Unveiling Densest Multilayer Subgraphs via Greedy Peeling

Dandan Liu[†], Zhaonian Zou[*], Run-An Wang[‡]
[†*‡]Harbin Institute of Technology, China
[†]ddliu@hit.edu.cn [*]znzou@hit.edu.cn [‡]wangrunan@stu.hit.edu.cn

***Abstract*—Densest subgraphs in multilayer (ML) graphs unveil intricate relationships that are hidden from the simple graph representations, offering profound insights and applications across diverse domains. In this paper, we present a layer-oriented view of the existing density measures for ML graphs and highlight the problems in identifying the densest subgraphs under the layer-oriented design, including inefficiency, poor approximation guarantees, and the lack of a unified algorithmic framework. In light of this, we introduce a new family of vertex-oriented density measures called generalized density. Controlled by two parameters q and p, the generalized density provides the flexibility to shift focus on higher or lower vertex degrees and the degrees across layers. We investigate the problem of finding the densest ML subgraphs based on this density framework and show that half of the problem space can be addressed using a unified greedy peeling paradigm. We delve into four settings of q and p, carefully design the vertex selection strategies for the peeling process, and provide non-trivial approximation guarantees and complexity results. We also present a near-linear-time fast implementation to expedite the greedy peeling algorithm for certain cases. Experiments on 10 real-world ML graphs show that our generalized density and greedy peeling algorithms effectively uncover various types of dense subgraphs in large-scale ML graphs.**

***Index Terms*—Multilayer graph, densest subgraph, approximation algorithm**

I. Introduction

Multilayer (ML) graph models are effective tools for modeling intricate real-world relationships established through various channels, such as interactions between individuals across diverse social media platforms [1], transportation links between hubs using different modes [2], and relationships among biomolecular entities derived from multiple biological experiments [3], [4]. As depicted in Fig. 1, an ML graph is represented as a set of layered graphs, with each graph (also called layer) naturally describing a specific kind of relationship.

Identifying dense structures within large graphs is a key graph primitive that has been extensively studied for decades. It finds applications in diverse domains [3], [5]–[16]. While dense subgraph discovery in simple graphs has made significant progress and is close to being fully resolved [16], its adaptation to ML graph settings is still in its infancy.

Finding dense subgraphs in ML graphs holds the potential to reveal more reliable and meaningful patterns unseen from single-layer representations [3], [17]–[19]. For example, in biological research, researchers organize gene co-expression networks derived from multiple microarray datasets into ML graphs and identify functionally homogeneous clusters by extracting vertices that are densely connected on multiple layers. This approach has been demonstrated to effectively reduce spurious edges compared to deriving clusters from a single microarray dataset [3].

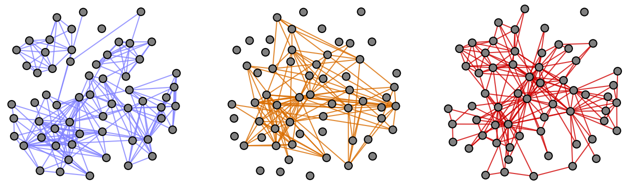


Fig. 1: An example of an ML graph extracted from the AUCS dataset [1] with three layers representing different relationships between employees in a university.

The task of identifying dense subgraphs is commonly approached as a combinatorial optimization problem of finding an induced subgraph that maximizes a density measure. In the context of ML graphs, density measures are usually formulated in a layer-oriented manner: the density of an ML graph is defined as the aggregation of the densities of individual layers. Different density measures [20], [21] utilize distinct single-layer density measures, such as the average degree density [22] or the minimum degree density [23], coupled with specific aggregation methods like averaging or taking the minimum. Recent efforts have recognized the impact of noise or insignificant layers and proposed measures of multilayer density [24] and p-mean multilayer density [25]. These measures identify a subset of layers that maximizes the density of the resulting subgraph and use its density to represent the overall density.

However, designing algorithms to optimize these layer-oriented density measures is challenging. Existing solutions often face scalability issues [20], [26], [27] or offer poor quality guarantees [27]. More importantly, these layer-oriented densities are typically studied in isolation, lacking a unified, scalable, and effective algorithmic framework that can provide insight into solutions for maximizing other layer-oriented density measures for various application scenarios.

In light of these challenges, we introduce a new family of density measures for ML graphs, which adopts an inherently different vertex-oriented design. The key idea is to compute a representative aggregated degree for each vertex that reflects its overall degree information across layers and then obtain the density of the ML graph by incorporating these aggregated

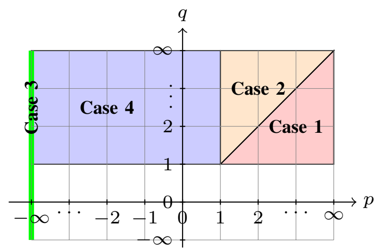


Fig. 2: An partitioning of the problem space in terms of the setting of the parameters p and q of the proposed vertex-oriented density measure.

TABLE I: The approximation ratios and the time complexity of the algorithms proposed for handling different cases, where V, E and L are the sets of vertices, edges and layers in the input ML graph, respectively, and $\epsilon \in [0, 1]$.

Case	Condition	Approximation Ratio	Time Complexity
1	$p \geq q \geq 1$	$(1+p)^{1/p}$	$O(\sum_{v \in V} d_v^2 \lvert L\rvert \log \lvert V\rvert)$
		$\left((1+2\epsilon)(1+p\lvert L\rvert^{1-\frac{1}{q}})\right)^{\frac{1}{p}}$	$O\left(\frac{\lvert E\rvert \log \lvert V\rvert \log \lvert L\rvert\lvert V\rvert}{\log(1+\frac{\epsilon}{p-1})}\right)$
2	$q \geq p \geq 1$	$\left(1+p\lvert L\rvert^{1-\frac{1}{q}}\right)^{1/p}$	$O(\sum_{v \in V} d_v^2 \lvert L\rvert \log \lvert V\rvert)$
		$\left((1+2\epsilon)(1+p\lvert L\rvert^{1-\frac{1}{q}})\right)^{\frac{1}{p}}$	$O\left(\frac{\lvert E\rvert \log \lvert V\rvert \log \lvert L\rvert\lvert V\rvert}{\log(1+\frac{\epsilon}{p-1})}\right)$
3	$p = -\infty$	1	$O(\lvert L\rvert\lvert E\rvert + \lvert E\rvert \log \lvert V\rvert)$
4	$-\infty < p \leq 1 \leq q$	$\left(1+\lvert L\rvert^{1-\frac{1}{q}}\right)$	$O(\lvert L\rvert\lvert E\rvert + \lvert E\rvert \log \lvert V\rvert)$

degrees using a single-layer density measure.

