$ of 0.6) illustrate that the proposed partitioning method in this work performs reasonably well, effectively creating clusters that closely match the user-specified density $U(\delta_r)$. It is worth noting that the key determinant of these outcomes lies in the network's inherent characteristics, as discussed in Section VIII, which provides a more detailed analysis of these findings.

B. PERFORMANCE ANALYSIS USING PRE-EXISTING METRICS

Similar to other processes, finding an appropriate method (metric) to evaluate an algorithm is essential, and the proposed clustering algorithm is no exception to this.

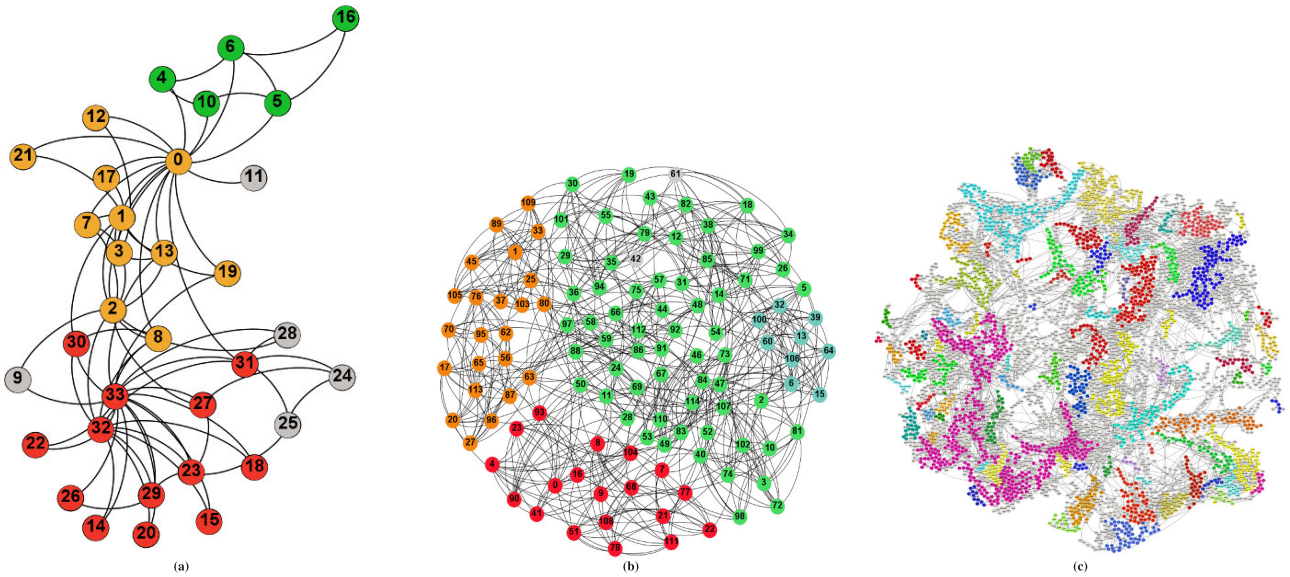

FIGURE 9. Graphically representation of partitioning clustering results of three selected networks at $U(\delta_r)$ of 0.6. (a) Zachary's karate network, (b) American college football network, (c) US-grid power network.

However, identifying a suitable metric is an indispensable yet challenging task. Section VIII elaborates on several widely recognized quality metrics that are frequently used to evaluate the efficacy of graph clustering techniques. However, the existing quality metrics are limited in their ability to evaluate clustering algorithms with different objectives. Therefore, they should be applied according to their objectives. This point is well_stated in [75]. To demonstrate the relevance of having an appropriate metric to evaluate the proposed graph clustering, the existing metrics were applied to the experiments. Tables 5 and 6 list the values of their application to the proposed partitioning clustering of all eight networks with respect to the internal and external connectivity, respectively.

TABLE 5. Resultant clusters with a $U(\delta_r)$ of 0.4 internal connectivity evaluation.

No.	Dataset	Cov	AIED	ATPR	FOMD	AE
1	ZKC	0.46	0.60	0.91	0.30	0.70
2	AWS	0.51	0.70	0.95	0.20	0.60
3	DI	0.52	0.60	0.90	0.30	0.60
4	LM	0.52	0.50	0.90	0.29	0.60
5	PB	0.54	0.40	0.90	0.40	0.58
6	ACFA	0.54	0.80	1.00	0.17	0.60
7	FBPF	0.51	0.40	0.80	0.35	0.70
8	USGP	0.21	0.60	0.80	0.30	0.65

Note that the values of the internal connectivity metrics their values range from 0 to 1 (1 is the best value). Among the existing metrics, the average TPR (Average Triangle Participation Ratio) yields the best value. The TPR metric favors the clique shape adopted in this study, but it may not necessarily hold true for all clustering methods.

