Unified Dense Subgraph Detection: Fast Spectral Theory Based Algorithms

Wenjie Feng, Shenghua Liu, *Member, IEEE*, Danai Koutra, and Xueqi Cheng, *Senior Member, IEEE*

***Abstract*— How can we effectively detect fake reviews or fraudulent links on a website? How can we spot communities that suddenly appear based on users' interactions? And how can we efficiently find the minimum cut in a large graph? All of these are related to the finding of dense subgraphs, a significant primitive problem in graph analysis with extensive applications across various domains. In this paper, we focus on formulating the problem of the densest subgraph detection and theoretically compare and contrast several correlated problems. Moreover, we propose a unified framework, GENDS, for the densest subgraph detection, provide some theoretical analysis based on the network flow and spectral graph theory, and devise simple and computationally efficient algorithms, SPECGDS and GEPGDS, to solve it by leveraging the spectral properties and greedy search. We conduct thorough experiments on 40 real-world networks with up to 1.47 billion edges from various domains. We demonstrate that our SPECGDS yields up to 58.6 ×speedup and achieves better or approximately equal-quality solutions for the densest subgraph detection compared to the baselines. GEPGDS also reveals some properties of generalized eigenvalue problems for the GENDS. Also, our methods scale linearly with the graph size and are proven effective in applications such as finding collaborations that appear suddenly in an extensive, time-evolving co-authorship network.**

***Index Terms*—Algorithm, anomaly detection, dense subgraph, graph spectral theory, large graph mining.**

I. Introduction

How can we capture the most contrasted groups in temporal or dynamic graphs, e.g., the hot topics or collaborations in a research community that appear suddenly? How can we efficiently determine the minimum cut for a large graph? And how can we find the most suspicious users based on their behaviors or spot the largest group with consensus opinions on controversial issues? All these real-world problems are related to the densest subgraph detection task.

Dense pattern mining in graphs is a key primitive task for extracting useful information and capturing underlying principles in relational data. It has benefited various application domains [1], such as capturing functional groups in biology [2], traffic patterns in human behaviors [3], communities in social networks [4], anomaly detection in financial and other networks [5], etc. The densest subgraph problem has garnered significant interest in the practice because it can be solved exactly in polynomial time and has an adequate approximation in almost linear time. Goldberg's maximum flow algorithm [6] and Charikar's LP-based algorithm [7] provide the exact solution; Charikar [7] proved that the simple greedy algorithm is guaranteed to find a result of better quality than the factor 2-approximation with linear time scales to the graph size. However, these algorithms still incur a prohibitive computational cost for the massive graphs that arise in modern data science applications without fully considering and utilizing the properties of real-world data.

To the best of our knowledge, there is no related work to study the connections among the above problems. Here, we summarize the differences and relations of some well-known allied problems, including detecting communities with sparse cut or suspicious dense subgraphs. We propose a unified formulation, *the generalized densest subgraph* (GENDS*)* problem, which explicitly highlights their relationship formally, that is, they are the different instantiations of the densest subgraph corresponding to different parameter configurations. It also leads to consistent and feasible methods for solving them. Theoretically, we provide the analysis through the lens of network flow and graph spectral theory. We thus devise efficient detection algorithms, SPECGDS and GEPGDS, that leverage the spectral properties of graphs and greedy search to solve the generalized problem.

With thorough experiments using 40 diverse real-world networks, we demonstrate that our algorithms are fast, highly effective, and linearly scalable. SPECGDS yields a 58.6 × speedup and achieves almost better or equal quality than baselines, even for a large network with 1.47 billion edges. Experiment results from GEPGDS also reveal some properties of generalized eigenvalues w.r.t. GENDS. Moreover, we find some interesting patterns, e.g., contrast dense collaborations in the DBLP co-authorship data. The main contributions are summarized as follows.

Manuscript received 16 March 2022; revised 9 January 2023; accepted 20 April 2023. Date of publication 17 July 2023; date of current version 6 February 2024. This work was supported in part by the National Science Foundation of China under Grants U1911401, U21B2046, 61872206, and 62377043, and in part by National Science Foundation CAREER under Grant IIS 1845491. This paper is also supported by 2022 Tencent Wechat Rhino-Bird Focused Research Program. Recommended for acceptance by X. Xiao. *(Corresponding authors: Wenjie Feng and Shenghua Liu.)*

Wenjie Feng is with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China, and also with the Institute of Data Science, National University of Singapore, Singapore 117602.

Shenghua Liu and Xueqi Cheng are with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: liu.shengh@gmail.com; cxq@ict.ac.cn).

Danai Koutra is with the University of Michigan, Ann Arbor, MI 48109 USA (e-mail: dkoutra@umich.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TKDE.2023.3272574, provided by the authors.

Digital Object Identifier 10.1109/TKDE.2023.3272574

- *Theory & Correspondences:* We propose the generalized densest subgraph detection formulation, GenDS, to unify several correlated problems, and provide theoretical analysis in the principle of network flow and spectral graph theory.
- *Algorithm:* We devise, SpecGDS and GepGDS, fast and scalable algorithms to solve the unified problem.
- *Experiment:* We conduct thorough empirical verification of various real-world graphs to validate the efficiency and linear scalability of SpecGDS and GepGDS. They also find interesting patterns, including contrast-dense subgraphs in co-authorship relations.

Reproducibility: Our open-sourced code and used dataset are online available.[1]

II. Related Work

In this section, we summarize works about the densest subgraph problem and various detection methods to capture those dense patterns in different applications.

Finding the densest subgraph in a large input graph is a widely studied problem [1], [8], [9]. The most recent tutorial [10] comprehensively summarizes the densest subgraph discovery, including different types of graphs, density metrics and variants, detection algorithms, and applications. In general, such a problem aims to find a set of nodes in a given input graph to maximize some notion of density. The so-called *densest subgraph problem* (DSP) finds a subgraph that maximizes the degree density, which is the average of the weights of all its edges. When the edge weights are non-negative, the densest subgraph can be identified optimally in polynomial time using maximum flow algorithms [6], [11]. Based on the maximum flow, Tatti and Gionis [12] present a local dense graph decomposition similar to the well-known k-core decomposition, with the additional property that its components are arranged in order of their densities. There are other different density metrics [13], including k-clique density [14], [15], edge-surplus for α-quasi-cliques [16], triangle density, k-core [17], general patterns [18], etc.

However, obtaining the exact solution with maximum flow requires expensive computations despite the theoretical progress achieved in recent years, thus making it prohibitive for large graphs. Charikar [7] introduces a linear-programming formulation of the problem and shows that the greedy algorithm proposed by Asashiro et al. [19] produces a $\frac{1}{2}$-approximation of the optimum density in linear time. [20] proposes an optimization model for local community detection by extending the densest subgraph problem. [21] devises an efficient algorithm via convex programming, that can compute exact local dense decomposition in real-world graphs with up to billions of edges and proposes a $(1+\epsilon)$-approximation solution based on the Frank-Wolfe algorithm. [18] develops exact and approximate solutions for the densest subgraph discovery by leveraging the k-core, which is suitable for edge and h-clique densities. Boob et al. [22] developed a simple iterative peeling algorithm, Greedy++, to improve the output quality of the subgraph over Charikar's greedy peeling algorithm [7] by drawing insights from the iterative approaches of convex optimization; the history of peeling information of nodes will help to escape the local optimal to some extent. [23] exploited the super-modular maximization and proposed a more efficient $(1-\epsilon)$-approximation algorithm in deterministic $\tilde{O}(m/\epsilon)$ time via approximate flow techniques for DSP, which gives evidence of the convergence and theoretical soundness of Greedy++; it also developed the 2-approximation peeling algorithm for the densest-at-least-k subgraph. [24] proposed the efficient ds-index to report all minimal densest subgraphs in a graph and enumerate them, where the minimal densest subgraph is strictly denser than all of its proper subgraphs. [25] provided an algorithm for maintaining a $(1-\epsilon)$-approximate densest subgraph within $O(\text{poly}\log n)$ time over dynamic directed graphs and extended to the problem of vertex-weighted static graphs. [18] improved the flow-based exact algorithm by locating the dense subgraph in a specific k-core. The k-core-based exact and approximate algorithms can be generalized by considering an arbitrary pattern graph and aiming to maximize the average number of occurrences of the pattern in the resulting subgraph. [26] proposed $[x, y]$-core-based algorithms with the divide-and-conquer strategy to find the densest subgraph for the directed graphs.

In addition to the original form of DSP, there are numerous variations and generalizations. However, for graphs with negative edge weights, the above problem becomes NP-hard [27]. When restrictions on the size lower bound are specified, the *densest k-subgraph problem* (DkS) becomes NP-complete [28], and there does not exist any PTAS (i.e., a polynomial-time algorithm A_ϵ with an approximation ratio of $(1+\epsilon)$, for each constant $\epsilon > 0$) under a reasonable complexity assumption. Other measures include edge surplus [16], triangle and k-clique density [14], and discounted average degree [29]. Rossi et al. [30] developed a fast, parallel maximum clique algorithm for sparse graphs; Mitzenmacher et al. [31] used the densest subgraph sparsifier with a sampling schema for the input graph and computed the densest subgraph in the resulting sparse graph. The densest subgraph problem is generalized to those on hypergraphs [32], [33], multilayer graphs [17], and uncertain graphs [34]. There are also extensions in dynamic [33], [35] and streaming settings [36], [37].

Another line of related research includes contrast graph pattern mining, which aims to discover subgraphs that manifest drastic differences between graphs. Yang et al. [38] proposed detecting the density contrast subgraphs, which is equivalent to mining the densest subgraph from a "difference" graph, and employed a local search algorithm to find the solution. Tsourakakis et al. [27] focused on the risk-aversion dense subgraph pattern for a graph with small negative weights and extended the greedy algorithm for this case. For signed networks, [39] mines dense subgraph patterns corresponding to *finding the gang in a war* problem; [39] detects the k-oppositive cohesive groups by solving a quadratic optimization problem for signed networks. Also, [40] considers the fairness constraints for the densest subgraph and devises approximation algorithms to find the densest fair subgraph with an arbitrary 2-coloring.

[1] http://www.github.com/wenchieh/specgreedy.

TABLE I
SYMBOLS AND DEFINITIONS USED IN THE PAPER

Symbol	Definition
$\mathcal{G} = (V, E)$	Undirected graph with node set V and edge set $E \subseteq V \times V$
$\hat{\mathcal{G}} = (L \cup R, E)$	Bipartite graph with node set, $L \cup R$ (left, right), and edge set $E \subseteq L \times R$
$\mathcal{G}_r = (V, E_r)$	Positive residual graph with node set V and residual edge set E_r
$\boldsymbol{x}$	Characteristic vector of a node subset
$\boldsymbol{u}, \boldsymbol{v}$	Eigenvector or singular vector of a matrix
$\boldsymbol{d}, \mathbf{D}$	Node degree vector and its diagonal matrix
$\mathbf{D}_{\boldsymbol{c}}$	Diagonal matrix for a constant vector $\boldsymbol{c}$
$\mathbf{A}, \mathbf{L} = \mathbf{D} - \mathbf{A}$	Adjacency and Laplacian matrix of a graph

The dense subgraphs are used to detect communities [2], [41], [42] and anomalies [43], [44], [45]. As one of the key characteristics, density, as well as other similar metrics like modularity [46], assortativity, and local density [47], are used as the (part of) optimization objective to detect the community structures. SPOKEN [43] utilizes the "eigenspokes" pattern of community in the EE-plots produced by pairs of eigenvectors of a graph, which is applied to fraud detection. CROSSSPOT [48] finds suspicious dense blocks by greedily adjusting the seed until it converges to a local optimum. Fraudar [45] proposed using the greedy method, which incorporates the suspiciousness of nodes and edges during optimization. In addition, dense pattern detection also generalizes to tensors [49], [50]. Similar greedy algorithms also achieve good results and can be used to detect anomalies.

Besides, many works utilize the spectral properties of graphs to detect communities [4], [43], [51] and dense subgraphs [52], [53], [54], or partition the input graph [55], [56], [57]; they can be extended to hyper or high-order graphs [58].

III. PROBLEM AND CORRESPONDENCES

A. Preliminaries and Definitions

Throughout the paper, vectors are denoted by boldface lowercase letters (e.g., $\boldsymbol{x}$), matrices are boldface uppercase letters (e.g., $\mathbf{A}$), and sets are uppercase letters (e.g., S). Unless stated otherwise, a vector is assumed to be a column vector. The operator $|\cdot|$ denotes the cardinality of a set or the number of non-zero ($\mathtt{nnz}$) elements in a vector, and $\|\cdot\|$ is the ℓ_2-norm of a vector. We denote $[x] \equiv \{1, \ldots, x\}$ for brevity. Table I gives the complete list of symbols used in the paper.

Consider an undirected graph $\mathcal{G} = (V, E)$ with $|V| = n$. Let $S \subseteq V$ and $E(S)$ be the edges of the subgraph $\mathcal{G}(S)$ induced by the subset S, i.e., $E(S) = \{e_{ij} : v_i, v_j \in S \wedge e_{ij} \in E\}$. Let $\mathbf{A} = (a_{ij}) \in \mathbb{R}^{n \times n}$ be the adjacency matrix of $\mathcal{G}$ with $a_{ij} \geq 0$. Here, we don't specify $\mathcal{G}$ to be the unweighted graph, and we also use $|E(S)| = \sum_{e \in E(S)} a_{ij}$ for the weighted graph whenever there is no ambiguity.