Leveraging the power of generalized means, which allow for flexible emphasis on larger or smaller elements in averaging and have been successfully adopted in the p-density framework for simple graphs [28], we introduce a vertex-oriented generalized density for ML graphs. Specifically, we define the q-mean degree of a vertex as the q-mean of its degrees across all layers. The generalized density, also called (q, p)-density, is then calculated as the ratio of the p-mean of the q-mean degrees of vertices to the total number of vertices. The (q, p)-density framework offers flexibility in adjusting the focus on higher or lower cross-layer degrees of vertices and the q-mean degrees of different vertices through parameters q and p. This flexibility enables the characterization of a wide range of dense patterns. Notably, when $q = p$, the (q, p)-density also shows equivalence to certain layer-oriented density measures [21].

This paper studies the *generalized densest ML subgraph problem*, which aims to find the densest subgraph in an ML graph that maximizes the (q, p)-density. We investigate this problem under four settings of q and p, as outlined in Fig. I, and address these cases with a unified greedy peeling framework [6], [29]–[33]. For each setting, we customize the peeling algorithm with a specific vertex selection strategy. An additional fast implementation is also proposed for Case 1 and Case 2. Table I summarizes the approximation guarantees and the time complexity results of the peeling algorithms proposed for handling different cases.

We apply our (q, p)-density framework and algorithms on 10 real-world ML graphs and obtain the following key results: (1) The (q, p)-density and the proposed algorithms can effectively uncover dense subgraphs with diverse characteristics in ML graphs. The q-mean degree can mitigate the influence of noisy layers, achieving a similar effect to selecting significant layers in layer-oriented density measures. (2) Our greedy peeling algorithms are fast and can gracefully scale to large graphs. They show greater efficiency improvements over layer-oriented densest subgraph detection algorithms as the number of layers in the ML graph increases.

The main contributions of this work are given below. **(1)** We formalize existing ML graph density measures within a layer-oriented framework and highlight the limitations of the densest ML subgraph detection algorithms based on this framework. **(2)** We introduce a novel vertex-oriented density measure for ML graphs and thoroughly discuss its features and relations to existing density measures. **(3)** We customize different versions of the greedy peeling framework to tackle the generalized densest ML subgraph problem under four settings of parameters q and p. The approximation guarantees and time complexity results are summarized in Table I. **(4)** We evaluate our density measure and algorithms on 10 real-world ML graphs. They showcase the ability to uncover various types of dense subgraphs in large-scale ML graphs.

II. Motivation

In this section, we discuss the motivation of this paper.

A. Preliminaries

Multilayer Graphs: A *multilayer graph* (abbreviated as *ML graph*) is a triple $\mathcal{G} = (V, E, L)$, where V is the set of vertices, $L = \{1, 2, \ldots, |L|\}$ is the set of layer numbers, and $E \subseteq V \times V \times L$ is the set of edges. An edge $(u, v, i) \in E$ indicates that vertices u and v are adjacent on layer i. All vertices and edges on layers i constitute the graph on layer i, denoted as $G_i = (V, E_i)$, where $E_i = \{(u, v, i) | u, v \in V\}$. For any vertex $v \in V$, $N_v(i)$ represents the set of neighbors of v in G_i, and $d_v(i) = |N_v(i)|$ represents the degree of v in G_i. When the layer is not explicitly specified, we use N_v to denote the set of all neighbors of v, i.e., $N_v = \bigcup_{i \in L} N_v(i)$, and let $d_v = |N_v|$.

Given a vertex subset $S \subseteq V$, the subgraph of G_i induced by S is denoted as $G_i[S] = (S, E_i[S])$, where $E_i[S]$ represents the set of edges in E_i with both endpoints in S. Similarly, the subgraph of the ML graph $\mathcal{G}$ induced by S is denoted as $\mathcal{G}[S] = (S, E[S], L)$, where $E[S]$ represents the set of edges in E with both endpoints in S. The set of neighbors and the degree of any vertex $v \in S$ in $G_i[S]$ are denoted as $N_v(S, i)$ and $d_v(S, i)$, respectively. Moreover, we let $N_v(S) = \bigcup_{i \in L} N_v(S, i)$ and $d_v(S) = |N_v(S)|$.

Generalized Means: The *generalized mean*, also known as the *Hölder mean* or p-mean, of a vector of positive real numbers $\mathbf{x} = [x_1, x_2, \cdots, x_n] \in \mathbb{R}_+^n$ with exponent p is

$$M_p(\mathbf{x}) = \left(\frac{1}{n} \sum_{i=1}^{n} x_i^p\right)^{1/p},$$

where $p \in \mathbb{R} - \{0\}$. When $p \in \{-\infty, 0, \infty\}$, the generalized mean can be defined by taking limits, and we have $M_{-\infty}(\mathbf{x}) = \min_{i=1}^{n} x_i, M_0(\mathbf{x}) = \left(\prod_{i=1}^{n} x_i\right)^{1/n}$, and $M_{\infty}(\mathbf{x}) = \max_{i=1}^{n} x_i$. Particularly, the 1-mean is the arithmetic mean, while the 0-mean is the geometric mean.