Similarly, the external connectivity metrics also ranged from 0 to 1 (with 0 being the best value). Among the existing metrics, cut_ratio yielded the best value. This is because of external degrees of each cluster, together with the other

TABLE 6. Resultant clusters with a $U(\delta_r)$ of 0.6 internal connectivity evaluation.

No.	Dataset	Cov	AIED	ATPR	FOMD	AE
1	ZKC	0.69	0.40	0.90	0.40	0.80
2	AWS	0.70	0.70	0.95	0.40	0.70
3	DI	0.63	0.50	1.00	0.30	0.70
4	LM	0.69	0.50	0.80	0.30	0.80
5	PB	0.65	0.40	0.90	0.29	0.60
6	ACF	0.80	0.50	1.00	0.33	0.74
7	FBPF	0.64	0.53	0.80	0.31	0.78
8	USGP	0.49	0.30	0.50	0.36	0.79

TABLE 7. Resultant clusters with a $U(\delta_r)$ of 0.4 external connectivity evaluation.

No.	Dataset	A-CR	A-Cond	A-Ncut	A-FODF
1	ZKC	0.08	0.41	0.50	0.20
2	AWS	0.09	0.42	0.50	0.30
3	DI	0.05	0.40	0.40	0.40
4	LM	0.06	0.42	0.50	0.20
5	PB	0.03	0.47	0.50	0.30
6	ACF	0.04	0.42	0.40	0.20
7	FBPF	0.006	0.43	0.45	0.18
8	USGP	0.003	0.41	0.40	0.10

TABLE 8. Resultant cluster with a $U(\delta_r)$ of 0.6 external connectivity evaluation.

No.	Dataset	A-CR	A-Cond	A-Ncut	A-FODF
1	ZKC	0.04	0.25	0.30	0.05
2	AWS	0.05	0.34	0.40	0.10
3	DI	0.03	0.27	0.30	0.05
4	LM	0.03	0.22	0.28	0.04
5	PB	0.02	0.33	0.40	0.20
6	ACF	0.02	0.25	0.30	0.04
7	FBPF	0.005	0.26	0.20	0.07
8	USGP	0.0015	0.37	0.23	0.05

parameters, as stated in equations 13 and 14 Section IV, favor this metric. In addition, coverage and conductance metrics are well-known internal and external connectivity metrics, respectively, because their meanings are simple, intuitive, and easy to understand. However, it will be manipulative to simply try to find the metric(s) that yield the best results for any proposed approach.

VII. MEAN RELATIVE DENSITY DEVIATION COEFFICIENT (MRDDC)

As discussed in the previous section, it is debatable whether existing quality metrics that focus only on one or both types of connections, are inadequate and inappropriate for evaluating the density proximity of clusters. This limitation impedes a comprehensive evaluation of the proposed partitioning clustering, particularly when the specific density of the resultant clusters is the objective. To address this shortcoming, a new quality metric was required. This study proposes the Mean Relative Density Deviation Coefficient (MRDDC) as a novel metric for evaluating graph_clustering processes. MRDDC considers user-specified density requirements and cluster-computed densities, enabling a more specific evaluation of the clustering process. It determines the proximity between the user-specified relative density and density of the clusters obtained by the algorithm. MRDDC can be determined using 29.

$$MRDDC = 1 - \frac{1}{k}\sum_{i=1}^{k}\left|\frac{U(\delta_r) - \delta_r(C_i)}{U(\delta_r)}\right| \tag{29}$$

where $\delta_r(C_i)$ is the density of cluster C_i, and k is the number of clusters.

The deviation from the ideal value for each cluster is zero (i.e., $U(\delta_r) - C_i(\delta_r)$ is zero). The maximum deviation (which is very unlikely to occur and should not be considered a cluster) is $U(\delta_r)$ where the cluster is a single node. By dividing this deviation by $U(\delta_r)$ and taking the absolute value, this metric lies between zero and one. Hence, the average deviation, $\frac{1}{k}\sum_{i=1}^{k}\left|\frac{U(\delta_r)-\delta_r(C_i)}{U(\delta_r)}\right|$ also lies between zero and one. However, in the case of the best clustering, this average deviation yielded a value of zero, and the worst case yielded a value of one. This is somewhat unintuitive, as most people are likely to associate a value of one as being the best and zero as vice versa. Therefore, the proposed MRDCC is the value of 1 subtracted by the average deviation (as stated in 29) to return the intuitive value of 1 for the best clustering and 0 for the worst. In summary, MRDCC identifies how close the overall resultant clusters are to $U(\delta_r)$ and hence reflects the graph clustering process. In addition, MRDCC should be applicable beyond this study. Any evaluation of a graph's clustering can benefit from MRDDC when the density deviation of the clusters relative to the graph's overall density is of interest. Table 9 presents the evaluation of the resultant clusters using MRDDC for the eight networks and for both user-specified densities.