Let $\boldsymbol{x}$ with size n be the characteristic vector of the subset S of V ($\boldsymbol{x}_i = 1$ if $i \in S$, and $\boldsymbol{x}_i = 0$ otherwise), the average degree density of the subgraph $\mathcal{G}(S)$, being the most commonly used density measure for the densest subgraph problem, is defined by Charikar [7] as

$$g(S) = \frac{|E(S)|}{|S|} = \frac{1}{2} \cdot \frac{\boldsymbol{x}^\top \mathbf{A} \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}, \quad \boldsymbol{x} \in [0,1]^n, \tag{1}$$

and avoids the trivial solution by limiting $\|\boldsymbol{x}\| \geq 1$. This result is also observed by Kannan and Vinay [59].

Generally, Hooi et al. [45] proposed considering the node weight (some constant for each node) as the total mass of the subgraph, thus the density of $\mathcal{G}(S)$ is

$$\begin{aligned} g(S) &= \frac{|E(S)| + \sum_{i \in S} c_i}{|S|} = \frac{\boldsymbol{x}^\top \mathbf{A} \boldsymbol{x}}{2 \cdot \boldsymbol{x}^\top \boldsymbol{x}} + \frac{\boldsymbol{x}^\top \mathbf{D}_{\boldsymbol{c}} \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}} \\ &= \frac{1}{2} \frac{\boldsymbol{x}^\top (\mathbf{A} + 2\mathbf{D}_{\boldsymbol{c}}) \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}, \quad \boldsymbol{x} \in [0,1]^n, \end{aligned} \tag{2}$$

where $c_i \in \mathbb{R}^+$ is the weight of the node i and $\mathbf{D}_{\boldsymbol{c}} = diag(\boldsymbol{c})$ is the diagonal matrix of the weight vector $\boldsymbol{c} = [c_1, \ldots, c_n]$.

In addition to dense subgraphs of a single graph, we also consider the "contrast" patterns of cross-graphs, i.e., a subset of nodes with significantly different edges or edge weights in two given graphs with the same nodeset, e.g., different snapshots of a dynamic graph.

B. Generalized Densest Subgraph Problem.

Therefore, we propose a generalized densest subgraph detection problem that unifies and revisits various well-known existing formulations, that is,

Problem 1 (GenDS: Generalized Densest Subgraph Detection). Given a graph $\mathcal{G} = (V, E^P)$ and its contrast graph $\mathcal{G}' = (V, E^Q)$ with $|V| = n$, *find* the optimal subset $S^* \subseteq V$ and $|S^*| \geq 1$ *such that*

$$\begin{aligned} S^* &= \arg\max\nolimits_{S \subseteq V, |S| \geq 1} g(S; \mathcal{G}, \mathcal{G}') \\ &= \arg\max\nolimits_{\boldsymbol{x} \in [0,1]^n, \|\boldsymbol{x}\| \geq 1} \frac{\boldsymbol{x}^\top \mathbf{P} \boldsymbol{x}}{\boldsymbol{x}^\top \mathbf{Q} \boldsymbol{x}}, \end{aligned} \tag{3}$$

where $\mathbf{P} = \mathbf{A} + 2\mathbf{D}_{\boldsymbol{c}}$ and $\mathbf{Q} = \mathbf{A}' + \gamma \mathbf{I}$ with $\mathbf{A}, \mathbf{A}'$ being the adjacency matrix of $\mathcal{G}$ and $\mathcal{G}'$ resp., and $\gamma > 0$. $g_{opt} = g(S^*; \mathcal{G}, \mathcal{G}')$ is the optimal subgraph.

Here, we define $\tilde{\mathbf{A}}' = \mathbf{A}' + \gamma \mathbf{I}$ as the *augmented adjacency matrix* of the graph $\mathcal{G}'$ where $\mathbf{I}$ is an identity matrix. The denominator in (3) simultaneously considers the size of the node subset and their connections in the subgraph $\mathcal{G}'(S)$, if any. Specifically, if $\mathcal{G}'$ is an empty graph, $\mathbf{Q}$ degenerates to a γ-scaled identity matrix by only considering subgraph size in GENDS. $\mathbf{P}$ also becomes an augmented adjacency matrix of $\mathcal{G}$ if node weights are equal, i.e., $c_i = c > 0$.

As shown in Theorem 1, our proposed GENDS problem is more general for DSP, and many existing dense subgraph-based formulations are special cases.

Theorem 1: GENDS is a general framework for the MinQuotientCut, the densest subgraph detection (Charikar), FAIRDS (fair densest subgraph), Fraudar (suspicious dense subgraph), SPARSECUTDS (dense community with sparse cut), TEMPDS

TABLE II
SUMMARY FOR CORRESPONDENCE TO PROBLEM GENDS

	Method	matrix $\mathbf{P}$					matrix $\mathbf{Q}$				Constraint
1	MinQuotientCut [55]	$\mathbf{A}$	$-$		$\mathbf{D}$	$= -\mathbf{L}$			$\mathbf{I}$		$\|\boldsymbol{x}\| < n$
2	Charikar [7]	$\mathbf{A}$							$\mathbf{I}$		$\|\boldsymbol{x}\| \geq 1$
3	FAIRDS [40]	$\mathbf{A}_F$							$\mathbf{I}$		
4	Fraudar [45]	$\mathbf{A}$	$+$	2	$\mathbf{D}_{\boldsymbol{w}}$				$\mathbf{I}$		
5	SPARSECUTDS[1] [20]	$\mathbf{A}$	$-$	$\frac{2\cdot\alpha}{2\alpha+1}$	$\mathbf{D}$				$\mathbf{I}$		
6	TEMPDS [2]	$\mathbf{A}_t$					$\mathbf{A}_{t-1}$	$+$	$2\ \mathbf{I}$	$= \tilde{\mathbf{A}}_{t-1}$	
7	Risk-averse DS [27]	$\mathbf{A}^+$	$+$	γ_1	$\mathbf{I}$	$= \tilde{\mathbf{A}}^+$	$\mathbf{A}^-$	$+$	$\gamma_2\mathbf{I}$	$= \tilde{\mathbf{A}}^-$	
	GENDS[2]	$\mathbf{A}$	$+$	2	$\mathbf{D}_{\boldsymbol{c}}$		$\mathbf{A}'$	$+$	$\gamma\ \mathbf{I}$	$= \tilde{\mathbf{A}}'$	

[1] The contrast subgraph pattern detection [60] equals to set $\alpha = 1$; $\alpha = \frac{1}{2}$ is considered in [61] for the community detection.
[2] Bipartite graphs can be transformed into an undirected graph as Lemma 1 shows.

(temporal dense subgraph), and Risk-averse DS (consensus dense subgraph), and more.

The following remarks provide detailed instantiations of GENDS for several problems. Table II summarizes the setting and provides the corresponding equation, carefully aligned to highlight the correspondences to GENDS.

Remark 1. MinQuotientCut: The *optimal quotient cut ratio* problem aims at partitioning a graph into two parts with the minimum cut size.

Given $S \subseteq V$, let the set of cut edges for S be $cut(S) = \{(u, v) \in E \,|\, u \in S, v \in V \setminus S\}$ and the corresponding characteristic vector be $\boldsymbol{x}$, the cut size can be represented as $|cut(S)| = \sum_{e_{ij} \in E} a_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j)^2 = \boldsymbol{x}^\top(\mathbf{D} - \mathbf{A})\boldsymbol{x} = \boldsymbol{x}^\top\mathbf{L}\boldsymbol{x}$, the cut ratio of S is $Rcut(S) = \frac{|cut(S)|}{\min\{|S|, |V \setminus S|\}}$. Assume that, without loss of generality, S is smaller than its complement set $V \setminus S$, we obtain the minimum cut ratio by maximizing $-\frac{\boldsymbol{x}^T\mathbf{L}\boldsymbol{x}}{\boldsymbol{x}^T\boldsymbol{x}}$; it corresponds to $\mathbf{P} = -\mathbf{L}$ with $\boldsymbol{c} = -\frac{\boldsymbol{d}}{2}$ and $\mathbf{Q} = \mathbf{I}$ with $\mathbf{A}' = \mathbf{0}, \gamma = 1$ in GENDS. In the other setting with $\mathbf{Q} = \mathbf{D}$, the problem will correspond to the Normalized Spectral Clustering [62], $Ncut(S, \bar{S}) = \frac{|cut(S)|}{vol(S)}$ where $vol(S) = \sum_{u \in S} \boldsymbol{d}_u$. Thus, it is equivalent to set $\mathbf{P} = -\mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2} = \mathbf{L}_{sym}$, i.e., the normalized Laplacian matrix of $\mathcal{G}$, and $\mathbf{Q} = \mathbf{I}$.

Remark 2. Charikar: The *densest subgraph detection* problem as formulated in (1) corresponds to $\mathbf{P} = \mathbf{A}$ and $\mathbf{Q} = \mathbf{I}$ ignoring the constant factor.

In a more general case, [63], [64] use $\tilde{\mathbf{A}}$ configured with different γ to explore the trade-off between density and size of the final dense subgraphs with the domain-set-based optimization method.

Remark 3. FAIRDS*:* The *fair densest subgraph detection* problem ensures each subgraph contains an equal number of representatives of the node labels via the fairness constraints [40]. Here, the node label is uncorrelated with community memberships.

Given the matrix $\mathbf{F}$ whose columns form an orthogonal basis of the subspace denoting the constraints that every label is featured equally often, let $\mathbf{A}_F = (\mathbf{I} - \mathbf{F}\mathbf{F}^\top)\mathbf{A}(\mathbf{I} - \mathbf{F}\mathbf{F}^\top)$, the number of edges of the induced subgraph by a fair subset S is $\frac{\boldsymbol{x}^\top\mathbf{A}_F\boldsymbol{x}}{2}$. Therefore, FAIRDS corresponds to $\mathbf{P} = \mathbf{A}_F$ and $\mathbf{Q} = \mathbf{I}$. If the node label is binary, $\mathbf{F}$ degenerates into a unit 2-norm vector.

Remark 4. Fraudar: The *suspicious densest group detection* problem treats the weights of nodes and edges as the suspiciousness score of nodes and edges, i.e., c_u and a_{ij} measure how individually suspicious the particular node u and edge e_{ij} are (can be determined by other information, like user profile and text of content). As (2) shows, it equals to $\mathbf{P} = \mathbf{A} + 2\mathbf{D}_{\boldsymbol{c}}$ and $\mathbf{Q} = \mathbf{I}$, where the numerator $\boldsymbol{x}^\top\mathbf{P}\boldsymbol{x}$ is the total suspiciousness of the subgraph, ignoring the constant factor.

Remark 5. SPARSECUTDS*:* The SPARSECUTDS finds a community that is densely connected internally but sparsely connected to the rest of the graph; it is optimized by maximizing the density while minimizing the average cut size [20]. With the formulation of the cut size ($|cut(S)|$) in remark 1, the objective to be maximized by SPARSECUTDS is denoted as

$$g_\alpha(S) = \frac{|E(S)| - \alpha \cdot |cut(S)|}{|S|} = \frac{\boldsymbol{x}^\top\left((\frac{1}{2} + \alpha)\mathbf{A} - \alpha\mathbf{D}\right)\boldsymbol{x}}{\boldsymbol{x}^\top\boldsymbol{x}}$$

$$= c \cdot \frac{\boldsymbol{x}^\top\left(\mathbf{A} - \frac{2\alpha}{2\alpha+1}\mathbf{D}\right)\boldsymbol{x}}{\boldsymbol{x}^\top\boldsymbol{x}},$$

where $\alpha \in [0, \infty)$ controls the significance of the $|cut(S)|$ term and $c = \frac{1}{2} + \alpha$ is a constant. Thus, it corresponds to $\mathbf{P} = \mathbf{A} + 2\mathbf{D}_{\boldsymbol{c}}$ with $\mathbf{D}_{\boldsymbol{c}} = -\frac{\alpha}{2\alpha+1}\mathbf{D}$ and $\mathbf{Q} = \mathbf{I}$.

Remark 6. TEMPDS*:* TEMPDS detects dense subgraphs with nodeset S appearing at time t suddenly while having very few edges at time $t - 1$ [2]. Let $\mathbf{A}_t$ and $\mathbf{A}_{t-1}$ be adjacency matrices of the snapshots of a temporal graph. Thus, $\boldsymbol{x}^\top\mathbf{A}_t\boldsymbol{x}$ and $\boldsymbol{x}^\top\mathbf{A}_{t-1}\boldsymbol{x}$ are twice the number of edges of the corresponding subgraphs. Taking the size of S into consideration, the objective of TEMPDS is to maximize $g(S) = \frac{\boldsymbol{x}^\top\mathbf{A}_t\boldsymbol{x}}{\boldsymbol{x}^\top(\mathbf{A}_{t-1}+2\mathbf{I})\boldsymbol{x}} = \frac{\boldsymbol{x}^\top\mathbf{A}_t\boldsymbol{x}}{\boldsymbol{x}^\top\tilde{\mathbf{A}}_{t-1}\boldsymbol{x}}$, i.e., $\gamma = 2$.

Remark 7. Risk-averse DS: Given a graph $\mathcal{G}$, the positive entry a_{ij} of its adjacency matrix $\mathbf{A}$ represents the expected *reward* of the edge e_{ij} and the negative entry is opposite to the *risk* of the edge, the absolute value $|a_{ij}|$ measures the strength. Then, $\mathbf{A}$ can be written into $\mathbf{A} = \mathbf{A}^+ - \mathbf{A}^-$, where $\mathbf{A}^+$ is the *reward network* and composed of all positive edges in $\mathbf{A}$, that is, its entry $\mathbf{A}^+_{i,j} = \max(a_{ij}, 0)$; and $\mathbf{A}^-$ is the *opposition risk network* and its entry $\mathbf{A}^-_{i,j} = |\min(a_{ij}, 0)|$.