This paper extends the domain of x_i to include 0, i.e., $\mathbf{x} \in \mathbb{R}^n_{0+}$. Without loss of generality, suppose $x_1 = x_2 = \cdots = x_m = 0$, where $m \le n$. The p-mean of $\mathbf{x}$ is defined as

$$M_p(\mathbf{x}) = \lim_{(x_1,x_2,\ldots,x_m)\to(0^+,0^+,\ldots,0^+)} \left(\frac{1}{n}\sum_{i=1}^{n} x_i^p\right)^{1/p}. \quad (1)$$

For brevity, we omit the lim symbol in Eq. (1).

Proposition 1 (MONOTONICITY). *For all $\mathbf{x} \in \mathbb{R}^n_{0+}$, $M_{p_1}(\mathbf{x}) \le M_{p_2}(\mathbf{x})$ if $p_1 \le p_2$.*

Generalized Vector Norms: Vector norms quantify the "length" of vectors in a vector space. The *p-norm* or *L^p-norm* for $p \ge 1$ of a vector $\mathbf{x} = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^n$ is defined as

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{1/p}. \quad (2)$$

Proposition 2 (TRIANGLE INEQUALITY). *For all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, $\|\mathbf{x}+\mathbf{y}\|_p \le \|\mathbf{x}\|_p + \|\mathbf{y}\|_p$.*

The p-norms with different $p \ge 1$ satisfy the following key relationship used to derive the results in this paper.

Proposition 3. *For all $\mathbf{x} \in \mathbb{R}^n$ and $p \ge 1$, $\|\mathbf{x}\|_1 \le n^{1-1/p}\|\mathbf{x}\|_p$.*

From Eq. (1) and Eq. (2), for all $\mathbf{x} \in \mathbb{R}^n_{0+}$ and $p \ge 1$, we can write $M_p(\mathbf{x}) = n^{-1/p}\|x\|_p$.

B. Layer-Oriented Density of ML Graphs

In the literature, the density of an ML graph $\mathcal{G} = (V, E, L)$ is typically defined in a *layer-oriented* manner. This approach aggregates the densities of the individual layers in $\mathcal{G}$ and can be expressed in the following unified way:

$$\delta_{q,p}(\mathcal{G}) = M_q([M_p([d_v(l)]_{v\in V})]_{l\in L}).$$

The unified layer-oriented density formulation is either a generalization or a component of many existing density measures for ML graphs. Fig. 3 depicts the relationships among these layer-oriented density measures.

(1) *Common density.* Jethava and Beerenwinkel [20] studied the first densest subgraph problem on ML graphs, which extracts the densest common subgraph (DCS) maximizing the common density. Specifically, the common density of an ML graph $\mathcal{G} = (V, E, L)$ is defined as

$$\delta^{COM}(\mathcal{G}) = MIN([AVG([d_v(l)]_{v\in V})]_{l\in L}),$$

where MIN returns the minimum, and AVG returns the average. Because $M_{-\infty}(\cdot) = MIN(\cdot)$ and $M_1(\cdot) = AVG(\cdot)$, we have $\delta^{COM}(\mathcal{G}) = \delta_{-\infty,1}(\mathcal{G})$.

(2) *Aggregate density.* Semertzidis et al. [21] studied the best friends forever (BBF) problem. Given an ML graph and two

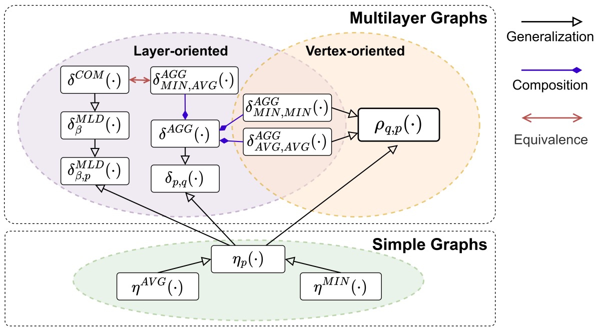


Fig. 3: The relationships between the density measures for simple graphs and ML graphs.

aggregation operations $AGG_1, AGG_2 \in \{MIN, AVG\}$, the BBF problem finds the induced subgraph with the maximum aggregate density. The aggregate density of an ML graph $\mathcal{G} = (V, E, L)$ is defined as

$$\delta^{AGG}(\mathcal{G}) = AGG_1([AGG_2([d_v(l)]_{v\in V})]_{l\in L}).$$

The common density $\delta^{COM}(\mathcal{G})$ is apparently a special case of $\delta^{AGG}(\mathcal{G})$ for $AGG_1 = MIN$ and $AGG_2 = AVG$. $\delta^{AGG}(\mathcal{G})$ is a special case of $\delta_{q,p}(\mathcal{G})$ for $q, p \in \{-\infty, 1\}$.

(3) *Multilayer density.* Considering the trade-off between the high density of a subgraph on some layers and the number of layers where this density holds, Galimberti et al. [24] define the multilayer density of an ML graph $\mathcal{G} = (V, E, L)$ as

$$\delta^{MLD}_{\beta}(\mathcal{G}) = \max_{L'\subseteq L} |L'|^{\beta} \cdot \delta_{-\infty,1}(\mathcal{G}|_{L'}),$$

where $\mathcal{G}|_{L'}$ denotes the ML graph with layers in L', and $\beta \ge 0$ controls the tradeoff. $\delta^{MLD}_{\beta}(\cdot)$ is a composition of $\delta_{q,p}(\cdot)$.

(4) *p-Mean multilayer density.* Behrouz et al. [25] extends the multilayer density to the p-mean multilayer density:

$$\delta^{MLD}_{\beta,p}(\mathcal{G}) = \max_{L'\subseteq L} |L'|^{\beta} \cdot \delta_{-\infty,p}(\mathcal{G}|_{L'}).$$

Certainly, $\delta^{MLD}_{\beta,p}(\cdot)$ is a composition of $\delta_{q,p}(\cdot)$.

C. Problems in Optimizing Layer-Oriented Density Measures

The densest ML subgraph problems optimizing the layer-oriented density measures are often difficult to solve. Existing solutions address only special cases and typically face scalability issues or yield suboptimal results. Moreover, there is a lack of a unified approach to tackle these problems effectively.