TABLE 9. Mean relative density deviation coefficient (MRDDC).

No	Network	MRDDC	
		$U(\delta_r) = 0.4$	$U(\delta_r) = 0.6$
1	**Zachary's Karate**	0.90	1.00
2	**Aves-Weaver**	0.86	0.84
3	**Dolphins-Interaction**	1.00	0.91
4	**Les Misérables**	1.00	1.00
5	**Political-Books**	0.89	0.76
6	**American College Football**	0.92	0.95
7	**Facebook-Pages-Food**	0.88	0.94
8	**US-Grid Power**	0.90	0.90

The proposed MRDDC reflects the proposed graph clustering in this study more appropriately than all existing metrics. The Aves-Weaver has the lowest value for $U(\delta_r)$ at 0.4, and the Political-Books for both $U(\delta_r)$ at 0.6. Both share the common characteristic that $|G(K)|$ (number of cliques) is relatively low. This is most likely due to the topology of the proposed partitioning clustering method.

VIII. DISCUSSION

Although only three real-world networks were selected in the illustration, the results in Tables 3 and 4 should be considered together with the metric values listed in Table 9 to comprehend the proposed overall partitioning clustering approach. The entire network has a $U(\delta_r)$ of 1, whereas a single node has a $U(\delta_r)$ of 0. Therefore, specifying a low value of $U(\delta_r)$ value is likely to result in a high number of clusters, and the same can be said for the opposite. One aspect that has a direct influence is the topology of each network (e.g., Avg_{deg}, etc.), as shown in Table 2. In general, if Avg_{deg} of a network is high, this implies a relatively dense network (i.e., a high ratio of numbers of edges to the number of nodes), and if the network is relatively sparse (i.e., a low ratio of the number of edges to the number of nodes), where Avg_{deg} is low. Therefore, the former is likely to comprise more cliques than the latter is. In addition, if the σ of the degrees of the nodes is low, this suggests that the network is relatively dense (not necessarily in one location). However, if the σ of the degrees of the nodes is high, this suggests that the network is relatively sparse. Therefore, for the same value of $U(\delta_r)$, fewer but perhaps more nodes in each cluster can be expected from a dense network, whereas more clusters and perhaps fewer nodes in each cluster may appear in a sparse network.

During partitioning clustering, cliques with an average total degree were prioritized during expansion. Numerous experiments have been conducted on this topic. Priorities were tested on cliques with minimum and maximum total degrees as well as randomly selected. It was found that using the average total degree was preferable. Randomly selected cliques for expansion were proven unreliable and their results were unexpected and unjustifiable. When maximum or minimum total degrees were experimented with, they unfavorably resulted in clusters of various sizes (even though δ_r may be similar), but more importantly, the number of

unclustered nodes was relatively high. Hence, they were rejected, and are not discussed in depth here.

In relation to the number of unclustered nodes, the outcome is significantly influenced by the topology, particularly the Avg_{deg}. Although it may be desirable to have as few unclustered nodes as possible, the combination of both Avg_{deg}, and $U(\delta_r)$ had a direct influence on partitioning clustering in this study. This renders the well-known coverage metric inappropriate because it is not intended for use if the specific density is of interest in clustering. Among the three selected networks, the American College Football network was the densest, the US-Grid Power network was the least dense, and Zachary's Karate Club network was in between. The results of unclustered nodes for these networks for both $U(\delta_r)$ of 0.4 and 0.6 (i.e., Tables 3 and 4) reflect this nature.

It may also be impossible to expect the exact value of $U(\delta_r)$ in each cluster; however, the proposed MRDDC has proven to be a suitable and meaningful metric for this study. It also verifies that the proposed approach to partitioning clustering is effective, as their values are close to the ideal value of 1 for all selected networks. In practice, it may be prudent to conduct a quick analysis of the characteristics of the network of interest before specifying $U(\delta_r)$. Many such networks have been reported in the literature, in which large clusters are already visible. In such cases, the requirement for a specific density from the clustering may be inappropriate. However, if these obvious clusters can be extracted as subgraphs where a specific density within large clusters is required, the proposed partitioning clustering may still have an application.

Bearing in mind that this work is about discovering clusters with a specific density in a network that was not initially apparent or visible. The application of this work allows the efficient deployment, management, and monitoring of resources to clusters within networks. Such networks can be large social communities, wireless communication networks, or hardware device networks.