The Risk-averse dense subgraph detection problem finds a subgraph that has a large positive average degree while a small negative average degree [27]. It is formulated in the GenDS by $\mathbf{P} = \mathbf{A}^+ + 2\mathbf{D}_{\boldsymbol{c}}$ with $\boldsymbol{c} = \frac{\gamma_1}{2}\mathbf{1}$ and $\mathbf{Q} = \mathbf{A}^- + \gamma_2\mathbf{I}$, where $\gamma_1, \gamma_2 \geq 0$ control the size of the subgraph considering the contribution of the size of S.

As for the densest subgraph detection for a bipartite graph $\hat{\mathcal{G}}$, it can be reduced to the GenDS framework by converting $\hat{\mathcal{G}}$ to be a monopartite graph as follows.

Lemma 1: Given a bipartite graph $\hat{\mathcal{G}} = (L \cup R, E)$ with the left-side nodeset L and right-side nodeset R, $L \cap R = \emptyset$ and $|L| + |R| = n$, the densest bipartite subgraph detection problem over $\hat{\mathcal{G}}$ corresponds to the setting that $\boldsymbol{x} = [\boldsymbol{y}; \boldsymbol{z}] \in \mathbb{R}^n$ concatenating $\boldsymbol{y} \in [0,1]^{|L|}$ and $\boldsymbol{z} \in [0,1]^{|R|}$, and $\mathbf{P} \in \mathbb{R}^{n \times n}$, $\mathbf{Q} \in \mathbb{R}^{n \times n}$,

$$\begin{aligned}\mathbf{P} &= \begin{bmatrix} \mathbf{D}_{\boldsymbol{c}_L} & \frac{\mathbf{A}}{2} \\ \frac{\mathbf{A}^T}{2} & \mathbf{D}_{\boldsymbol{c}_R} \end{bmatrix} \\ &= anti\text{−}diag([\mathbf{A}, \mathbf{A}^T]/2) + diag([\mathbf{D}_{\boldsymbol{c}_L}, \mathbf{D}_{\boldsymbol{c}_R}]) \\ \mathbf{Q} &= diag([\mathbf{I}_{|L|\times|L|}, \mathbf{I}_{|R|\times|R|}])\end{aligned}$$

where $\boldsymbol{c}_L$ and $\boldsymbol{c}_R$ are the node weight vectors for L and R, respectively. $\mathbf{A} \in \mathbb{R}^{|L|\times|R|}$ is the adjacency matrix corresponding to $\hat{\mathcal{G}}$, and $\mathbf{I}$ is the identity matrix with subscripts to denote its shape.

There is a line of works that use different density metrics [7], [59] for dense subgraph detection of a bipartite graph $\hat{\mathcal{G}}$, i.e., $d(S,T) = \frac{|E(S,T)|}{\sqrt{|S|}\sqrt{|T|}}$ is the density of the subgraph induced by $S \cup T$, where $S \subseteq L$ and $T \subseteq R$; $|E(S,T)|$ is the edgeset of the induced subgraph. The relative merits of $d(S,T)$ and $g(S)$ as objective functions for density were discussed in [7], [26], [59].

Considering the metric $g(S)$, in this paper, we formalize a unified framework for our problem and reveal the relationship between the densest subgraph and the graph spectral in Theorem 4. We know of no similar results to ours.

To avoid some trivial solution for the weighted graph, e.g., the single edge with a heavy weight, we can introduce column weights as $\mathbf{A} \cdot diag(\frac{1}{h(\mathbf{1}^\top\mathbf{A})})$ for some function h, e.g., $h(x) = x^\alpha$ with $\alpha \in \mathbb{R}^+$ or $h(x) = \log(x + c)$ with a small constant c to prevent the zero denominators, as used in [45]. Besides, we can use motif-based high-order graphs [58] to recognize more complex dense patterns.

IV. Flow-Based Exact Solution and Analysis

In this section, we demonstrate through analysis the hardness of our problem and solve instances of an approximate GenDS problem, by mapping to the Min-Cut problem. The rationale behind this idea is similar to [6].

Given a positive value $\beta \in \mathbb{R}^+$ as a guess, we determine whether there is a subset S such that $\max_{S \subseteq V} g(S; \mathcal{G}, \mathcal{G}') \geq \beta$ for the GenDS problem.

We construct an edge-weighted directed network $\mathcal{N} = (V_N, E_N)$ as follows. The nodeset $V_N = \{s, t\} \cup V$ with $s, t \notin V$ and the edgeset $E_N = E_1 \cup E_2 \cup E_3$ where E_1 is the edges from s to V, i.e., $E_1 = \{(s,u) \mid u \in V\}$; E_2 denotes replacing each undirected edge of $E^P \cup E^Q$ by two directed edges, i.e., $E_2 = \{(u,v), (v,u) \mid (u,v) \in E^P \cup E^Q\}$; and E_3 is the edges from V to t, i.e., $E_3 = \{(v,t) \mid v \in V\}$. The edge weight w for different edgeset is given as

$$w = \begin{cases} d_u^P + 2c_u & \text{if } e = (s,u) \in E_1, \\ w_e^P - \beta w_e^Q & \text{if } e \in E_2, \\ \gamma\beta + \beta d_v^Q & \text{if } e = (v,t) \in E_3, \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

where d_u^P (d_v^Q) is the degree of node u (v) of $\mathcal{G}$ ($\mathcal{G}'$), w_e^P (w_e^Q) is the weight of the edge e of $\mathcal{G}$ ($\mathcal{G}'$). And $w_e = 1$ for all edge e of an unweighted graph.

In the graph theory, we know that the Min-Cut problem for an undirected, (non-negative) weighted graph can be exactly solved in polynomial time using various algorithms, including the Stoer-Wagner algorithm, Gomory-Hu algorithm, and Karger's algorithm, etc., while it becomes an NP-Complete problem by a trivial transformation from the maximum-cut problem [65], [66] when there are negative-weight edges in the graph, except in some special cases that are polynomial-solvable [67]. In the above network $\mathcal{N}$ we constructed, E_2 may contain some negative-weight edges in (4), resulting from those edges in $E^Q \backslash E^P$ or some large values of β. Thus, our problem is generally NP-hard to find the exact solution.

In an approximate way, we can modify the network $\mathcal{N}$ to construct the edge set E_2 by removing the edge $e \in E^Q \backslash E^P$ and making the guess $\beta \leq \bar{w}$ where $\bar{w} = \min\{\frac{w_e^P}{w_e^Q} \mid e \in E^P \cap E^Q\}$. Then, we get the following polynomial solvable s-t min-cut problem.

Let $W_{E^P} = \sum_{e \in E^P} w_e^P$ (similar for W_{E^Q}) and $C = \sum_{v \in V} c_v$, which are constants for a given network. Given the node subset S, let $W_{E_S^P} = \sum_{e \in E^P \cap S \times S} w_e^P$ (similar for $W_{E_S^Q}$) and $C_S = \sum_{v \in S} c_v$. Thus, the total weight of the induced subgraph by S with the characteristic vector $\boldsymbol{x}$ is $W_{E_S^P} = \frac{\boldsymbol{x}^\top\mathbf{A}\boldsymbol{x}}{2}$, and the total weight of nodes in S is C_S.

A partition of V_N determines a s-t cut by two sets $\{s\} \cup S$ and $\{t\} \cup V \backslash S$. If $|S| = 0$, then the capacity of the cut is $c(s,t) = \sum_{u \in V} d_u^P + 2c_u = 2(W_{E^P} + C) = \hat{C}$; otherwise

$$\begin{aligned} c(s,t) = & \sum_{u \in \{s\} \cup S,\, v \in \{t\} \cup V \backslash S} w_{(u,v)} \\ = & \sum_{e=(s,u),\, u \in V \backslash S} w_e + \sum_{e \in cut(S) \subseteq E_2} w_e + \sum_{e=(v,t),\, v \in S} w_e \\ = & \sum_{u \in V \backslash S} (d_u^P + 2c_u) + \sum_{e \in cut(S) \subseteq E_2} (w_e^P - \beta w_e^Q) \\ & + \sum_{v \in S} (\gamma\beta + \beta d_v^Q) \\ & = 2(W_{E^P} + C) - 2W_{E_S^P} - 2C_S + 2\beta W_{E_S^Q} + \gamma\beta|S| \\ & = \hat{C} + (2W_{E_S^Q} + \gamma|S|) \cdot \left(\beta - \frac{2W_{E_S^P} + 2C_S}{2W_{E_S^Q} + \gamma|S|}\right) \\ & = \hat{C} + (2W_{E_S^Q} + \gamma|S|) \cdot (\beta - g(S; \mathcal{G}, \mathcal{G}')), \end{aligned}$$

where $2W_{E^Q_S} + \gamma|S| > 0$ (even only consider $E^Q \cap E^P$ for $W_{E^Q_S}$). The following theorem helps to determine whether β is too large or too small based on the conclusion in [6].

Theorem 2 (Parametric Min-Cut Optimality[2]): Given the subset S, assume that the partition $\{s\} \cup S$ and $\{t\} \cup V \backslash S$ gives the minimum capacity cut for the graph. If $|S| \neq 0$ then $\beta \leq g_{opt}$; if $|S| = 0$ then $\beta \geq g_{opt}$.

Therefore, we can use the binary search for the guess $\beta \in (0, \bar{w}]$ to find the subset S with the minimum capacity [6], [11]. Note that this is just a constrained variant of the solution for an approximate version of our GENDS problem.

V. Spectral-Based Theoretical Analysis

Here, we connect the optimization of GENDS to the graph spectral theory and generalized eigenvalue problem, showing that we can efficiently approximate the solution with the skewness properties of the spectrum of the real-world graphs, thus guiding the design of our algorithm.

A. GENDS Under Graph Spectral Theory

Derived from the theoretical analysis of the problem hardness for exact solutions in Section IV, we construct an approximation problem for the GENDS, which permits a polynomial-solvable solution based on the maximum flow algorithms or a fast approximate solution by our proposed method.

Given the graphs $\mathcal{G}$ and $\mathcal{G}'$, we can construct a *positive residual* graph $\mathcal{G}_r = (V, E_r)$ with the edgeset $E_r = \{(u,v) \,|\, (u,v) \in E^P \wedge (u,v) \notin E^Q\}$ and the edge-weight is $w_e = w_e^P - \beta w_e^Q$ for $\forall e \in E_r$, where the user-defined parameter $0 < \beta < \bar{w}$ controls the contribution of the contrast graph. The adjacency matrix of $\mathcal{G}_r$ is denoted as $\mathbf{A}_r = (\mathbf{P} - \beta\mathbf{Q})^+$, which only keeps the positive entry in the matrix. In particular, $\beta = 1$ for unweighted graphs.

Hence, the densest subgraph detection for $\mathcal{G}_r$ finds the induced subgraph that maximizes the density in $\mathcal{G}$ while minimizing that in $\mathcal{G}'$, approximating our original problem 1. Thus, the objective function in (3) is approximately formulated as

$$S^* \approx \text{arg max}_{\boldsymbol{x} \in [0,1]^n, \|\boldsymbol{x}\| \geq 1} \frac{\boldsymbol{x}^\top (\mathbf{P} - \mathbf{Q})^+ \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}$$
$$= \text{arg max}_{\boldsymbol{x} \in [0,1]^n, \|\boldsymbol{x}\| \geq 1} \frac{\boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}. \tag{5}$$

Being non-negative edge weights for $\mathcal{G}_r$, we can formulate (5) as a standard MIN-CUT problem and apply those algorithms mentioned in Section IV for the exact solution. However, its complexity is still prohibitively costly, especially for large graphs. In the rest of this section, we will connect the densest subgraph detection problem with graph spectral theory and aim to find a fast approximation algorithm by using the spectral properties of real-world graphs.

Consider the optimization problem with a similar form to (5) defined over the real vector space (i.e., $\boldsymbol{x} \in \mathbb{R}^n \setminus \mathbf{0}$), it is formalized in the *Rayleigh quotient* format as

$$R(\mathbf{A}_r, \boldsymbol{x}) = \frac{\boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}, \ \boldsymbol{x} \in \mathbb{R}^n, \ \boldsymbol{x} \neq \mathbf{0}, \tag{6}$$

where $\mathbf{A}_r \in \mathbb{R}^{n \times n}$ is a symmetric matrix. It holds that $R(\mathbf{A}_r, c \cdot \boldsymbol{x}) = R(\mathbf{A}_r, \boldsymbol{x})$ for any non-zero scalar c. The objective of (5) is a special case for a binary vector.

The following Rayleigh–Ritz Theorem [55] from the spectral theory gives the optimality solution of (6).

Theorem 3 (Rayleigh–Ritz Theorem[3]): Let $\mathbf{A}_r$ be a symmetric matrix with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_n$ and corresponding eigenvectors $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$, i.e., $\mathbf{A}_r = \sum_i \lambda_i \boldsymbol{u}_i^T \boldsymbol{u}_i$. Then

$$\lambda_1 = \max_{\boldsymbol{x} \neq \mathbf{0}} R(\mathbf{A}_r, \boldsymbol{x}) = \max_{\boldsymbol{x} \in \mathbb{R}^n, \|\boldsymbol{x}\|=1} \boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{x} \Longrightarrow \boldsymbol{x} = \boldsymbol{u}_1$$
$$\lambda_n = \min_{\boldsymbol{x} \neq \mathbf{0}} R(\mathbf{A}_r, \boldsymbol{x}) = \min_{\boldsymbol{x} \in \mathbb{R}^n, \|\boldsymbol{x}\|=1} \boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{x} \Longrightarrow \boldsymbol{x} = \boldsymbol{u}_n.$$

In general, for $1 \leq k \leq n$, let $\mathcal{S}_k = \text{span}(\mathbf{U})$ as the span of column vectors of the matrix $\mathbf{U}$ w.r.t. top-k eigenvalues, i.e., $\mathcal{S}_k = \{\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k\}$, and set $\mathcal{S}_0 = \mathbf{0}$. Then

$$\lambda_k = \max_{\boldsymbol{x} \neq \mathbf{0}, \, \boldsymbol{x} \perp \mathcal{S}_{k-1}} R(\mathbf{A}_r, \boldsymbol{x}) = \max_{\|\boldsymbol{x}\|=1, \, \boldsymbol{x} \perp \mathcal{S}_{k-1}} \boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{x}$$
$$\Longrightarrow \boldsymbol{x} = \boldsymbol{u}_k.$$

Therefore, λ_k is the largest value of $R(\mathbf{A}_r, \boldsymbol{x})$ over the complement space of $\mathcal{S}_{k-1}$.