(1) The DCS problem [20], which aims to maximize the common density $\delta^{COM}(\cdot)$, cannot be approximated within a factor of $2^{\log^{(1-\epsilon)}|V|}$, where $\epsilon \in (0, 1)$ [27]. The algorithm based on linear programming [20] is slow and cannot scale to large ML graphs. The greedy-selection-based algorithm [27] produces $\sqrt{|V|\log|L|}$-approximate solutions, and the greedy peeling algorithm [20] yields $2r$-approximations, where r depends on the features of the input ML graph with an uncommon best-case value of 1. These approximation algorithms are typically unstable and may produce bad results.

(2) The BFF problem [21] based on the aggregate density $\delta^{AGG}(\cdot)$ with $AGG_1 = AVG$ and $AGG_2 = MIN$ is NP-hard

and cannot be approximated within a factor of $|V|^{1-\epsilon}$, where $\epsilon \in [0,1]$. The exact algorithm [27] established on ML core decomposition [24] requires $O(|V|^{|L|}(|V||L|+|E|))$ time and is inapplicable to large ML graphs with a lot of layers. The approximation algorithm [21] based on greedy peeling attains a poor approximation ratio of $|V|$.

(3) The densest ML subgraph problem based on the multilayer density $\delta_{\beta}^{MLD}(\cdot)$ is also NP-hard [24]. Galimberti et al. [24] prove that the densest ML core in the input ML graph is a $2|L|^{\beta}$-approximation to the optimal solution. However, finding the densest ML core requires exponential time in $|L|$. In addition, the best approximation ratio of 2 is achieved when $\beta \to 0$. In cases where dense subgraphs across more layers are preferred, a larger β value is necessary, leading to significant degradation to the approximation ratio.

(4) The densest ML subgraph problem based on the p-mean multilayer density $\delta_{\beta,p}^{MLD}(\cdot)$ is certainly NP-hard [25]. Behrouz et al. [25] propose a cohesive subgraph model called generalized FirmCore and use the densest generalized FirmCore to approximate the optimal solution, achieving an approximation ratio of $\frac{(p+1)^{1/p}|L|^{\beta+1/p}}{\lambda^{\max\{1,\beta\}}}$, where $\lambda \in [1,|L|]$ depends on the features of the input ML graph. Finding the desired generalized FirmCore requires conducting $|L|$ peeling processes on the input ML graph, running in $O(|E||L|^2+|V||L|\log|L|)$ time. Notably, when $p=1$, this algorithm provides a better approximation for the problem of maximizing the multilayer density $\delta_{\beta}^{MLD}(\cdot)$ for $\beta > 1$. However, despite the improvements, this algorithm still shares some limitations of the ML-core-based approach, including being time-consuming when handling a large number of layers and the approximation ratio tending to degrade as β increases.

The limitations of these densest ML subgraph extraction algorithms originate from the layer-oriented design of the density measures they aim to optimize. In the rest of this paper, we introduce a new vertex-oriented density for ML graphs. This measure not only can characterize a wide range of dense patterns in ML graphs but also has elegant properties that enable half of the problems in the problem space (as shown in Fig. 2) to be efficiently solved with sound approximation guarantees using a unified greedy peeling framework.

III. Problem Statement

In this section, we propose a new family of density measures for ML graphs and formalize the problem studied in this paper.

1) **Vertex-Oriented Density of ML Graphs:** The layers in an ML graph usually have different structures, so the degree of a vertex varies across layers.

Definition 1 (Degree Vector). *Given an ML graph $\mathcal{G}=(V,E,L)$, the degrees of a vertex $v \in V$ on all layers form the degree vector $[d_v(1), d_v(2), \cdots, d_v(|L|)]$ of v.*

Let $\mathbf{d}_v$ denote the degree vector of the vertex v. To extend the p-density [28] to ML graphs, we introduce a representative aggregated vertex degree that reflects the overall degree information of a vertex across layers:

Definition 2 (q-mean degree). *Given an ML graph $\mathcal{G}=(V,E,L)$ and $q \in \mathbb{R}\cup\{-\infty,\infty\}$, the q-mean degree d_v^q of a vertex $v \in V$ is the q-mean of the degree vector $\mathbf{d}_v$.*

The q-mean degree offers a flexible and comprehensive way to characterize a vertex's degree distribution across layers. For $q=1$, the q-mean degree is the average vertex degree across layers. As q grows larger, more emphasis is placed on larger degrees. For $q=\infty$, the q-mean degree is the maximum degree across layers. For $q=-\infty$, the q-mean degree becomes the minimum vertex degree on all layers. By adjusting q, we can capture different aspects of a vertex's degree distribution, tailored to the specific characteristics of the ML graphs and application needs. In the special case where $\mathcal{G}$ consists of only one layer, i.e., $|L|=1$, the q-mean degree of a vertex is its degree on the only layer.

Then, we give a formal definition of the *vertex-oriented generalized density* of an ML graph:

Definition 3 (Vertex-Oriented Generalized Density). *Given an ML graph $\mathcal{G}=(V,E,L)$ and $q,p \in \mathbb{R}\cup\{-\infty,\infty\}$, the generalized density (or the (q,p)-density) of $\mathcal{G}$ is the p-mean of the q-mean degrees of all vertices in V, denoted as*

$$\rho_{q,p}(\mathcal{G}) = M_p([d_v^q]_{v\in V}).$$

The vertex-oriented generalized density $\rho_{q,p}(\cdot)$ is parameterized by two parameters q and p, with q adjusting emphasis on each vertex's cross-layer degrees and p shifting focus towards higher or lower q-mean degrees of vertices in the ML graph. The (q,p)-density exhibits a monotonic property that directly follows the monotonicity of generalized means (Proposition 1):

Lemma 1 (Monotonicity). *$\rho_{q_1,p_1}(\mathcal{G}) \le \rho_{q_2,p_2}(\mathcal{G})$ if $q_1 \le q_2$ and $p_1 \le p_2$.*

2) **Relationships with Existing Density Measures:** As depicted in Fig. 3, the vertex-oriented generalized density $\rho_{q,p}(\cdot)$ generalizes a number of existing density measures.