IX. CONCLUSION

The cutting of graphs at appropriate locations can be an intuitive approach if the clusters are clearly visible. However, in practice, many scenarios can be transformed into connected graphs, where hidden clusters may not be obvious. Hence, graph clustering has gained significant importance owing to its diverse range of applications. The research community in this field has developed numerous clustering algorithms and methods to assess them. Nevertheless, the community has not yet come up with a clustering approach that allows the type of density in the resulting clusters to be specified. This study presents the first attempt at partitioning the clustering of an unweighted undirected graph that allows users to specify the required density using the concept of 'relative density.' The work begins by classifying existing graph clustering algorithms into parameter-free and parametric categories and categorizing existing quality metrics from the perspectives of internal connectivity, external connectivity, and both. The major contribution of this study is the approach to partitioning clustering, which allows users to specify relative density. It consists of two phases: determining all simple cliques within the graph and partitioning clustering based on the expansion of a cluster by adding cliques while monitoring its relative density. To evaluate the proposed approach, eight real-world networks were selected and three were chosen for visual illustration. This paper discusses the essential findings and major factors that influence partitioning clustering. The study also revealed that existing quality metrics are deemed inappropriate for evaluating the proposed approach because they are developed for other specific objectives. Therefore, a new and appropriate metric, known as the Mean Relative Density Deviation Coefficient (MRDDC), was introduced to assess the proposed approach. The results demonstrate that the proposed approach is satisfactory with respect to the MRDDC.

X. FUTURE WORK

As this study represents the first attempt in this area, there are numerous possibilities for further research. It may be beneficial to conduct an in-depth analysis of the proposed approach, particularly from the sensitivity perspective. Because priority is given to cliques with total average degrees, situations in which several total average-degree cliques exist naturally imply multiple starting points. Therefore, it is important to explore the impact of different starting points on the partitioning clustering results, which may lead to valuable insights into the influence of the network topology. This requires the selection of appropriate networks and relative densities for further exploration. From a graph clustering perspective, this study primarily focused on partitioning clustering. Another possibility for this type of clustering is overlapping clustering, which may not be a complex task because the concept of clustered nodes can be disregarded, and each node or edge can be allowed to be a member of multiple clusters. This poses greater challenges owing to the need to determine an appropriate level of hierarchy, which may be difficult on its own, especially given the higher relative density at deeper levels. In addition, two commonly used approaches (i.e., Agglomerative and Divisive) can be investigated. Another important consideration is determining whether to use partitioning or overlapping clustering at each level. From a technical perspective, this study adopted clique-based and expansion-based approaches. However, other approaches may be possible, such as a simple edge-based approach, in which a single edge (i.e., the sum of the degrees of nodes at both endpoints) merits consideration. In addition, the mathematics of edge centrality may be combined with the current approach to develop a more efficient graph clustering algorithm.

ACKNOWLEDGMENT

The authors are grateful for the useful and constructive comments from anonymous reviewers. Provision of computing facilities with the School of Information Technology

(SIT), KMUTT, is gratefully acknowledged. The author Rohi Tariq wishes to express her deep gratitude to her parents for their unwavering support and to her husband for his encouragement throughout this research journey. She is also indebted to her late grandfather, whose constant motivation and unwavering belief in higher education continue to inspire her pursuit of knowledge.

REFERENCES

[1] K. Berahmand, S. Haghani, M. Rostami, and Y. Li, "A new attributed graph clustering by using label propagation in complex networks," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 5, pp. 1869–1883, May 2022, doi: 10.1016/j.jksuci.2020.08.013.

[2] X. Huang, H. Cheng, and J. X. Yu, "Dense community detection in multi-valued attributed networks," *Inf. Sci.*, vol. 314, pp. 77–99, Sep. 2015, doi: 10.1016/j.ins.2015.03.075.

[3] Y. Yin, Y. Zhao, H. Li, and X. Dong, "Multi-objective evolutionary clustering for large-scale dynamic community detection," *Inf. Sci.*, vol. 549, pp. 269–287, Mar. 2021, doi: 10.1016/j.ins.2020.11.025.

[4] S. E. Schaeffer, "Graph clustering," *Comput. Sci. Rev.*, vol. 1, no. 1, pp. 27–64, Aug. 2007, doi: 10.1016/j.cosrev.2007.05.001.

[5] K. Xia, X. Gu, and Y. Zhang, "Oriented grouping-constrained spectral clustering for medical imaging segmentation," *Multimedia Syst.*, vol. 26, no. 1, pp. 27–36, Feb. 2020, doi: 10.1007/s00530-019-00626-8.

[6] M. Rostami, M. Oussalah, and V. Farrahi, "A novel time-aware food recommender-system based on deep learning and graph clustering," *IEEE Access*, vol. 10, pp. 52508–52524, 2022, doi: 10.1109/ACCESS.2022.3175317.

[7] B. Shao, X. Li, and G. Bian, "A survey of research hotspots and frontier trends of recommendation systems from the perspective of knowledge graph," *Exp. Syst. Appl.*, vol. 165, Mar. 2021, Art. no. 113764, doi: 10.1016/j.eswa.2020.113764.

[8] J. MacQueen, "Classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, 1967, pp. 281–297.