In analogy to eigenvalues, the singular values of a matrix achieve an optimality property that resembles that of Rayleigh quotient matrices [68]. To avoid negative eigenvalues of a large magnitude for real graphs [69], we instead use singular values and singular vectors in the following.

Let $\mathbf{A}_r = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_{i=1}^r \sigma_i \boldsymbol{u}_i \boldsymbol{v}_i^T$ be the singular value decomposition of $\mathbf{A}_r$, the columns of $\mathbf{U}$ and $\mathbf{V}$ are the left- and right-singular vectors respectively, i.e., $\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_r]$ and $\mathbf{V} = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r]$, $\mathbf{\Sigma} = diag([\sigma_1, \ldots, \sigma_r])$ for the singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$. We also have the following formulation in regard to the GENDS problem,

Lemma 2: The optimal solution for the GENDS in (3) can be written as

$$S^* = \text{arg max}_{\boldsymbol{x} \in [0,1]^n, \|\boldsymbol{x}\| \geq 1} \frac{\boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}$$
$$= \text{arg max}_{S \subseteq V, |S| \geq 1} \frac{1}{|S|} \sum_{i=1}^n \sigma_i \left(\sum_{j \in S} \boldsymbol{u}_{ij} \right) \left(\sum_{j \in S} \boldsymbol{v}_{ij} \right) \tag{7}$$

where $\boldsymbol{u}_{ij}$ and $\boldsymbol{v}_{ij}$ denote the j-th element of $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$ corresponding to σ_i, respectively. It is clear that the optimal density value $g_{\text{opt}} \leq \sigma_1$.

For the bipartite graph, we define the related quadratic optimization problem for an asymmetric $\mathbf{A}_r \in \mathbb{R}^{m \times n}$ as

$$R(\mathbf{A}_r; \boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{y}}{\boldsymbol{x}^\top \boldsymbol{x} + \boldsymbol{y}^\top \boldsymbol{y}},$$
$$\text{s.t. } \boldsymbol{x} \in \mathbb{R}^m, \boldsymbol{y} \in \mathbb{R}^n; \ \boldsymbol{x} \neq \mathbf{0}, \ \boldsymbol{y} \neq \mathbf{0}.$$

[2] Refer to [6] for the detailed proof.

[3] The proof details of the theorem refer to [55].

We also obtain the following theorem that leads to a similar statement as Theorem 3, which avoids constructing the big matrix of $\mathbb{R}^{(m+n)\times(m+n)}$ for the bipartite graph.

Theorem 4 (Bigraph Spectral): Suppose $\mathbf{A}_r$ is a $m \times n$ matrix, and its singular value decomposition is $\mathbf{A}_r = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. For any vector $\boldsymbol{x} \in \mathbb{R}^m$ and $\boldsymbol{y} \in \mathbb{R}^n$

$$\sigma_1 = \max_{\|\boldsymbol{x}\|=\|\boldsymbol{y}\|=1} \boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{y} \geq \max_{\boldsymbol{x}\neq\mathbf{0},\boldsymbol{y}\neq\mathbf{0}} 2 \cdot R(\mathbf{A}_r; \boldsymbol{x}, \boldsymbol{y})$$

$$\Rightarrow \boldsymbol{x} = \boldsymbol{u}_1, \ \ \boldsymbol{y} = \boldsymbol{v}_1.$$

In general, for $1 \leq k \leq r$, let $\mathcal{S}_k^U = \text{span}(\mathbf{U})$ and $\mathcal{S}_k^V = \text{span}(\mathbf{V})$ denote the span of column vectors of $\mathbf{U}$ and $\mathbf{V}$, respectively. Let $\mathcal{S}_0^U = \mathbf{0}$ and $\mathcal{S}_0^V = \mathbf{0}$. Then

$$\begin{aligned}\sigma_k = & \max_{\substack{\|\boldsymbol{x}\|=\|\boldsymbol{y}\|=1 \\ \boldsymbol{x}\perp\mathcal{S}_{k-1}^U,\, \boldsymbol{y}\perp\mathcal{S}_{k-1}^V}} \boldsymbol{x}^\top \mathbf{A}_r \boldsymbol{y} \\ \geq & \max_{\substack{\boldsymbol{x}\neq\mathbf{0},\, \boldsymbol{y}\neq\mathbf{0} \\ \boldsymbol{x}\perp\mathcal{S}_{k-1}^U,\, \boldsymbol{y}\perp\mathcal{S}_{k-1}^V}} 2 \cdot R(\mathbf{A}_r, \boldsymbol{x}, \boldsymbol{y}) \Rightarrow \boldsymbol{x} = \boldsymbol{u}_k, \ \ \boldsymbol{y} = \boldsymbol{v}_k.\end{aligned}$$

Therefore, given a bipartite graph $\hat{\mathcal{G}} = (L \cup R, E)$ with the adjacency matrix $\mathbf{A} \in \mathbb{R}^{|L|\times|R|}$, we will have similar properties to Lemma 2 as

Lemma 3: For the densest bipartite subgraph detection in Fraudar with $\mathbf{P} = diag([\frac{\mathbf{A}}{2}, \frac{\mathbf{A}^\top}{2}])$ and $\boldsymbol{x}^\top \mathbf{P} \boldsymbol{x} = |E(S)|$, the optimal solution can be written as $S^* = \arg\max_{\boldsymbol{x}\in[0,1]^n, |\boldsymbol{x}|\geq 1} \frac{\boldsymbol{x}^\top \mathbf{P} \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}}$; and the optimal g_{opt} is

$$\begin{aligned}& \max_{\boldsymbol{x}\in[0,1]^n, |\boldsymbol{x}|\geq 1} \frac{\boldsymbol{x}^\top \mathbf{P} \boldsymbol{x}}{\boldsymbol{x}^\top \boldsymbol{x}} = \max_{\substack{\boldsymbol{y}\in[0,1]^{|L|},\, \boldsymbol{z}\in[0,1]^{|R|} \\ |\boldsymbol{y}|>0,\, |\boldsymbol{z}|>0}} R(\mathbf{A}_r, \boldsymbol{y}, \boldsymbol{z}) \\ & \leq \max_{\substack{S=\delta(\boldsymbol{y})\cup\delta(\boldsymbol{z}) \\ |S|\geq 1}} \frac{1}{|S|} \sum_{i=1} \sigma_i \left(\sum_{j\in\delta(\boldsymbol{y})} \boldsymbol{u}_{ij} \right) \left(\sum_{j\in\delta(\boldsymbol{z})} \boldsymbol{v}_{ij} \right), \quad (8)\end{aligned}$$

where $\boldsymbol{u}_{ij}, \boldsymbol{v}_{ij}$ denote the j-th element of $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$. It also holds that the optimal density $g_{\text{opt}} \leq \sigma_1$.

B. GenDS Under Generalized Eigenvalue Problem

For the GenDS problem, $\mathbf{P}$ and $\mathbf{Q}$ are symmetric and have a real spectrum as undirected graphs. It always holds that $\boldsymbol{x}^\top \mathbf{Q} \boldsymbol{x} > 0$ for $\boldsymbol{x} \in [0,1]^n \setminus \mathbf{0}$, since it counts the number of edges and the size of the induced subgraph in $\mathcal{G}'$. Assuming that $\mathbf{Q}$ is positive definite, i.e., $\boldsymbol{x}\mathbf{Q}\boldsymbol{x}^\top > 0$ for any $\boldsymbol{x} \neq \mathbf{0}$; with the relaxation $\boldsymbol{x} \in \mathbb{R}^n$, the objective function in (3) is equivalent to the *generalized Rayleigh quotient* problem [55], [57], [70], i.e., $R(\mathbf{P}, \mathbf{Q}; \boldsymbol{x}) := \frac{\boldsymbol{x}^\top \mathbf{P} \boldsymbol{x}}{\boldsymbol{x}^\top \mathbf{Q} \boldsymbol{x}}$. Thus, according to the Rayleigh-Ritz quotient method [71], it can be restated as

$$\underset{\boldsymbol{x}}{\text{maximize}} \ \ \boldsymbol{x}^\top \mathbf{P} \boldsymbol{x}, \quad \text{subject to } \boldsymbol{x}^\top \mathbf{Q} \boldsymbol{x} = 1. \quad (9)$$

The Lagrangian [72] for the above (9) is

$$\mathcal{L} = \boldsymbol{x}^\top \mathbf{P} \boldsymbol{x} - \lambda(\boldsymbol{x}^\top \mathbf{Q} \boldsymbol{x} - 1),$$

where λ is the Lagrange multiplier. Equating the derivative of $\mathcal{L}$ to zero gives

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{x}} = 2\mathbf{P}\boldsymbol{x} - 2\lambda\mathbf{Q}\boldsymbol{x} \overset{\text{set}}{=} \mathbf{0} \implies \mathbf{P}\boldsymbol{x} = \lambda\mathbf{Q}\boldsymbol{x}, \quad (10)$$

which is a *generalized eigenvalue problem (GEP)* denoted by the pair $(\mathbf{P}, \mathbf{Q})$, then $\boldsymbol{x}$ corresponds to the eigenvector, and λ is the eigenvalue. So, the eigenpair (Φ, Λ) corresponds to a set of generalized eigenvectors and eigenvalues of $(\mathbf{P}, \mathbf{Q})$.

Assume that there is some $\mathbf{Q}$ that satisfies the condition. The eigenvector is the one having the largest eigenvalue that achieves the goal of the *maximization* of (9).

Given the leading eigenpairs (Φ, Λ) with the eigenvalues in decreasing order, our objective function in GenDS has local bounds as Lemma 2 and Lemma 3 over the discrete vector space (i.e., $\boldsymbol{x} \in [0,1]^n$); that is, the i-th optimal value $g_{opt}^i \leq \lambda_i$ based on the eigenvector ϕ_i and the final optimal density $g_{opt} = \max_i g_{opt}^i \leq \lambda_1$, where ϕ_i is the i-th column of Φ and λ_i is the i-th element of Λ.

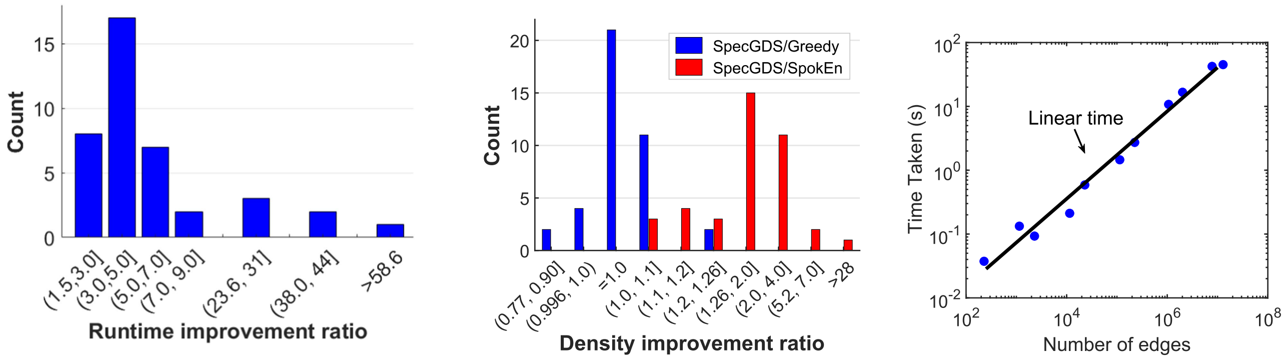
For the eigenvalues $\lambda_i^{\mathbf{A}'}$ of the adjacency matrix $\mathbf{A}'$ with $1 \leq i \leq n$, it is known that $\sum_i^n \lambda_i^{\mathbf{A}'} = \mathbf{tr}(\mathbf{A}') = 0$. Thus, maximizing (3) will be problematic due to the singularity of $\mathbf{Q}$ in some real scenarios. However, we can construct $\mathbf{Q}$ to be *strictly diagonally dominant* and *positive-definite* by introducing a large value of γ [73], i.e., $\mathbf{Q}_{ii} > \sum_{j,j\neq i} |\mathbf{Q}_{ij}|$, which gives higher priority to the size of the subgraph in $\mathcal{G}'$, for example, the signless Laplacian (i.e., $\mathbf{Q} = \mathbf{A} + \mathbf{D}$) for connected non-bipartite graphs. To adapt to more general situations, we usually take the residual graph $\mathcal{G}_r$ as an approximation format, as described in Section V-A.

C. Skewness Properties of Real-World Graphs.

In theory, Milena and Christos [74] proved that the largest eigenvalues of graphs, whose highest degrees are Zip-like distributed with a slope of α, are distributed according to a power law with a slope of $\alpha/2$. Therefore, the spectral-formulated densest subgraph detection problem benefits from the power-law node degrees of a graph concerning the above conclusion, since the majority of the contribution to the objective value comes from those w.r.t. the leading eigenvalues. Moreover, many graph generators create more realistic graphs that match these patterns, including the power-law degree and eigenvalue distributions, including STORM [75], R-MAT [76], Kronecker graph [77], etc.