(1) *Aggregate density.* Recall the aggregate density $\delta^{AGG}(\cdot)$ [21] described in Section II-B. For $AGG_1 = AGG_2 = MIN$, we have

$$\begin{aligned}\delta^{AGG}(\mathcal{G}) &= \min([\min([d_v(l)]_{v\in V})]_{l\in L})\\ &= \min([\min([d_v(l)]_{l\in L})]_{v\in V}) = \rho_{-\infty,-\infty}(\mathcal{G}).\end{aligned}$$

Similarly, for $AGG_1 = AGG_2 = AVG$, we have

$$\begin{aligned}\delta^{AGG}(\mathcal{G}) &= \frac{1}{|L|}\sum_{l\in L}\frac{1}{|V|}\sum_{v\in V} d_v(l)\\ &= \frac{1}{|V|}\sum_{v\in V}\frac{1}{|L|}\sum_{l\in L} d_v(l) = \rho_{1,1}(\mathcal{G}).\end{aligned}$$

(2) *p-Density.* The *p-density* [28] of a simple graph $G=(V,E)$ is the p-mean of the degrees of all vertices in G, denoted as $\eta_p(G)$. For the ML graph $\mathcal{G}=(V, E\times\{1\},\{1\})$ consisting of only one layer $G=(V,E)$, $\rho_{q,p}(\mathcal{G}) = \eta_p(G)$.

(3) *Average Degree Density.* The *average degree density* [22] of a simple graph $G=(V,E)$ is $\eta^{AVG}(G) =$

Algorithm 1 MGDS Framework

```
Input: An ML graph $\mathcal{G} = (V, E, L)$ and two parameters $q, p \in \mathbb{R} \cup \{-\infty, \infty\}$
Output: The subgraph $\mathcal{G}[S]$ where $S \subseteq V$ that maximizes the $(q, p)$-density $\rho_{q,p}(S)$
1: $S_1 \leftarrow V$; $k \leftarrow 1$
2: for $i \leftarrow 1$ to $|V|$ do
3:     $u \leftarrow \arg\min_{v \in S_i} \texttt{score}_{q,p}(v, S_i)$
4:     $S_{i+1} \leftarrow S_i - \{u\}$
5:     if $\rho_{q,p}(S_i) > \rho_{q,p}(S_k)$ then
6:         $k \leftarrow i$
7: return $\mathcal{G}[S_k]$
```

$|E|/|V|$. For the ML graph $\mathcal{G} = (V, E \times \{1\}, \{1\})$ containing one layer $G = (V, E)$, $\rho_{q,1}(\mathcal{G}) = 2\eta^{AVG}(G)$.

(4) *Minimum Degree Density.* The *minimum degree density* [23] of a simple graph $G = (V, E)$ is the minimum degree of all vertices in G, denoted as $\eta^{MIN}(G)$. For the ML graph $\mathcal{G} = (V, E \times \{1\}, \{1\})$, we have $\rho_{q,-\infty}(\mathcal{G}) = \eta^{MIN}(G)$.

3) **Problem Formulation:** The *vertex-oriented generalized densest ML subgraph (MGDS) problem* studied in this paper is stated as follows: Given an ML graph $\mathcal{G} = (V, E, L)$ and $q, p \in \mathbb{R} \cup \{-\infty, \infty\}$, find the induced subgraph $\mathcal{G}[S]$ where $S \subseteq V$ that has the maximum generalized density $\rho_{q,p}(\mathcal{G}[S])$.

For ease of notation, we write the degree vector of a vertex v in a subgraph $\mathcal{G}[S]$ as $\mathbf{d}_v(S)$, the q-mean degree of v in $\mathcal{G}[S]$ as $d_v^q(S)$ and the (q, p)-density of $\mathcal{G}[S]$ as $\rho_{q,p}(S)$ in the rest of this paper.

IV. Algorithms

In this section, we propose a number of algorithms for the MGDS problem. These algorithms are designed based on the well-known greedy peeling framework [6], [29]–[33], wherein the condition for choosing a vertex to be peeled (removed from the graph) depends on the setting of q and p. We provide the theoretical bounds on the accuracy and the time complexity of these algorithms for the cases when $q \geq 1$ or $p = -\infty$, as detailed in Table I.

A. Greedy Peeling Framework

The greedy peeling framework has been widely adopted in finding the densest subgraphs [6], [29], [32], [33] or cohesive graphs [17], [24], [34], [35] from various types of graphs. Algorithm 1 outlines its adaptation to solve the MGDS problem. Given an ML graph $\mathcal{G} = (V, E, L)$, the for loop in lines 2–6 repeats the greedy peeling step. For $i = 1, 2, \ldots, |V|$, let $S_i \subseteq V$ be the set of vertices remaining before the ith peeling step. Initially, S_1 is set to V. In the ith peeling step, a vertex $v \in S_i$ with the minimum score $\texttt{score}_{q,p}(v, S_i)$ is removed from S_i, thus obtaining S_{i+1} (lines 3–4). The formulation of $\texttt{score}_{q,p} : V \times 2^V \to \mathbb{R}$ depends on q and p, which will be introduced in the rest of this section. The greedy peeling step repeats $|V|$ times until all vertices have been removed. The algorithm then returns the induced subgraph $\mathcal{G}[S_i]$ that has the highest (q, p)-density $\rho_{q,p}(S_i)$.

The formulation of the score function $\texttt{score}_{q,p}(\cdot, \cdot)$ is pivotal in ensuring a sufficiently good approximation to the densest ML subgraph. We have studied the design of the score function and found that the suitable score function depends on the values of q and p. In the rest of this section, we propose specific score function formulations for four settings of q and p, as summarized in Table II.

TABLE II: The score functions adopted in different cases.