[9] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 226–233.

[10] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002, doi: 10.1073/pnas.122653799.

[11] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Exp.*, vol. 2008, no. 10, Oct. 2008, Art. no. P10008, doi: 10.1088/1742-5468/2008/10/p10008.

[12] J. Bae, T. Helldin, M. Riveiro, S. Nowaczyk, M. R. Bouguelia, and G. Falkman, "Interactive clustering: A comprehensive review," *ACM Comput. Surv.*, vol. 53, no. 1, pp. 1–39, 2020, doi: 10.1145/3340960.

[13] M. Landauer, F. Skopik, M. Wurzenberger, and A. Rauber, "System log clustering approaches for cyber security applications: A survey," *Comput. Secur.*, vol. 92, May 2020, Art. no. 101739, doi: 10.1016/j.cose.2020.101739.

[14] R. Kothari and D. Pitts, "On finding the number of clusters," *Pattern Recognit. Lett.*, vol. 20, no. 4, pp. 405–416, Apr. 1999, doi: 10.1016/s0167-8655(99)00008-2.

[15] H. Liao, J. Hu, T. Li, S. Du, and B. Peng, "Deep linear graph attention model for attributed graph clustering," *Knowl.-Based Syst.*, vol. 246, Jun. 2022, Art. no. 108665, doi: 10.1016/j.knosys.2022.108665.

[16] L. Hu and C. Zhong, "An internal validity index based on density-involved distance," *IEEE Access*, vol. 7, pp. 40038–40051, 2019, doi: 10.1109/ACCESS.2019.2906949.

[17] M. Jafarzadegan, F. Safi-Esfahani, and Z. Beheshti, "Combining hierarchical clustering approaches using the PCA method," *Exp. Syst. Appl.*, vol. 137, pp. 1–10, Dec. 2019, doi: 10.1016/j.eswa.2019.06.064.

[18] P. Dawyndt, H. D. Meyer, and B. D. Baets, "The complete linkage clustering algorithm revisited," *Soft Comput.*, vol. 9, no. 5, pp. 385–392, May 2005, doi: 10.1007/s00500-003-0346-3.

[19] A. Lancichinetti and S. Fortunato, "Community detection algorithms: A comparative analysis," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 80, no. 5, Nov. 2009, doi: 10.1103/physreve.80.056117.

[20] V. A. Traag, L. Waltman, and N. J. van Eck, "From Louvain to leiden: Guaranteeing well-connected communities," *Sci. Rep.*, vol. 9, no. 1, p. 5233, Mar. 2019, doi: 10.1038/s41598-019-41695-z.

[21] S. Sankar, S. Ramasubbareddy, A. K. Luhach, A. Nayyar, and B. Qureshi, "CT-RPL: Cluster tree based routing protocol to maximize the lifetime of Internet of Things," *Sensors*, vol. 20, no. 20, p. 5858, Oct. 2020, doi: 10.3390/s20205858.

[22] J. Amutha, S. Sharma, and S. K. Sharma, "Strategies based on various aspects of clustering in wireless sensor networks using classical, optimization and machine learning techniques: Review, taxonomy, research findings, challenges and future directions," *Comput. Sci. Rev.*, vol. 40, May 2021, Art. no. 100376, doi: 10.1016/j.cosrev.2021.100376.

[23] Y. Liao, H. Qi, and W. Li, "Load-balanced clustering algorithm with distributed self-organization for wireless sensor networks," *IEEE Sensors J.*, vol. 13, no. 5, pp. 1498–1506, May 2013, doi: 10.1109/JSEN.2012.2227704.

[24] P. T. Hanh, P. D. Thanh, and H. T. Binh, "Evolutionary algorithm and multifactorial evolutionary algorithm on clustered shortest-path tree problem," *Inf. Sci.*, vol. 553, pp. 280–304, Apr. 2021, doi: 10.1016/j.ins.2020.10.024.

[25] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, nos. 3–5, pp. 75–174, Feb. 2010, doi: 10.1016/j.physrep.2009.11.002.

[26] Y. Gao, X. Yu, and H. Zhang, "Graph clustering using triangle-aware measures in large networks," *Inf. Sci.*, vol. 584, pp. 618–632, Jan. 2022, doi: 10.1016/j.ins.2021.11.008.

[27] M. Adraoui, A. Retbi, M. K. Idrissi, and S. Bennani, "Maximal cliques based method for detecting and evaluating learning communities in social networks," *Future Gener. Comput. Syst.*, vol. 126, pp. 1–14, Jan. 2022, doi: 10.1016/j.future.2021.07.034.

[28] X. Bao and L. Wang, "A clique-based approach for co-location pattern mining," *Inf. Sci.*, vol. 490, pp. 244–264, Jul. 2019, doi: 10.1016/j.ins.2019.03.072.