In addition, sparsity and various power laws are indeed the key components of the real-world networks gathered from the world-wide-web, social networks, E-commerce, online reviews, recommendation systems, and more. Those primary properties contribute to time- and space-efficient computing or storage and synthetically modeling realistic networks. Various studies [77], [78] have shown that most real-world graphs have a statistically significant power law distribution with degree distribution, the distribution of "bipartite cores" ($\approx$ communities), a cutoff in the eigenvalue or singular values of the adjacency matrix, and the Laplacian matrix, etc. Also, the distribution of eigenvector elements (as indicators of "network value") associated with the leading eigenvalues of the graph adjacency matrix is skewed [79].

Taking the `soc-twitter` network as an example, which is the largest network with 1.47 billion edges used in experiments, Fig. 2(a)–(b) shows the above properties. The distributions of its

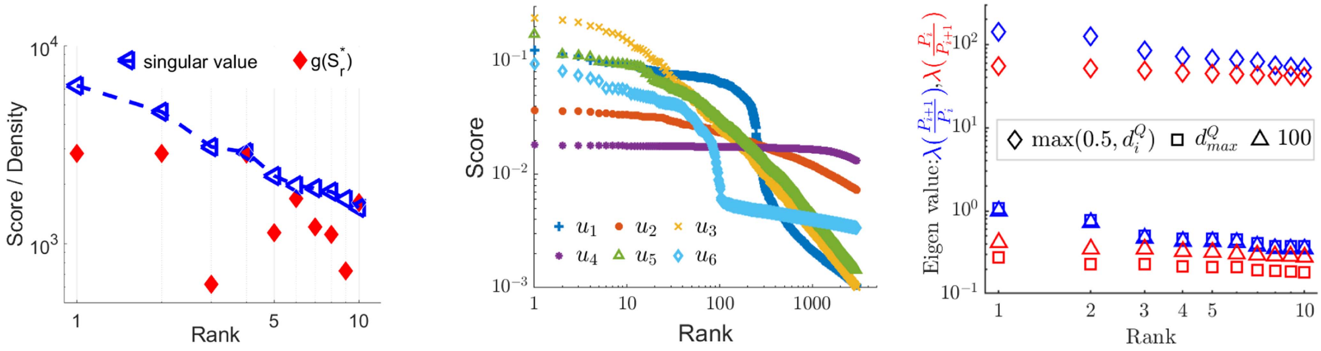


(a) Speedup statistics for the densest subgraph detection (SPECGDS vs. GREEDY).

(b) Comparison for the density quality of the optimal densest subgraph.

(c) The linear scalability of SPECGDS w.r.t. the number of edges in a graph.

Fig. 1. **The proposed algorithm SPECGDS is fast, effective, and linear-scalable.** (a) Our proposed method detects the densest subgraphs (qualities in Fig. 1b) up to 58.6× faster than the widely-used GREEDY algorithm for various real-world datasets (40 in total). (b) SPECGDS has better or comparable density quality compared with GREEDY and SPOKEN algorithm in the densest subgraph detection. It consistently outperforms SPOKEN for all graphs and finds up to 28× denser subgraph; it obtains the same or denser (more than 1.26×) optimal density for most graphs compared with GREEDY, and 4 graphs with very close densities ($\geq 0.996\times$) and only 2 graphs with less than 0.9 density improvement. (c) The time taken of SPECGDS grows linearly with the size of the graph.



(a) The distribution of the leading singular values of $\mathbf{A}$ and the optimal density $g(S_r^*)$ corresponding to the singular vector $\boldsymbol{u}_r$.

(b) The distribution of the elements in top singular vectors $\boldsymbol{u}_r$ of the adjacency matrix.

(c) The distribution of the top ten eigenvalues λ from the generalized eigenvalue decomposition of two matrices.

Fig. 2. **The running example and properties of the real-world networks.** (a) shows the power-law distribution phenomenon of the top-ten singular values σ_i of the adjacency matrix $\mathbf{A}$ and the optimal density $g(S_r^*)$ detected based on the corresponding singular vectors $\boldsymbol{u}_i$ by SPECGDS. (b) shows the skewed distributions of the leading 3,000 elements of the top six singular vectors from `soc-twitter` network. (c) shows the similar power-law distribution of the top ten generalized eigenvalues of subgraphs (2006 and 2007) from the `DBLP co-authorship` network ($i = 2006$), under different construction strategies for γ in $\mathbf{Q}_{ii} + \gamma$ as Section V-B, i.e., $\max(0.5, \boldsymbol{d}_i^Q)$, $\boldsymbol{d}_{max}^Q$, and 100; $\boldsymbol{d}_{max}^Q$ is the maximal node degree in $\mathcal{G}'$.

top singular values and the elements of the large magnitude of the corresponding singular vectors follow the power law or skewed distribution. The distribution of eigenvalues and elements of eigenvectors w.r.t. the GEP also have similar properties, as Fig. 2(c) shows.

Therefore, with the spectral formulation of GENDS, the skewness, coming from singular values and components in singular vectors of the real-world graphs, guarantees that we can only consider the leading singular vectors and use a few elements of a large magnitude in them to efficiently construct the candidates for the dense subgraphs detection and obtain the optimal. We will introduce it in more detail in the next.

VI. ALGORITHMS & COMPLEXITY ANALYSIS

In this section, we present our proposed methods SPECGDS and GEPGDS for the unified GENDS problem, and provide analysis for their properties.

A. *Preliminary: GREEDY Algorithm*

We first review the related Charikar's peeling algorithm as GREEDY 1 with edge density as the metric g as in (1). It takes into the entire original graph, greedily removes the node with the smallest degree from the graph (in Line 3), and returns the densest one among the shrinking sequence of subgraphs during the procedure. It is guaranteed to obtain a solution of at least half of the optimum density, i.e., $g^* \geq \frac{1}{2} g_{opt}$. By utilizing the *priority tree* to manage the nodes during the peeling process, the complexity is $O(|E| \log |V|)$.

Moreover, though there is an efficient implementation for GREEDY, which results in $O(|E| + |V|)$ time complexity [7], it only suits the unweighted graph, and the linear time implementation does not carry over since it depends on the fact that vertex degrees are integers bounded by $|V|$. The GREEDY algorithm can be implemented using Fibonacci heaps to determine the minimum degree vertex in Line 3 and achieve $O(|E| + |V| \log |V|)$

Algorithm 1: GREEDY: Densest Subgraph Detection.

```
Input: Undirected graph $\mathcal{G}$, density metric $g$.
Output: Nodeset $S^*$ w.r.t. the densest subgraph in $\mathcal{G}$.
1 $S^* \leftarrow S$                          ▷ $S$ is the nodeset of $\mathcal{G}$
2 while $S \neq \emptyset$ do
      ▷ find the vertex to maximize the metric
3     $\hat{u} \leftarrow \arg\max_{u \in S} g(S/\{u\})$
      ▷ $S/\{u\}$: the remaining nodeset without vertex $u$
4     Remove $\hat{u}$ and all its adjacent edges from $\mathcal{G}$
5     if $g(S) > g(S^*)$ then
6         $S^* \leftarrow S$
7 return $S^*$.
```

time for the weighted graph. However, many analysis reports and empirical results have demented the inferior performance of the Fibonacci heap compared with other similar data structures in real applications. For instance, compared to binary heaps, Fibonacci heaps are more complicated when it comes to coding them. They are not as efficient in practice compared with the theoretically less efficient forms of heaps since, in their simplest version, they require storage and manipulation of four pointers per node, compared to the two or three pointers per node needed for other structures [80]. Hence, we adopt the priority tree as the efficient implementation of GREEDY, which has been widely used [45], [60], [81], to fit different cases, including the weighted graphs and the introduced column weights.

However, it is worth mentioning that the densest subgraphs are usually much smaller and embedded in a large graph (as background). Thus, searching from scratch results in many inefficient searches and update steps to find an approximate solution or even candidates for GREEDY.

B. Implications of Theoretical Analysis

Lemma 2–3 show the upper bound of the optimal density, i.e., $g_{opt} \leq \sigma_1$, and σ_k is the optimal value for the real space orthogonal to $\mathbb{S}_{k-1}$ $(k > 1)$ as Theorem 3-4. The formulation of S^* highlights that the real-value singular vectors provide insight into finding the optimal densest subgraph, that is, these nodes in S^* will have greater importance in the singular vectors associated with the leading singular values.

Considering the skewed distribution of the magnitude of the elements in a singular vector, we will construct some small-size nodeset candidates, from which we can derive some subgraphs, with the nodes having a large magnitude value in the leading singular vectors to avoid searching from scratch, that is, $S_C = \{S_1, \ldots, S_k; 1 \leq k \ll n\}$ with the i-th candidate $S_i = \{j; \boldsymbol{u}_{ij} > \Delta_L, j \in [|L|]\} \cup \{j; \boldsymbol{v}_{ij} > \Delta_R, j \in [|R|]\}$ for the singular vectors $\boldsymbol{u}_i$ and $\boldsymbol{v}_i$, Δ_L and Δ_R are some pre-defined truncation thresholds; the optimal density for $\mathcal{G}(S_i)$ is $g_i \leq \sigma_i$. Here, we determine the selection thresholds as $\Delta_L = 1/\sqrt{|L|}$ and $\Delta_R = 1/\sqrt{|R|}$ [4] based on the re-formulation of the optimal solution in (7)-(8).

C. Proposed Algorithm: SPECGDS

Therefore, we propose SPECGDS, which utilizes spectral graph properties and the greedy peeling strategy to solve the GENDS problem. Algorithm 2 summarizes our approach.

Given the adjacency matrix $\mathbf{A}_r$ of the residual graph $\mathcal{G}_r$, density metric g, and the top-approximation rank k, which controls the maximum size of the candidate set. SPECGDS first finds the top-k spectral decomposition of $\mathbf{A}_r$ (Line 2), then detects the possible densest subgraphs based on the leading singular vectors. In each round, it constructs the candidate subset S_r based on the truncated singular vectors $\boldsymbol{u}_r$ and $\boldsymbol{v}_r$, and uses the *greedy algorithm* to search for the densest subgraph for $\mathcal{G}_r(S_r)$ to maximize g. It checks the stop condition with the next singular value for the current optimal result, i.e., g^*_{cur}, at Line 7 for early stopping.

Specifically, the selection of $\mathcal{G}_r(S_r)$ in GREEDY depends on different instantiations of metrics at Line 5. For example,

- $\hat{u} = \arg\min_{u \in \mathcal{G}_r} d_{\mathcal{G}_r}(u)$ for Charikar
- $\hat{u} = \arg\min_{u \in \mathcal{G}_r} d_{\mathcal{G}_r}(u) + c_u$ for Fraudar
- $\hat{u} = \arg\min_{u \in \mathcal{G}_r} \frac{1}{2} \cdot d_{\mathcal{G}_r}(v) - \alpha \cdot d_{V \setminus \{\mathcal{G}_r \setminus \{u\}\}}(u)$ for SPARSECUTDS

where $d_{\mathcal{G}_r}(u)$ is the degree of the node u of $\mathcal{G}_r$ and $d_{V \setminus \{\mathcal{G}_r \setminus \{u\}\}}(u)$ is the number of edges from u the other part $V \setminus \mathcal{G}_r$ of the original nodeset.

How many subgraph candidates do we need to check, i.e., the lower bound of k? Using some off-the-shelf detection methods, e.g., GREEDY, let g^*_{cur} be the current detected optimal density based on the singular vectors. If there exists some $r \in (1, k]$ satisfying $g^*_{cur} \geq \sigma_r$, the optimal density g_{opt} can be determined to g^*_{cur}, due to the decreasing order of singular values $(\sigma_i \geq \sigma_{i+1})$ and the upper-bound $(g_i \leq \sigma_i)$. Finally, the subgraph concerning g_{opt} is returned as a result.

Furthermore, the power-law distribution nature of the singular values of real-world graphs and the theoretical bounds of solutions from detection algorithms (the exact or $\frac{1}{2}$-optimal approximate result) guarantee that the size of the candidates will be very small, as shown in Section VII.

Fig. 2(a) gives an example of the relation between the leading singular value σ_r and the optimal density $g(S^*_r)$ detected from the corresponding singular vector $\boldsymbol{u}_r$. The algorithm terminates at $r = 5$ due to the early stopping criterion, and returns $\mathcal{G}_r(S^*_2)$ as the result of $g^*_{cur} = g(S^*_2)$.

Besides the pre-computing top-k spectral decomposition strategy in Line 2, we can use other ways to get further optimization, e.g., a lazy or online way to compute the $(r+1)$-th largest spectral decomposition result with the power methods or the efficient Krylov subspace methods, such as the Lanczos method [82]. In the experiments, we adopt an incremental decomposition method that gets top-l singular values and singular vectors at first; if the stop criterion in Line 7 is not satisfied, we then get the further next s singular values and singular vectors with step-size s. The step-wise increasing decomposition will continue until reaching at most k singular vectors or the early-stopping criterion holds.

Since S_r is quite small, we can use other approaches to detect the densest subgraph in Line 5, except for GREEDY, considering

[4] If $\mathbf{A}_r$ is symmetric in Eq.(7), $|L| = |R| = n$ and $\Delta_L = \Delta_R = \frac{1}{\sqrt{n}}$.