Case	Condition	Score Function $\texttt{score}_{q,p}(v, S)$	Description
1	$1 \leq q \leq p$	$\Delta_{q,p}(v, S)$ (Eq. (4))	Section IV-B
2	$1 \leq p \leq q$	$\hat{\Delta}_{q,p}(v, S)$ (Eq. (5))	Section IV-C
3	$p = -\infty$	$\|\mathbf{d}_v(S)\|_q$	Section IV-D
4	$-\infty < p \leq 1 \leq q$	$\|\mathbf{d}_v(S)\|_q$	Section IV-E

B. Case 1 ($1 \leq q \leq p$)

This subsection considers the situation $1 \leq q \leq p$. For $S \subseteq V$, we devise another objective function

$$f_{q,p}(S) = \frac{1}{|S|} \sum_{v \in S} \left(\sum_{l \in L} d_v(S, l)^q \right)^{p/q}. \tag{3}$$

Apparently, $\rho_{q,p}(S) = \left(\frac{1}{|L|}\right)^{1/q} f_{q,p}(S)^{1/p}$. Thus, the subset $S \subseteq V$ maximizing $f_{q,p}(S)$ must maximize $\rho_{q,p}(S)$. A c-approximation to the problem of maximizing $f_{q,p}(\cdot)$ implies a $c^{1/p}$-approximation to the problem of maximizing $\rho_{q,p}(\cdot)$.

Let $g_{q,p}(S)$ be the numerator of $f_{q,p}(S)$. We next present a key property of $g_{q,p}(\cdot)$, which establishes the design of the score function for this case and its approximation guarantee.

Definition 4. *Given a discrete set X, let $h : 2^X \to \mathbb{R}$ be a function. The function h is* supermodular *if $h(A) + h(B) \leq h(A \cap B) + h(A \cup B)$ for all $A, B \subseteq X$.*

Lemma 2. *For $p \geq q \geq 1$, $g_{q,p}(\cdot)$ is supermodular.*

Proof. It is easy to extend the proof of Lemma 4.1 in [28] to prove this lemma. See Appendix A of the full paper [36]. ☐

Based on Lemma 2, any instance of the MGDS problem for this case can be reduced to a series of instances of the supermodular maximization problem, which can be solved exactly in polynomial time. We present a detailed solution in Appendix B of the full paper. However, this exact method is impractical for large real-world ML graphs, and thus, we proceed to explore a more practical approximate solution using the greedy peeling framework described in Algorithm 1.

The key idea behind this approximate solution is to greedily improve the density of the remaining subgraph by iteratively removing vertices that cause the minimum decrease of $g_{q,p}(\cdot)$, as defined below:

$$\Delta_{q,p}(v, S) = g_{q,p}(S) - g_{q,p}(S - \{v\}), \tag{4}$$

where $S \subseteq V$ and $v \in S$. We have the following result:

Theorem 1. *For $p \geq q \geq 1$, if $\Delta_{q,p}(v, S)$ is used as the score function in Algorithm 1, the subgraph returned by Algorithm 1 is a $(p+1)^{1/p}$-approximation to the optimal solution.*

Since $g_{q,p}(S)$ is supermodular, the problem of maximizing $f_{q,p}(S)$, that is, $g_{q,p}(S)/|S|$ is a special case of the densest supermodular subset (DSS) problem studied by Chekuri et al. [30]. The DSS problem is rephrased as follows: Given a ground set X and a non-negative supermodular function $h : 2^X \to \mathbb{R}_{\geq 0}$ where $h(\emptyset) = 0$, find the subset $A \subseteq X$ to maximize $h(A)/|A|$.

Theorem 2 ([30]). *By iteratively removing elements minimizing the decrement of $h(X)$ from X, the subset X' obtained in the process with the highest $h(X')/|X'|$ is a c_h-approximation to the optimal solution of the DSS problem, where*

$$c_h = \max_{A \subseteq X} \frac{\sum_{v \in A} h(A) - h(A - \{v\})}{h(A)}.$$

Based on Theorem 2, we can establish the result in Theorem 1 by letting $h = g_{q,p}$ and showing the following lemma:

Lemma 3. *For $p \geq q \geq 1$, we have $c_h \leq p + 1$.*

Proof. See Appendix C of the full paper [36]. □

Time Complexity: For a vertex subset $S \subseteq V$, computing the score of a vertex $v \in S$ costs $O(|L| d_v(S))$ time. Consider using a set or min-heap to dynamically maintain the scores of vertices. Computing the scores of all vertices and initializing the data structure cost $O(\sum_{v \in V} |L| d_v(V) + |V| \log |V|) = O(|L||E| + |V| \log |V|)$ time. Each peeling step identifies and removes a vertex with the minimum score, taking $O(\log |V|)$ time. When a vertex v is removed, the scores of vertices within a 2-hop distance from v are affected and need update. For any neighbor u of v, the scores of up to $O(d_u)$ neighbors of u are recomputed and reflected in the data structure, costing $O(d_u |L| \log |V|)$ time. As u can be a neighbor for at most d_u vertices, the total score updates across all peeling steps require $O(\sum_{v \in V} d_v^2 |L| \log |V|)$ time. Overall, the time complexity of Algorithm 1 in handling this case is $O(\sum_{v \in V} d_v^2 |L| \log |V|)$.

C. Case 2 ($1 \leq p \leq q$)

This subsection focuses on the situation $q \geq p \geq 1$. Since $p \geq 1$, the subset $S \subseteq V$ maximizing $\rho_{q,p}(S)$ also maximizes $f_{q,p}(S)$ (Eq. (3)). Unfortunately, the function $g_{q,p}(\cdot)$ is no longer supermodular in this case, and using the score function $\Delta_{q,p}(v, S)$ defined in Eq. (4) in Algorithm 1 cannot achieve the same approximation guarantee as stated in Theorem 1.