[29] S. K. Gupta and D. D. P. Singh, "CBLA: A clique based Louvain algorithm for detecting overlapping community," *Proc. Comput. Sci.*, vol. 218, pp. 2201–2209, Jan. 2023, doi: 10.1016/j.procs.2023.01.196.

[30] G. W. Flake, S. Lawrence, and C. L. Giles, "Efficient identification of web communities," in *Proc. 6th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2000, pp. 150–160.

[31] S. Papadopoulos, Y. Kompatsiaris, A. Vakali, and P. Spyridonos, "Community detection in social media," *Data Mining Knowl. Discovery*, vol. 24, no. 3, pp. 515–554, May 2012, doi: 10.1007/s10618-011-0224-z.

[32] A. Tandon, A. Albeshri, V. Thayananthan, W. Alhalabi, and S. Fortunato, "Fast consensus clustering in complex networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 99, no. 4, Apr. 2019, Art. no. 042301, doi: 10.1103/physreve.99.042301.

[33] J. Hou, W. Liu, E. Xu, and H. Cui, "Towards parameter-independent data clustering and image segmentation," *Pattern Recognit.*, vol. 60, pp. 25–36, Dec. 2016, doi: 10.1016/j.patcog.2016.04.015.

[34] A. Chhabra, K. Masalkovaite, and P. Mohapatra, "An overview of fairness in clustering," *IEEE Access*, vol. 9, pp. 130698–130720, 2021, doi: 10.1109/ACCESS.2021.3114099.

[35] F. H. Kuwil, F. Shaar, A. E. Topcu, and F. Murtagh, "A new data clustering algorithm based on critical distance methodology," *Exp. Syst. Appl.*, vol. 129, pp. 296–310, Sep. 2019, doi: 10.1016/j.eswa.2019.03.051.

[36] A. Clauset, M. E. J. Newman, and C. Moore, "Finding community structure in very large networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 70, no. 6, Dec. 2004, Art. no. 066111, doi: 10.1103/physreve.70.066111.

[37] G. Rossetti, L. Milli, and R. Cazabet, "CDLIB: A Python library to extract, compare and evaluate communities from complex networks," *Appl. Netw. Sci.*, vol. 4, no. 1, pp. 1–26, Dec. 2019, doi: 10.1007/s41109-019-0165-9.

[38] A. Hagberg, P. Swart, and D. S. Chult, "Exploring network structure, dynamics, and function using NetworkX," Los Alamos Nat. Lab. (LANL), Los Alamos, NM, USA, Tech. Rep. TR-0200, 2008. [Online]. Available: https://www.osti.gov/servlets/purl/960616

[39] G. Csárdi and T. Nepusz, "The igraph software package for complex network research," *InterJ., Complex Syst.*, vol. 1695, no. 5, pp. 1–9, 2006.

[40] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proc. Nat. Acad. Sci. USA*, vol. 105, no. 4, pp. 1118–1123, Jan. 2008, doi: 10.1073/pnas.0706851105.

[41] M. Rosvall, D. Axelsson, and C. T. Bergstrom, "The map equation," *Eur. Phys. J. Special Topics*, vol. 178, no. 1, pp. 13–23, Nov. 2009, doi: 10.1140/epjst/e2010-01179-1.

[42] R. Lambiotte and M. Rosvall, "Ranking and clustering of nodes in networks with smart teleportation," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 85, no. 5, May 2012, Art. no. 056107, doi: 10.1103/physreve.85.056107.

[43] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 76, no. 3, Sep. 2007, Art. no. 036106, doi: 10.1103/physreve.76.036106.

[44] A. M. Fiscarelli, M. R. Brust, G. Danoy, and P. Bouvry, "Local memory boosts label propagation for community detection," *Appl. Netw. Sci.*, vol. 4, no. 1, pp. 1–17, Dec. 2019, doi: 10.1007/s41109-019-0210-8.

[45] H. Li, R. Zhang, Z. Zhao, and X. Liu, "LPA-MNI: An improved label propagation algorithm based on modularity and node importance for community detection," *Entropy*, vol. 23, no. 5, p. 497, Apr. 2021, doi: 10.3390/e23050497.

[46] P. Pons and M. Latapy, "Computing communities in large networks using random walks," in *Proc. 20th Int. Symp. Comput. Inf. Sci.* Istanbul, Turkey: Springer, Oct. 2005, pp. 284–293.

[47] W.-B. Xie, Y.-L. Lee, C. Wang, D.-B. Chen, and T. Zhou, "Hierarchical clustering supported by reciprocal nearest neighbors," *Inf. Sci.*, vol. 527, pp. 279–292, Jul. 2020, doi: 10.1016/j.ins.2020.04.016.