Algorithm 2: SpecGDS: General Dense Subgraph Detection.

```
   Input: Matrix $\mathbf{P}, \mathbf{Q}$ for the graph $\mathcal{G}$ and $\mathcal{G}'$, density
          metric $g$, top-approximation rank $k$.
   Output: Nodeset $S$ of the densest subgraph in $\mathcal{G}_r$.
1  $S = \emptyset$
2  $\mathbf{A}_r = (\mathbf{P} - \beta\mathbf{Q})^+$ ▷ construct positive residual graph $\mathcal{G}_r$
3  $\mathbf{U}, \Sigma, \mathbf{V}$ = SVD($\mathbf{A}_r, k$)   ▷ top-$k$ spectral decomposition
4  for $r \leftarrow 1$ to $k$ do
       ▷ construct the candidate set $S_r$ based on $\boldsymbol{u}_r$ & $\boldsymbol{v}_r$
5      $S_r = \{i : \boldsymbol{u}_{ri} > \Delta_L, i \in [|L|]\} \cup \{j : \boldsymbol{v}_{rj} > \Delta_R, j \in [|R|]\}$
6      $S_r^* \leftarrow$ Greedy($\mathcal{G}_r(S_r)$, $g$)   ▷ greedily remove nodes
       ▷ the current optimal density $g^*_{cur} := g(S)$
7      if $g(S_r^*) > g(S)$ then
8          $S \leftarrow S_r^*$
       ▷ spectral early-stopping criterion
9      if $r < k$ & $g(S) > \sigma_{r+1}$ then break ;
10 return $S$.
```

Algorithm 3: GepGDS: General Dense Subgraph Detection With Generalized Eigenvalues.

```
   Input: Matrix $\mathbf{P}, \mathbf{Q}$ for the graph $\mathcal{G}$ and $\mathcal{G}'$, density
          metric $g$, top-approximation rank $k$.
   Output: Nodeset $S$ w.r.t. the GenDS.
1  $S = \emptyset$
   ▷ top-$k$ GEVD for the matrix pair $(\mathbf{P}, \mathbf{Q})$
2  $[\Phi, \Lambda]$ = Eigs($\mathbf{P}, \mathbf{Q}, k$)
3  for $r \leftarrow 1$ to $k$ do
       ▷ $S_r$ is a candidate set for the target subgraph
4      $S_r = \{v : \Phi_r(v) \geq \Delta\}$
5      while $\exists\, v \in S_r$, $g(S_r) < g(S_r \setminus \{v\})$ do
           ▷ remove candidate to improve the objective func.
6          $S_r = S_r \setminus \{v\}$
7      if $g(S_r) > g(S)$ then
8          $S \leftarrow S_r$
9      if $r < k$ & $g(S) > \lambda_{r+1}$ then break ;
10 return $S$.
```

the enhancement of the solution, e.g., Greedy++ [22], Sweep-cut [57], or the LP method, etc.

We provide an analysis of the time complexity as,

Theorem 5 (Time Complexity of **SpecGDS** *):* The time complexity of the SpecGDS algorithm is

$$O(K \cdot |E| + K \cdot |E(\tilde{S})| \log |\tilde{S}|),$$

where K is the top approx. rank and $\tilde{S} = \arg\max_{S_i} |S_i|$ with $i \in [K]$. Ideally, $K = \min\{k, r_{opt} + 1\}$ where k is the input parameter ($k \ll \log |V|$) and r_{opt} is the rank with the optimal resultant density g^*.

Proof 1: The complexity of computing a top eigenvector/singular vector in sparse graphs is linear, i.e., $O(|E(V)|)$, and the total complexity of the greedy algorithm in Line 5 is $O(|E(S)| \log |S|)$ for $\mathcal{G}(S)$, Thus the algorithm takes $O(K \cdot |E| + K \cdot |E(\tilde{S})| \log |\tilde{S}|)$ time.

Given the skewness of the top singular vectors in real-world graphs, we usually have $|\tilde{S}| \ll |V|$, making SpecGDS a linear algorithm in the number of edges.

D. Proposed Algorithm: GepGDS

Based on the analysis in Section V-B, we proposed GepGDS to find the densest subgraph under the GEP. Algorithm 3 summarizes the details.

Given the symmetric matrices $\mathbf{P}$ and $\mathbf{Q}$ of the graphs $\mathcal{G}$ and $\mathcal{G}'$ as input, where $\mathbf{Q}$ is strictly diagonally dominant with some γ, GepGDS computes generalized eigenvalue decomposition (GEVD) to obtain the top-k eigenpair (Φ, Λ). Then, it searches for the optimal solution for (3) over these eigenvectors (Lines 3-8). In each loop, a truncation threshold Δ (like $1/\sqrt{n}$) is also used for the eigenvector Φ_r to obtain the set S_r as a candidate solution; and greedily remove nodes in S_r to improve the objective function until convergence. We utilize a similar early stopping criterion to SpecGDS, i.e., comparing the density $g(S)$ with λ_{r+1} (Line 8). It returns the nodeset S w.r.t. the GenDS.

Theorem 6 (Time Complexity of **GepGDS** *):* The time complexity of the GepGDS algorithm is

$$O\left(K \frac{nnz(\mathbf{P}, \mathbf{Q})\sqrt{\kappa}}{\rho} \log \frac{1}{\epsilon} \log \frac{K\kappa}{\rho} + K|\tilde{S}| \log |\tilde{S}|\right),$$

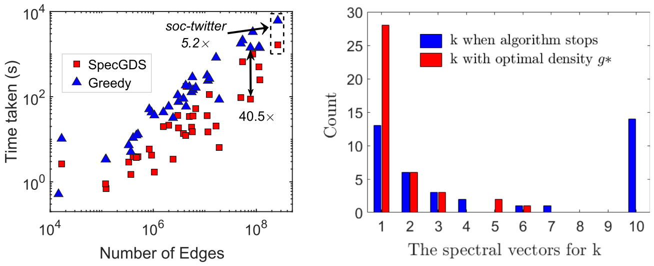
where $nnz(\mathbf{P}, \mathbf{Q}) \overset{\text{def}}{=} nnz(\mathbf{P}) + nnz(\mathbf{Q}) = |E^P| + |E^Q| + |V|$, $\kappa \overset{\text{def}}{=} \max(\kappa(\mathbf{P}), \kappa(\mathbf{Q}))$ with $\kappa(\cdot)$ is the condition number, $\rho \overset{\text{def}}{=} 1 - \frac{|\tilde{\lambda}_{K+1}|}{|\tilde{\lambda}_K|}$, $|\tilde{\lambda}_1| \geq \cdots \geq |\tilde{\lambda}_K|$ being the top-K eigenvalues of $\mathbf{Q}^{-1}\mathbf{P}$, and ϵ is an error, based on the method [83]. K and $\tilde{S}$ are similar to those in Theorem 5.

VII. Experiments

We design experiments to answer the following questions:

1) *Q1. Efficiency:* How does SpecGDS compare to the state-of-the-art greedy algorithm for detecting the densest subgraph?
2) *Q2. Effectiveness:* How well does SpecGDS work on real-world data, and perform on detecting the contrast dense subgraphs and injected subgraphs? How about the performance of the GepGDS?
3) *Q3. Scalability:* Do our methods scale with the input graph size?

Data: We used a variety of datasets (40 in total) obtained from 5 popular publicly available network repositories, including Stanford's SNAP database [84], Network Repository [85], and AMiner scholar datasets [86], etc. There are 32 monopartite graphs and 8 bipartite graphs; 5 of them also have edge weights. Among them, the largest unweighted graph is the `soc-twitter` with 1.47 B edges, while the smallest unweighted graph has roughly 14.5 K edges. Multiple edges and self-loops are removed, and directionality is ignored for directed graphs.



(a) The densest subgraph detected for all graphs.

(b) The statistic information of spectral vectors for k.

Fig. 3. **The performance of SPECGDS for the real-world networks.** (a) SPECGDS runs faster than GREEDY in all graphs for detecting the densest subgraph with the same or comparable density, achieves 58.6× speedup for *ca-DBLP2012* and about 5.2× for the largest graph *soc-twitter*. (b) The statistic information about k for spectral vectors. The densest subgraphs with optimal density $g*$ are achieved in the first singular vector for most of the datasets. The blue bars show the statistics of k when the algorithm stops given the parameter $k = 10$.

Detailed information about those networks is provided in the supplement, available online.

Implementations: We compared efficient dense subgraph detection algorithms. We implemented our methods, GREEDY [7], and SPOKEN [43] in Python (Fraudar [45] is the same as GREEDY in our case; no column weight is needed); SPOKEN detects the densest subgraph only based on the truncation of the singular vectors like our method. For a fair comparison, we reimplement the CoreApp proposed in [18] in Python based on the author-provided source code.

In all the experiments, we set the parameters of the top-approximation rank $k = 10$ and $l = s = 3$ for SPECGDS. We ran all experiments on a machine with 2.4 GHz Intel(R) Xeon(R) CPU and 64 GB of main memory.

A. Q1. Efficiency

To answer Q1, we apply our method SPECGDS and the baseline GREEDY to 40 unweighted networks and compare their runtime.

1) Performance Over Real-World Networks.: Fig. 1(a) shows the statistical information about the runtime improvement ratio of SPECGDS compared with the GREEDY algorithm for detecting the densest subgraphs. Fig. 3(a) illustrates more detailed information about the time taken by the two methods: for each network dataset, it provides their runtimes and network size.

Observation 1: Our method runs faster than GREEDY and achieves the same or comparable optimal densities, as shown in Fig. 1(b). Among these varied-size datasets, SPECGDS achieves 3.0-5.0× speedup for 17 of them, 1.5-3.0× for eight, 5.0-7.0× for seven networks, and more than 58.6× for the `ca-DBLP2012` graph. As we can see, SPECGDS is more efficient for large graphs, e.g., its speedups are up to 30× for `ca-DBLP-NET`, 25× for `cit-Patents`, and 3× for `soc-twitter`.

For the five weighted graphs, we observe similar results as above. SPECGDS achieves 24-39× speedups for 3 of them and 11-17× for the rest. GREEDY will perform poorly for graphs dominated by a few edges with large weights, since it needs to peel each edge of the whole graph.

Moreover, we find that the SPECGDS detects larger (nearly 2× and up to 20.87×) and denser subgraphs than CoreApp for almost all datasets. As for the runtime, SPECGDS runs faster than CoreApp for some very large datasets, including `ca-DBLP-NET` and `com-Orkut`; it takes no more than 2× time for most of the remains and at most 3.64× for some. See detailed results in the supplementary, available online.

Fig. 3(b) summarizes the statistics about spectral vectors k for obtaining the optimal density g^* and the actual k when the algorithm stops. A larger k means taking more time for SVD and detecting candidate subgraphs.

Observation 2: From the results in Fig. 3(b), we can see that the densest subgraphs with the optimal density g^* are achieved in the first spectral vector for most of the datasets, the second one for *six* of the graphs, and only *three* graphs need to check more than *five* singular vectors. There are 26 graphs where SPECGDS stops for the early-stopping condition, while the rest need to check all ten singular vectors due to the small optimal density or flat power-law factor of the singular values.

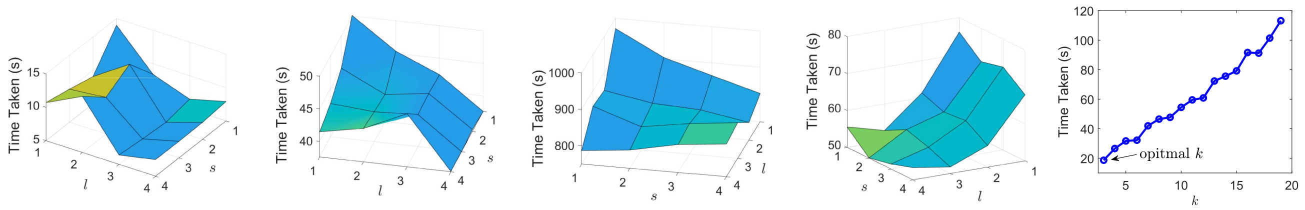
Besides, we find that some subgraphs detected based on the top $k - 1$ vectors are also cliques with a smaller size than the optimal one, like the `soc-twitter` and `ca-DBLP-NET` networks. So, the above heuristic observation and the power-law distribution of singular values contribute to the efficiency of SPECGDS, and the small k is enough for good results.

2) Parameter Analysis: To verify the effect of the parameters on the efficiency of SPECGDS, including (l, s) in the lazy-decomposition in Section VI-C and the rank k in Algorithm 2, we select some real networks to test different parameter settings and report the averaged runtimes of SPECGDS over three trials.

Fig. 4(a)–(d) shows the result under varying (l, s) configurations with $k = 10$ for four networks of various sizes, i.e., `ca-DBLP-NET`, `com-WikiTalk`, `com-Orkut`, and `soc-Youtube`. The first three obtain the optimal solution with the early-stopping condition, and the last stops until the tenth singular vector is checked. We can see that the algorithm reaches the shortest runtime at different (l, s) w.r.t. different networks, and the best configurations of them are $(3, 2)$, $(4, 1)$, $(4, 2)$, and $(4, 3)$, resp.; $l = 1$ usually results in the longest running time, while they all achieved significant improvements in efficiency compared with GREEDY. Therefore, we choose $l = s = 3$ as a *trade-off* for true incremental decomposition when utilizing the off-the-shelf SVD algorithms [87], although they may not be the optimal settings for performance. Fig. 4(e) shows the average runtimes of SPECGDS for different ranks k over `soc-Youtube`. Near linear runtime w.r.t. k verifies the conclusion in Theorem 5. The densest subgraph obtained at the third singular vector, $k = 3$, leads to the shortest runtime.