Despite this, we will demonstrate that by deriving an upper bound of $\Delta_{q,p}(v, S)$ as the score function, Algorithm 1 can still yield a dense ML subgraph with a guaranteed approximation. Let $\boldsymbol{\delta}_{u,v}$ denote an $|L|$-dimensional 0-1 vector where the ith component is 1 if and only if the vertices u and v are adjacent on layer i. We can express $\Delta_{q,p}(v, S)$ using norms:

$$\begin{aligned}
&\Delta_{q,p}(v, S) \\
=&\|\mathbf{d}_v(S)\|_q^p + \sum_{u \in N_v(S)} \|\mathbf{d}_u(S)\|_q^p - \|\mathbf{d}_u(S) - \boldsymbol{\delta}_{u,v}\|_q^p \\
\leq&\|\mathbf{d}_v(S)\|_q^p + \sum_{u \in N_v(S)} p\|\mathbf{d}_u(S)\|_q^{p-1} \|\boldsymbol{\delta}_{u,v}\|_q,
\end{aligned}$$

where the inequality is derived from the following Proposition 4 and the triangle inequality of q-norms (Proposition 2).

Proposition 4. *For $p \geq 1$ and $y \geq x \geq 0$, it holds that $p(y - x)x^{p-1} \leq y^p - x^p \leq p(y - x)y^{p-1}$.*

Let

$$\hat{\Delta}_{q,p}(v, S) = \|\mathbf{d}_v(S)\|_q^p + \sum_{u \in N_v(S)} p\|\mathbf{d}_u(S)\|_q^{p-1} \|\boldsymbol{\delta}_{u,v}\|_q. \quad (5)$$

For $q, p \geq 1$, we have $\Delta_{q,p}(v, S) \leq \hat{\Delta}_{q,p}(v, S)$, with the equality when $q = p = 1$. Choosing $\hat{\Delta}_{q,p}(v, S)$ as the score function is due to its monotonicity:

Lemma 4. *When $p \geq 1$, for all $R \subseteq T \subseteq V$, $\hat{\Delta}_{q,p}(v, R) \leq \hat{\Delta}_{q,p}(v, T)$ for all $v \in R$.*

Proof. See Appendix E of the full paper [36]. □

Theorem 3. *When $q \geq p \geq 1$, if Algorithm 1 uses $\hat{\Delta}_{q,p}(v, S)$ as the score function, it returns a $\left(p|L|^{1-1/q} + 1\right)^{1/p}$-approximation to the optimal solution.*

Proof. As $p \geq 1$, a c-approximation to the problem of maximizing $f_{q,p}(\cdot)$ is directly a $c^{1/p}$-approximation to the problem of maximizing $\rho_{q,p}(\cdot)$. We rewrite $f_{q,p}(S)$ with norms:

$$f_{q,p}(S) = \frac{1}{|S|} \sum_{v \in S} \|\mathbf{d}_v(S)\|_q^p$$

Let S^* be the optimal solution that maximizes $f_{q,p}(S)$ for $S \subseteq V$. Certainly, S^* maximizes $\rho_{q,p}(S)$. The optimality of S^* ensures that removing any vertex v from S^* will result in a reduction of $f_{q,p}(\cdot)$:

$$\frac{\sum_{u \in S^*} \|\mathbf{d}_u(S^*)\|_q^p}{|S^*|} \geq \frac{\left(\sum_{u \in S^*} \|\mathbf{d}_u(S^*)\|_q^p\right) - \Delta_{q,p}(v, S^*)}{|S^*| - 1}.$$

By simple mathematics, we have $\Delta_{q,p}(v, S^*) \geq f_{q,p}(S^*)$.

Suppose v is the first vertex in S^* removed by Algorithm 1 and $T \subseteq V$ is the vertex subset obtained during the peeling process just before v is removed. It is obvious that $S^* \subseteq T$. By Lemma 4, we have $\hat{\Delta}_{q,p}(v, S^*) \leq \hat{\Delta}_{q,p}(v, T)$. Moreover, since v is greedily selected in T, v must minimize $\hat{\Delta}_{q,p}(u, T)$ among all vertices $u \in T$. Therefore,

$$\begin{aligned}
f_{q,p}(S^*) &\leq \Delta_{q,p}(v, S^*) \leq \hat{\Delta}_{q,p}(v, S^*) \leq \hat{\Delta}_{q,p}(v, T) \\
&\leq \frac{1}{|T|} \sum_{u \in T} \hat{\Delta}_{q,p}(u, T). \quad (6)
\end{aligned}$$

The right-hand side of the inequality can be expanded to

$$\begin{aligned}
&\frac{1}{|T|} \sum_{u \in T} \hat{\Delta}_{q,p}(u, T) \\
=&\frac{1}{|T|} \sum_{u \in T} \left(\|\mathbf{d}_u(T)\|_q^p + \sum_{w \in N_u(T)} p\|\mathbf{d}_w(T)\|_q^{p-1} \left(\|\boldsymbol{\delta}_{w,u}\|_q\right) \right) \\
=&f_{q,p}(T) + \frac{1}{|T|} \sum_{u \in T} \sum_{w \in N_u(T)} p\|\mathbf{d}_w(T)\|_q^{p-1} \|\boldsymbol{\delta}_{w,u}\|_q) \\
=&f_{q,p}(T) + \frac{1}{|T|} \sum_{w \in T} p\|\mathbf{d}_w(T)\|_q^{p-1} \sum_{u \in N_w(T)} \|\boldsymbol{\delta}_{w,u}\|_q.
\end{aligned}$$