[48] H. C. Rustamaji, Y. S. Suharini, A. A. Permana, W. A. Kusuma, S. Nurdiati, I. Batubara, and T. Djatna, "A network analysis to identify lung cancer comorbid diseases," *Appl. Netw. Sci.*, vol. 7, no. 1, pp. 1–23, Dec. 2022, doi: 10.1007/s41109-022-00466-y.

[49] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 74, no. 3, Sep. 2006, Art. no. 036104, doi: 10.1103/physreve.74.036104.

[50] L. Galluccio, O. Michel, P. Comon, and A. O. Hero, "Graph based k-means clustering," *Signal Process.*, vol. 92, no. 9, pp. 1970–1984, Sep. 2012, doi: 10.1016/j.sigpro.2011.12.009.

[51] A. Lancichinetti and S. Fortunato, "Consensus clustering in complex networks," *Sci. Rep.*, vol. 2, no. 1, pp. 1–7, Mar. 2012, doi: 10.1038/srep00336.

[52] S. Fortunato and M. Barthélemy, "Resolution limit in community detection," *Proc. Nat. Acad. Sci. USA*, vol. 104, no. 1, pp. 36–41, Jan. 2007, doi: 10.1073/pnas.0605965104.

[53] F. Parés, D. G. Gasulla, A. Vilalta, J. Moreno, E. Ayguadé, J. Labarta, and T. Suzumura, "Fluid communities: A competitive, scalable and diverse community detection algorithm," in *Proc. 6th Complex Netw. Their Appl.* Lyon, France: Springer, Nov./Dec. 2017, pp. 229–240.

[54] A. Ng, M. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Adv. Neural Inf. Process. Syst.*, vol. 14, 2001, pp. 1–8.

[55] U. von Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, 2007, doi: 10.1007/s11222-007-9033-z.

[56] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Stat. Soc., Ser. B*, vol. 39, no. 1, pp. 1–22, Sep. 1977, doi: 10.1111/j.2517-6161.1977.tb01600.x.

[57] A. Biswas and B. Biswas, "Defining quality metrics for graph clustering evaluation," *Exp. Syst. Appl.*, vol. 71, pp. 1–17, Apr. 2017, doi: 10.1016/j.eswa.2016.11.011.

[58] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowl. Inf. Syst.*, vol. 42, no. 1, pp. 181–213, Jan. 2015, doi: 10.1007/s10115-013-0693-z.

[59] J. Casas-Roma, J. Herrera-Joancomartí, and V. Torra, "Anonymizing graphs: Measuring quality for clustering," *Knowl. Inf. Syst.*, vol. 44, no. 3, pp. 507–528, Sep. 2015, doi: 10.1007/s10115-014-0774-7.

[60] Y. Zhou, H. Cheng, and J. X. Yu, "Graph clustering based on structural/attribute similarities," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 718–729, Aug. 2009, doi: 10.14778/1687627.1687709.

[61] H. Almeida, D. Guedes, W. Meira, and M. J. Zaki, "Is there a best quality metric for graph clusters?" in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*. Berlin, Germany: Springer, 2011, pp. 44–59.

[62] S. Emmons, S. Kobourov, M. Gallant, and K. Börner, "Analysis of network clustering algorithms and cluster quality metrics at scale," *PLoS ONE*, vol. 11, no. 7, Jul. 2016, Art. no. e0159161, doi: 10.1371/journal.pone.0159161.

[63] T. Chakraborty, A. Dalmia, A. Mukherjee, and N. Ganguly, "Metrics for community analysis: A survey," *ACM Comput. Surv.*, vol. 50, no. 4, pp. 1–37, Jul. 2018, doi: 10.1145/3091106.

[64] P. Wagenseller, F. Wang, and W. Wu, "Size matters: A comparative analysis of community detection algorithms," *IEEE Trans. Computat. Social Syst.*, vol. 5, no. 4, pp. 951–960, Dec. 2018, doi: 10.1109/TCSS.2018.2875626.

[65] D. Hric, R. K. Darst, and S. Fortunato, "Community detection in networks: Structural communities versus ground truth," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 90, no. 6, Dec. 2014, Art. no. 062805, doi: 10.1103/physreve.90.062805.

[66] L. Hagen and A. B. Kahng, "New spectral methods for ratio cut partitioning and clustering," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 11, no. 9, pp. 1074–1085, Sep. 1992, doi: 10.1109/43.159993.

[67] P. K. Chan, M. D. F. Schlag, and J. Y. Zien, "Spectral K-way ratio-cut partitioning and clustering," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 13, no. 9, pp. 1088–1096, 1994, doi: 10.1109/43.310898.

[68] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000, doi: 10.1109/34.868688.

[69] Z. Lu, J. Wahlström, and A. Nehorai, "Community detection in complex networks via clique conductance," *Sci. Rep.*, vol. 8, no. 1, pp. 1–16, Apr. 2018, doi: 10.1038/s41598-018-23932-z.