B. Q2. Effectiveness

In this section, we verify that SPECGDS detects the densest subgraphs with high quality, i.e., higher densities, in real-world



(a) ca-DBLP-NET ($t_{gr} = 162.6, k^* = 2$)

(b) com-WikiTalk ($t_{gr} = 142.7, k^* = 3$)

(c) com-Orkut ($t_{gr} = 2,737.9, k^* = 4$)

(d) soc-Youtube ($t_{gr} = 91.5, \underline{k = 10}$)

(e) soc-Youtube (No early-stopping)

Fig. 4. **Performance vs. different parameter settings.** (a)-(d) shows the runtimes of SpecGDS w.r.t (l, s) for some networks by setting the input parameter $k = 10$. t_{gr} is the running time of Greedy as comparison and k^* is the k when the algorithm stops; Fig. (d) is the only one that does not meet early-stopping conditions (doesn't stop until the end). (e) shows the running time of SpecGDS w.r.t k over `soc-Youtube` where $l = s = 3$. $k = 3$ means the optimal solution is obtained at the 3 rd singular vector, which is also the smallest k while guaranteeing the best density.

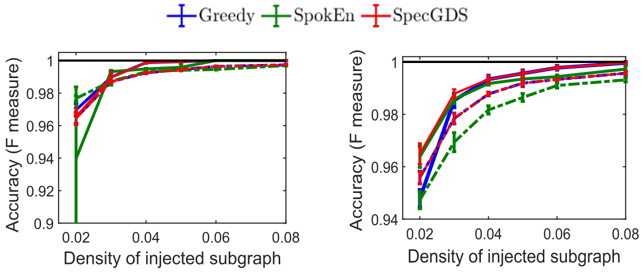
networks and accurately spots the injected subgraphs with different injection densities. Moreover, focusing on a large collaboration network, we empirically analyze the properties of the detection results of GepGDS, and we show that our methods find significant contrast-dense subgraphs.

1) Density Improvement: Following the same setup in Q1, Fig. 1(b) shows the improvement ratio of optimal densities found by SpecGDS compared to the Greedy and SpokEn algorithms.

As we can see, SpecGDS consistently outperforms SpokEn by detecting denser densest subgraphs for all real-world datasets. It even achieves more than 28.3× higher density for the `soc-twitter` network. Also, SpecGDS obtains the same or denser (about 1.26× for the `com-Amazon`) optimal density for most graphs compared with Greedy. There are *four* networks whose optimal densities detected by SpecGDS have less than but very close ($\geq 0.996\times$) densities detected by Greedy, and *two* networks with less than 0.9 density improvement. So, utilizing the spectral distribution of the densest subgraph, SpecGDS can improve the quality of the solution of Greedy in most cases by avoiding arbitrary tie-breaks in graphs for the removal in Greedy to some extent.

We explore the details of the case where SpecGDS achieves density improvement compared to Greedy. Fig. 6(a) shows the detection trace of SpecGDS on the `com-Amazon`, which is the largest connected component of the Amazon product co-purchasing network [84]. The power-law distribution of its singular values is flat, which means it consists of some similarly structured subgraphs; this is verified by the detection results for each singular vector, i.e., most of them have very similar densities. SpecGDS terminates at $k = 10$ based on the parameter setting. The optimal densest subgraph is S_1^* with $g(S_1^*) \approx 4.566$ and $|S_1^*| = 290$, while Greedy returns a subgraph with a size of 57,480 and a density of about 3.624, which means that it is highly affected by those small and similar communities in the original graph and recognizes them as a dense subgraph. This is consistent with the conclusion in [22].

Thus, it helps to avoid being trapped in the local-optimal solution by utilizing the spectral properties of the graphs as SpecGDS. Fig. 6(b) illustrates the network structure of the optimal densest subgraph, which forms a community with a node degree of obviously more than 4.



(a) Injection without camou.

(b) Injection with random camou.

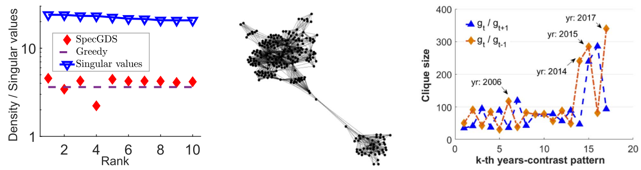
Fig. 5. **Performance comparison for injection detection in the synthetic graphs.** Some dense subgraphs with different densities are injected into the graph; the solid and dash lines correspond to two different subsets of `amazon-Art` data. SpecGDS achieves similar accuracy as the Greedy algorithm and outperforms SpokEn.

2) Injection Detection: We further evaluate the performance of SpecGDS by performing a synthetic experiment where we inject dense subgraphs as the ground truth. For a more realistic setting, we also added extra edges as 'camouflage' between the nodes in the selected injection subgraph and the rest.

We compared SpecGDS, Greedy, and SpokEn regarding the F measure for detecting the injected patterns and reported the average F-score over 5 trials. Specifically, we injected a 600×600 subgraph with different injection densities into the `Amazon-Art` review subgraph of size $4\,K \times 4\,K$. For comparison, we selected two different cases with background densities of 2.7E-5 and 3.4E-5.

Fig. 5 shows the detection accuracy of each method for detecting injected dense subgraphs with different densities. From the results for these low-density injected dense subgraphs, we observe that SpecGDS achieves equally high accuracy as Greedy and is better than SpokEn; the injected camouflage will do harm to the detection performance more or less. SpecGDS and Greedy achieve the best stable results when the injection density reaches a certain threshold (0.05 for no camouflage and 0.08 for random camouflage).

3) Case Study: As a case study for possible applications, we also applied our methods to the DBLP co-authorship data [86] from 2000 to 2017 to identify interesting contrast-dense patterns.



(a) The singular values and densities of detection of GREEDY and SPECGDS.

(b) The structure of the densest subgraph detected by SPECGDS.

(c) Contrast patterns detected from the DBLP co-authorship graphs.

Fig. 6. **Cases study for the patterns detected in real-world graphs.** (a) shows the detection trace (the densest subgraph for each singular vector) of SPECGDS under the setting $k = 10$, the optimal densest subgraph detected by GREEDY, and the top-k singular values for the `com-Amazon` network. (b) the network structure of the optimal densest subgraph (S_1^*) detected by SPECGDS in Fig. 6(a). (c) SPECGDS detects contrast patterns from the `DBLP co-authorship` networks in $2000 - 2017$. There are some very large cliques detected from the positive residual graph $\mathcal{G}_r$ in 2014, 2015, and 2017, they correspond to different publications from some large collaborative groups of different disciplines.

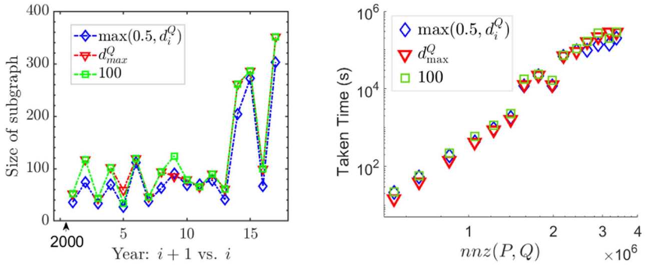


Fig. 7. **Performance of GEPGDS algorithm over the DBLP co-authorship data**. The detected dense subgraph patterns and near linear scalability of GEPGDS under different construction strategies of $\mathbf{Q}$.

Fig. 6(c) shows the contrast dense subgraph patterns detected by SPECGDS by constructing the positive residual graphs, $\mathcal{G}_r$. Those densest contrast subgraphs are all cliques of different sizes, which means the connections that form a clique appear only in $\mathcal{G}_t$ rather than $\mathcal{G}_{t-1}$ (or $\mathcal{G}_{t+1}$).

As we can see, there are three extremely large cliques for 2017, 2015, and 2014, which are related to the publications in 'Brain Network and Disease', 'Neurology and Medicine', and 'Physics' from some large collaborative groups of different disciplines.

Fig. 7 illustrates an example of the GEPGDS algorithm for detecting the contrast dense subgraphs in DBLP datasets with $\mathbf{P}/\mathbf{Q}$ corresponding to g_{i+1}/g_i; it is also near-linear scalable w.r.t. the number of none-zeros $nnz(\mathbf{P}, \mathbf{Q})$ as Theorem 6 states. More results are given in the supplement, available online.

C. Q3. Scalability

Fig. 1(c) shows the linear scaling of SPECGDS's running time with the number of edges in the graph, as Theorem 5 explains. Here, we used the `ca-Patents-AM` network and randomly subsampled different proportions of the edges of it to detect the densest subgraph. The slope parallel to the main diagonal indicates linear growth.

VIII. CONCLUSION

In this paper, we propose the unification form of the generalized densest subgraph detection, GENDS, which subsumes several well-known instances of related problems. We devise SPECGDS and GEPGDS algorithms to solve the generalized problem based on spectral graph properties and a greedy search approach. Our chief contributions include the following:

- *Theory & Correspondences:* We propose a unified formulation for the densest subgraph detection from different applications; give a theoretical analysis of the principles of network flow and spectral graph theory.
- *Algorithm:* We devised SPECGDS and GEPGDS, fast and scalable algorithms to solve the GENDS problem.
- *Experiments:* The efficiency of SPECGDS is verified on 40 real-world graphs. SPECGDS & GEPGDS run linearly with the graph size and are effective for pattern detection in real applications, i.e., they can find sudden bursts in research co-authorship relationships.

However, there are still many directions for the possible extension of this work. Among others, the interesting problem is exploring similar spectral properties in the local scope for streaming graphs and quickly detecting the dense temporal subgraph. The arrival-time locality and connection locality help to avoid exploring an enormous scope in a graph, and a fast detection algorithm with theoretical guarantees that can cooperate with the spectral properties can expand the space of our framework.

REFERENCES

[1] V. E. Lee, N. Ruan, R. Jin, and C. C. Aggarwal, "A survey of algorithms for dense subgraph discovery," in *Managing and Mining Graph Data*, Berlin, Germany: Springer, 2010.

[2] S. W. Wong, C. Pastrello, M. Kotlyar, C. Faloutsos, and I. Jurisica, "SDREGION: Fast spotting of changing communities in biological networks," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 867–875.

[3] Y. Liu, L. Zhu, P. A. Szekely, A. Galstyan, and D. Koutra, "Coupled clustering of time-series and networks," in *Proc. SIAM Int. Conf. Data Mining*, SIAM, 2019, pp. 531–539.

[4] H.-W. Shen and X.-Q. Cheng, "Spectral methods for the detection of network community structure: A comparative analysis," *J. Statist. Mechanics Theory Experiment*, vol. 2010, 2010, Art. no. P10020.

[5] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining Knowl. Discov.*, vol. 29, pp. 626–688, 2015.

[6] A. V. Goldberg, "Finding a maximum density subgraph," Univ. California Berkeley, USA, Tech. Rep., 1984.

[7] M. Charikar, "Greedy approximation algorithms for finding dense components in a graph," in *Proc. Approximation Algorithms Combinatorial Optim., 3rd Int. Workshop*, 2000, pp. 84–95.

[8] A. Gionis and C. E. Tsourakakis, "Dense subgraph discovery: KDD 2015 tutorial," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 2313–2314.

[9] F. D. Malliaros, C. Giatsidis, A. N. Papadopoulos, and M. Vazirgiannis, "The core decomposition of networks: Theory, algorithms and applications," *VLDB J.*, vol. 29, pp. 61–92, 2020.

[10] Y. Fang, W. Luo, and C. Ma, "Densest subgraph discovery on large graphs: Applications, challenges, and techniques," in *Proc. VLDB Endowment*, vol. 15, no. 12, pp. 3766–3769, 2022.

[11] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan, "A fast parametric maximum flow algorithm and applications," *SIAM J. Comput.*, vol. 18, no. 1, pp. 30–55, 1989.

[12] N. Tatti and A. Gionis, "Density-friendly graph decomposition," in *Proc. Int. Conf. World Wide Web*, 2015, pp. 1089–1099.

[13] A. Gionis and C. E. Tsourakakis, "Dense subgraph discovery: KDD 2015 tutorial," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 2313–2314.

[14] C. Tsourakakis, "The k-clique densest subgraph problem," in *Proc. Int. Conf. World Wide Web*, 2015, pp. 1122–1132.

[15] B. Sun, M. Danisch, T. Chan, and M. Sozio, "Kclist : A simple algorithm for finding k-clique densest subgraphs in large graphs," in *Proc. VLDB Endowment*, vol. 13, no. 10, pp. 1628–1640, 2020.

[16] C. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarli, "Denser than the densest subgraph: Extracting optimal quasi-cliques with quality guarantees," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2013, pp. 104–112.

[17] E. Galimberti, F. Bonchi, and F. Gullo, "Core decomposition and densest subgraph in multilayer networks," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2017, pp. 1807–1816.

[18] Y. Fang, K. Yu, R. Cheng, L. V. S. Lakshmanan, and X. Lin, "Efficient algorithms for densest subgraph discovery," in *Proc. VLDB Endow.*, vol. 12, no. 11, pp. 1719–1732, 2019.

[19] Y. Asahiro, K. Iwama, H. Tamaki, and T. Tokuyama, "Greedily finding a dense subgraph," *J. Algorithms*, vol. 34, no. 2, pp. 203–221, 2000.

[20] A. Miyauchi and N. Kakimura, "Finding a dense subgraph with sparse cut," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2018.

[21] M. Danisch, T.-H. H. Chan, and M. Sozio, "Large scale density-friendly graph decomposition via convex programming," in *Proc. Int. Conf. World Wide Web*, 2017, pp. 547–556.

[22] D. Boob et al., "Flowless: Extracting densest subgraphs without flow computations," in *Proc. Web Conf.*, 2020, pp. 573–583.

[23] C. Chekuri, K. Quanrud, and M. R. Torres, "Densest subgraph: Supermodularity, iterative peeling, and flow," in *Proc. ACM-SIAM Symp. Discrete Algorithms*, SIAM, 2022, pp. 1531–1555.

[24] L. Chang and M. Qiao, "Deconstruct densest subgraphs," in *Proc. Web Conf.*, 2020, pp. 2747–2753.

[25] S. Sawlani and J. Wang, "Near-optimal fully dynamic densest subgraph," in *Proc. 52nd Annu. ACM SIGACT Symp. Theory Comput.*, 2020, pp. 181–193.

[26] C. Ma, Y. Fang, R. Cheng, L. V. Lakshmanan, W. Zhang, and X. Lin, "Efficient algorithms for densest subgraph discovery on large directed graphs," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2020, pp. 1051–1066.

[27] C. E. Tsourakakis, T. Chen, N. Kakimura, and J. W. Pachocki, "Novel dense subgraph discovery primitives: Risk aversion & exclusion queries," in *Proc. Mach. Learn. Knowl. Discov. Databases-Eur. Conf. Mach. Learn.*, 2019, pp. 378–394.

[28] R. Andersen and K. Chellapilla, "Finding dense subgraphs with size bounds," in *Proc. 6th Int. Workshop Algorithms Models Web-Graph*, 2009, pp. 25–37.

[29] H. Yanagisawa and S. Hara, "Discounted average degree density metric and new algorithms for the densest subgraph problem," *Networks*, vol. 71, no. 1, pp. 3–15, 2018.

[30] R. A. Rossi, D. F. Gleich, A. H. Gebremedhin, and M. M. A. Patwary, "Fast maximum clique algorithms for large graphs," in *Proc. Int. Conf. World Wide Web*, 2014, pp. 365–366.

[31] M. Mitzenmacher, J. Pachocki, R. Peng, C. Tsourakakis, and S. C. Xu, "Scalable large near-clique detection in large-scale networks via sampling," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 815–824.

[32] H. Liu, L. J. Latecki, and S. Yan, "Dense subgraph partition of positive hypergraphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 541–554, Mar. 2015.

[33] S. Hu, X. Wu, and T. H. Chan, "Maintaining densest subsets efficiently in evolving hypergraphs," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2017, pp. 929–938.

[34] Z. Zou, "Polynomial-time algorithm for finding densest subgraphs in uncertain graphs," in *Proc. MLG Workshop*, 2013.

[35] A. Epasto, S. Lattanzi, and M. Sozio, "Efficient densest subgraph computation in evolving graphs," in *Proc. Int. Conf. World Wide Web*, 2015, pp. 300–310.

[36] M. Svendsen, A. Angel, N. Koudas, N. Sarkas, D. Srivastava, and S. Tirthapura, "Dense subgraph maintenance under streaming edge weight updates for real-time story identification," *VLDB J.*, vol. 23, pp. 175–199, 2014.

[37] A. McGregor, D. Tench, S. Vorotnikova, and H. T. Vu, "Densest subgraph in dynamic graph streams," in *Proc. Int. Symp. Math. Found. Comput. Sci.*, Springer, 2015, pp. 472–482.

[38] Y. Yang, L. Chu, Y. Zhang, Z. Wang, J. Pei, and E. Chen, "Mining density contrast subgraphs," in *Proc. IEEE Int. Conf. Data Eng.*, 2018, pp. 221–232.

[39] L. Chu, Z. Wang, J. Pei, J. Wang, Z. Zhao, and E. Chen, "Finding gangs in war from signed networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 1505–1514.

[40] A. Anagnostopoulos, L. Becchetti, A. Fazzone, C. Menghini, and C. Schwiegelshohn, "Spectral relaxations and fair densest subgraphs," in *Proc. ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 35–44.

[41] J. Chen and Y. Saad, "Dense subgraph extraction with application to community detection," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 7, pp. 1216–1230, Jul. 2012.

[42] A. Costa, "Milp formulations for the modularity density maximization problem," *Eur. J. Oper. Res.*, vol. 245, pp. 14–21, 2015.

[43] B. A. Prakash, A. Sridharan, M. Seshadri, S. Machiraju, and C. Faloutsos, "Eigenspokes: Surprising patterns and scalable community chipping in large graphs," in *Proc. Pacific-Asia Conf. Knowl. Discov. Data Mining*, 2010, pp. 435–448.

[44] A. Beutel, W. Xu, V. Guruswami, C. Palow, and C. Faloutsos, "Copycatch: Stopping group attacks by spotting lockstep behavior in social networks," in *Proc. Int. Conf. World Wide Web*, 2013, pp. 119–130.

[45] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "FRAUDAR: Bounding graph fraud in the face of camouflage," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 895–904.

[46] M. E. Newman, "Modularity and community structure in networks," in *Proc. Nat. Acad. Sci.*, vol. 3, no. 3, pp. 348–360, 2006.

[47] L. Qin, R.-H. Li, L. Chang, and C. Zhang, "Locally densest subgraph discovery," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2015, pp. 965–974.

[48] M. Jiang, A. Beutel, P. Cui, B. Hooi, S. Yang, and C. Faloutsos, "A general suspiciousness metric for dense blocks in multimodal data," in *Proc. IEEE Int. Conf. Data Mining*, 2015, pp. 781–786.

[49] K. Shin, B. Hooi, J. Kim, and C. Faloutsos, "D-cube: Dense-block detection in terabyte-scale tensors," in *Proc. ACM Int. Conf. Web Search Data Mining*, 2017, pp. 681–689.

[50] W. Feng, S. Liu, and X. Cheng, "Catchcore: Catching hierarchical dense subtensor," in *Proc. Mach. Learn. Knowl. Discov. Databases-Eur. Conf. Mach. Learn.*, 2019, pp. 156–172.

[51] P. Shi, K. He, D. Bindel, and J. E. Hopcroft, "Locally-biased spectral approximation for community detection," *Knowl. Based Syst.*, vol. 164, pp. 459–472, 2019.

[52] D. Papailiopoulos, I. Mitliagkas, A. Dimakis, and C. Caramanis, "Finding dense subgraphs via low-rank bilinear optimization," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1890–1898.

[53] R. Andersen and S. M. Cioaba, "Spectral densest subgraph and independence number of a graph," *J. Universal Comput. Sci.*, vol. 13, no. 11, pp. 1501–1513, 2007.

[54] B. A. Miller, M. S. Beard, P. J. Wolfe, and N. T. Bliss, "A spectral framework for anomalous subgraph detection," *IEEE Trans. Signal Process.*, vol. 63, no. 16, pp. 4191–4206, Aug. 2015.

[55] F. R. K. Chung, Spectral graph theory, 1996. [Online]. Available: https://academic.microsoft.com/paper/1578099820

[56] R. Andersen, F. Chung, and K. Lang, "Local graph partitioning using pagerank vectors," in *Proc. IEEE 47th Annu. Symp. Found. Comput. Sci.*, 2006, pp. 475–486.

[57] J. R. Shewchuk, "Allow me to introduce spectral and isoperimetric graph partitioning," Tech. Rep., 2016.

[58] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 555–564.

[59] R. Kannan and V. Vinay, *Analyzing the Structure of Large Graphs*. Bonn, Germany: Forschungsinst. Für Diskrete Mathematik, 1999.

[60] S. Liu, B. Hooi, and C. Faloutsos, "A contrast metric for fraud detection in rich graphs," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 12, pp. 2235–2248, Dec. 2019.

[61] Z. Li, S. Zhang, R.-S. Wang, X.-S. Zhang, and L. Chen, "Erratum: Quantitative function for community detection," *Phys. Rev. E*, vol. 77, no. 3, p. 036109, Mar. 2008, doi: 10.1103/PhysRevE.77.036109.

[62] U. V. Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, 2007.

[63] Z. Wang, L. Chu, J. Pei, A. Al-Barakati, and E. Chen, “Tradeoffs between density and size in extracting dense subgraphs: A unified framework,” in *Proc. IEEE/ACM Int. Conf. Adv. Soc. Netw. Anal. Mining*, 2016, pp. 41–48.

[64] M. Pavan and M. Pelillo, “Dominant sets and pairwise clustering,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 1, pp. 167–172, Jan. 2006.

[65] D. R. Karger and C. Stein, “An $\tilde{O}(n^2)$ algorithm for minimum cuts,” in *Proc. 25th Annu. ACM Symp. Theory Comput.*, 1993, pp. 757–765.

[66] S.-P. Hong, “Inapproximability of the max-cut problem with negative weights,” *Manage. Sci. Financial Eng.*, vol. 14, no. 1, pp. 87–90, 2008.

[67] S. T. McCormick, M. R. Rao, and G. Rinaldi, “Easy and difficult objective functions for max cut,” *Math. Program.*, vol. 94, pp. 459–466, 2003.

[68] A. Dax, “From eigenvalues to singular values: A review,” *Adv. Pure Math.*, vol. 3, 2013, Art. no. 41122.

[69] C. E. Tsourakakis, “Fast counting of triangles in large real networks without counting: Algorithms and laws,” in *Proc. IEEE Int. Conf. Data Mining*, 2008, pp. 608–617.

[70] 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.

[71] E. Croot, “The Rayleigh principle for finding eigenvalues,” Georgia Institute of Technology, School of Mathematics, Tech. Rep., 2005. Accessed: Mar. 2019. [Online]. Available: https://ecroot.math.gatech.edu/notes_linear.pdf

[72] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[73] S. A. Gershgorin, “Uber die abgrenzung der eigenwerte einer matrix,” *Izv. Akad. Nauk. USSR Otd. Fiz.-Mat. Nauk*, vol. 7, no. 3, pp. 749–754, 1931.

[74] M. Mihail and C. Papadimitriou, “On the eigenvalue power law,” in *Proc. Int. Workshop Randomization Approximation Techn. Comput. Sci.*, Springer, 2002, pp. 254–262.

[75] C. R. Palmer and J. G. Steffan, “Generating network topologies that obey power laws,” in *Proc. Glob. Telecommun. Conf.. Conf. Rec.*, 2000, pp. 434–438.

[76] D. Chakrabarti, Y. Zhan, and C. Faloutsos, “R-MAT: A recursive model for graph mining,” in *Proc. SIAM Int. Conf. Data Mining*, SIAM, 2004, pp. 442–446.

[77] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani, “Kronecker graphs: An approach to modeling networks,” *J. Mach. Learn. Res.*, vol. 11, pp. 985–1042, 2010.

[78] N. Eikmeier and D. F. Gleich, “Revisiting power-law distributions in spectra of real world networks,” in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 817–826.

[79] D. Chakrabarti, Y. Zhan, and C. Faloutsos, “R-mat: A recursive model for graph mining,” in *Proc. 4th SIAM Int. Conf. Data Mining*, SIAM, 2004, pp. 442–446.

[80] M. L. Fredman, R. Sedgewick, D. D. Sleator, and R. E. Tarjan, “The pairing heap: A new form of self-adjusting heap,” *Algorithmica*, vol. 1, pp. 111–129, 1986.

[81] N. V. Gudapati, E. Malaguti, and M. Monaci, “In search of dense subgraphs: How good is greedy peeling?,” *Networks*, vol. 77, no. 4, pp. 572–586, 2021.

[82] G. H. Golub and C. F. Van Loan, *Matrix Computations*. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 2012.

[83] R. Ge et al., “Efficient algorithms for large-scale generalized eigenvector computation and canonical correlation analysis,” in *Proc. Int. Conf. Mach. Learn.*, PMLR, 2016, pp. 2741–2750.

[84] J. Leskovec and A. Krevl, “SNAP datasets: Stanford large network dataset collection,” Jun. 2014. http://snap.stanford.edu/data

[85] R. A. Rossi and N. K. Ahmed, “The network data repository with interactive graph analytics and visualization,” in *AAAI*, 2015. [Online]. Available: [Online]. Available: http://networkrepository.com

[86] H. Wan, Y. Zhang, J. Zhang, and J. Tang, “Aminer: Search and mining of academic social networks,” *Data Intell.*, vol. 1, no. 1, pp. 58–76, 2019.

[87] N. Halko, P.-G. Martinsson, and J. A. Tropp, “Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions,” *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.

Wenjie Feng received the PhD degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences in 2020. He is a research fellow with the Institute of Data Science at the National University of Singapore. His research focuses on large-scale graph mining, graph learning, machine learning and application, and anomaly detection.

Shenghua Liu (Member, IEEE) received the PhD degree from the Computer Science and Technology Department, Tsinghua University. He is an associate professor with the Institute of Computing Technology, Chinese Academy of Sciences. He was a visiting scholar with the University of California, Los Angeles and Carnegie Mellon University respectively. His current research interests are designing intelligent and automated algorithms for Big Data mining problems, related to big graphs and series.

Danai Koutra is a Morris Wellman assistant professor in computer science and engineering with the University of Michigan, where she leads the Graph Exploration and Mining, Scale (GEMS) Lab. Her research focuses on practical and scalable methods for large-scale real networks, and has applications in neuroscience, organizational analytics, and social sciences. Her research interests include large-scale graph mining, analysis of multi-source network data, graph summarization, similarity and matching, and anomaly detection.

Xueqi Cheng (Senior Member, IEEE) is a professor with the Institute of Computing Technology, Chinese Academy of Sciences. His main research interests include network science, web search and data mining, Big Data processing and distributed computing architecture, etc. He has published more than 100 publications in prestigious journals and conferences, including the *IEEE Transactions on Information Theory*, *IEEE Transactions on Knowledge and Data Engineering*, *Journal of Statistics Mechanics: Theory and Experiment*, *Physical Review E.*, ACM SIGIR, WWW, ACM CIKM, WSDM, IJCAI, ICDM, etc.