[70] F. Aidi, D. Archambault, and G. Melançon, "Evaluating the quality of clustering algorithms using cluster path lengths," in *Proc. 10th Adv. Data Mining, Appl. Theor. Aspects*. Berlin, Germany: Springer, Jul. 2010, pp. 42–56, doi: 10.1007/978-3-642-14400-4-4.

[71] G. Mishra and S. K. Mohanty, "RDMN: A relative density measure based on MST neighborhood for clustering multi-scale datasets," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 419–432, Jan. 2022, doi: 10.1109/TKDE.2020.2982400.

[72] Y. Wang and Y. Yang, "Relative density-based clustering algorithm for identifying diverse density clusters effectively," *Neural Comput. Appl.*, vol. 33, no. 16, pp. 10141–10157, Aug. 2021, doi: 10.1007/s00521-021-05777-2.

[73] Q.-B. Liu, S. Deng, C.-H. Lu, B. Wang, and Y.-F. Zhou, "Relative density based k-nearest neighbors clustering algorithm," in *Proc. Int. Conf. Mach. Learn. Cybern.*, 2003, pp. 17–133, doi: 10.1109/icmlc.2003.1264457.

[74] H. A. Chowdhury, D. K. Bhattacharyya, and J. K. Kalita, "UIFDBC: Effective density based clustering to find clusters of arbitrary shapes without user input," *Exp. Syst. Appl.*, vol. 186, Dec. 2021, Art. no. 115746, doi: 10.1016/j.eswa.2021.115746.

[75] S. Harenberg, G. Bello, L. Gjeltema, S. Ranshous, J. Harlalka, R. Seay, K. Padmanabhan, and N. Samatova, "Community detection in large-scale networks: A survey and empirical evaluation," *WIREs Comput. Statist.*, vol. 6, no. 6, pp. 426–439, Nov. 2014, doi: 10.1002/wics.1319.

ROHI TARIQ received the M.Sc. degree in computer science from the University of Malakand, Malakand, Pakistan, in 2009, and the M.S. degree in computer science from the ISPAR, Bacha Khan University Charsadda, Pakistan, in 2014. She is currently pursuing the Ph.D. degree with the School of Information Technology (SIT), King Mongkut's University of Technology Thonburi, Bangkok, Thailand.

Her research interests include computer science, artificial intelligence, machine learning, big data mining, and graph clustering. Her academic journey and diverse research interests showcase a strong commitment to advancing knowledge in these cutting-edge areas of study.

KITTICHAI LAVANGNANANDA (Senior Member, IEEE) received the B.Sc. degree in computational science from the University of Hull, U.K., in 1985, the M.Sc. degree in computing from Cardiff University, U.K., in 1987, and the Ph.D. degree in artificial intelligence from the Mechanical Engineering Centre (MEC), Cardiff University, in 1995. He is currently an Associate Professor and the Head of the Software Division Technology, School of Information Technology (SIT), King Mongkut's University of Technology Thonburi (KMUTT), Thailand. He is also an active Research Member of the Data and Knowledge Engineering Laboratory (D-Lab), SIT. His research interest includes computational intelligence-related areas such as data mining, evolutionary computation, machine learning, and neural networks, along with their applications. He is also involved in various academic activities and holds the position of an Editorial Board Member of the *Cogent Engineering* journal.

PASCAL BOUVRY received the bachelor's degree in economic and social sciences and the master's degree in computer science from the University of Namur, Belgium, in 1991, and the Ph.D. degree (Hons.) in computer science from the University of Grenoble (INPG), France, in 1994.

He is currently a Full Professor with the University of Luxembourg, ''Chargé de Mission auprès du Recteur'' in charge of the University High-Performance Computing, heading the Parallel Computing and Optimization Group (PCOG), SnT.

Dr. Bouvry is the Luxembourg national delegate for (Partnership for Advanced Computing in Europe). He is also active in various scientific committees and technical workgroups (IEEE CIS Cloud Computing Vice-Chair, IEEE TCSC GreenIT Steering Committee, ERCIM WG, ANR, and COST TIST). The mission of PRACE is to enable high-impact scientific discovery and engineering research and development across all disciplines to enhance European competitiveness for the benefit of society. He is a part of the editorial boards of IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, *Journal on Communications*, *Sustainable Computing* (Springer), *IET Cyber-Physical Systems: Theory and Applications*, and *Swarm and Evolutionary Computation* (Elsevier).

PORNCHAI MONGKOLNAM received the Ph.D. degree in computer science from Arizona State University, in 2003. He is currently an Associate Professor with the School of Information Technology (SIT), King Mongkut's University of Technology Thonburi (KMUTT), Thailand. His research interests include artificial intelligence, data visualization, applied information technology, and software engineering.