As $\|\boldsymbol{\delta}_{w,u}\|_q$ is strictly decreasing as q increases, we have

$$\sum_{u \in N_w(T)} \|\boldsymbol{\delta}_{w,u}\|_q \leq \sum_{u \in N_w(T)} \|\boldsymbol{\delta}_{w,u}\|_1 = \sum_{u \in N_w(T)} \sum_{l \in L} \boldsymbol{\delta}_{w,u}[l]$$

$$=\sum_{l\in L}\sum_{u\in N_w(T,l)} 1=\sum_{l\in L} d_w(T,l)=\|\mathbf{d}_w(T)\|_1$$

$$\leq |L|^{1-\frac{1}{q}}\|\mathbf{d}_w(T)\|_q, \tag{7}$$

where the last inequality follows from Proposition 3. Then,

$$\begin{aligned}
&\frac{1}{|T|}\sum_{u\in T}\hat{\Delta}_{q,p}(u,T)\\
\leq& f_{q,p}(T)+\frac{1}{|T|}\sum_{w\in T} p\|\mathbf{d}_w(T)\|_q^{p-1}\left(|L|^{1-\frac{1}{q}}\|\mathbf{d}_w(T)\|_q\right)\\
=& f_{q,p}(T)+\frac{1}{|T|}\sum_{w\in T} p|L|^{1-\frac{1}{q}}\|\mathbf{d}_w(T)\|_q^{p}\\
=& f_{q,p}(T)+p|L|^{1-\frac{1}{q}}f_{q,p}(T)=\left(1+p|L|^{1-\frac{1}{q}}\right)f_{q,p}(T). \quad (8)
\end{aligned}$$

Consequently, $f_{q,p}(S^*)\leq\left(1+p|L|^{1-\frac{1}{q}}\right)f_{q,p}(T)$, and therefore, $\rho_{q,p}(S^*)\leq\left(1+p|L|^{1-\frac{1}{q}}\right)^{1/p}\rho_{q,p}(T)$. As T is in the sequence of vertex subsets greedily generated by Algorithm 1, a $\left(1+p|L|^{1-\frac{1}{q}}\right)^{1/p}$-approximation to S^* must be returned. □

Notably, the proof of Theorem 3 does not rely on $q\geq p$. Therefore, using $\hat{\Delta}_{q,p}(v,S)$ as the score function in Algorithm 1 also provides a $\left(1+p|L|^{1-\frac{1}{q}}\right)^{1/p}$-approximation for the situation $p\geq q\geq 1$ (Case 1). However, this algorithm is suboptimal compared with the $(p+1)^{1/p}$-approximation algorithm specifically designed for Case 1 in Section IV-B.

Time Complexity: Similar to Case 1 where $p\geq q\geq 1$, removing a vertex will result in changes to the scores of vertices within its 2-hop neighborhood. Thus, Algorithm 1 incurs the same time cost $O(\sum_{v\in V} d_v(V)^2|L|\log|V|)$ to handle Case 2 as it deals with Case 1.

D. Case 3 ($p=-\infty$)

This subsection studies the situation $p=-\infty$. In this situation, $\rho_{q,p}(S)$ is exactly the minimum q-mean degree of the vertices in $\mathcal{G}[S]$, that is, $\rho_{q,p}(S)=\min_{v\in S} d_v^q(S)$. Inspired by the algorithm for identifying the densest subgraph maximizing the minimum degree in simple graphs, which iteratively removes the vertex with the minimum degree, also known as the process of computing the nonempty k-core with the maximum integer k [34], we use the q-mean degree of a vertex v in $\mathcal{G}[S]$ as the score function in Algorithm 1. Since $d_v^q(S)=|L|^{-1/q}\|\mathbf{d}_v(S)\|_q$ for all $v\in S$, we can simply use $\|\mathbf{d}_v(S)\|_q$ as the score function.

Theorem 4. *When $p=-\infty$, Algorithm 1 using $\|\mathbf{d}_v(S)\|_q$ as the score function returns the optimal solution.*

Proof. Let $S^*\subseteq V$ induce the optimal solution $\mathcal{G}[S^*]$. Suppose v is the first vertex in S^* removed by Algorithm 1 during the peeling process. Let $T\subseteq V$ be the vertex subset obtained in Algorithm 1 just before v is removed. It is clear that $S^*\subseteq T$ and $\mathbf{d}_v(S^*)$ is element-wise no larger than $\mathbf{d}_v(T)$. Therefore, $\|\mathbf{d}_v(S^*)\|_q\leq\|\mathbf{d}_v(T)\|_q$.

Since v is greedily selected from T, v must have the minimum score $\|\mathbf{d}_v(T)\|_q$ among all vertices $u\in T$. Thus, we have $\min_{u\in T}\|\mathbf{d}_u(T)\|_q=\|\mathbf{d}_v(T)\|_q\geq\|\mathbf{d}_v(S^*)\|_q=\min_{u\in S^*}\|\mathbf{d}_u(S^*)\|_q$. It follows that

$$\begin{aligned}
\rho_{q,p}(T)=&\min_{u\in T}|L|^{-\frac{1}{q}}\|\mathbf{d}_u(T)\|_q\geq\\
&\min_{u\in S^*}|L|^{-\frac{1}{q}}\|\mathbf{d}_u(S^*)\|_q=\rho_{q,p}(S^*).
\end{aligned}$$

Since S^* induces the optimal solution, we have $\rho_{q,p}(S^*)\geq\rho_{q,p}(T)$. Therefore, $\rho_{q,p}(S^*)=\rho_{q,p}(T)$. Of course, $\mathcal{G}[T]$ is returned by Algorithm 1. Thus, the theorem holds. □

Time Complexity: Unlike previous cases, when adopting $\|\mathbf{d}_v(S)\|_q$ as the score function, removing a vertex solely changes the scores of its immediate neighbors. For a removed vertex v, recomputing the scores of v's neighbors and reflecting the changes in the set or heap structure takes $O(d_v(|L|+\log|S|))$ time. Therefore, the entire peeling process runs in $O(\sum_{v\in V} d_v(|L|+\log|S|)=O(|L||E|+|E|\log|V|)$ time. For the extreme case when $q=-\infty$, $\|\mathbf{d}_v(S)\|_q$ becomes an integer. It allows for a linear heap [37] to efficiently maintain vertex scores, with insertion and update operations costing amortized $O(1)$ time. Consequently, the peeling process runs in $O(|L|(|V|+|E|))$ time.

E. Case 4 ($-\infty<p\leq 1\leq q$)

This subsection investigates the situation $-\infty<p\leq 1\leq q$. In simple graphs, the densest subgraph maximizing the p-density for $p=-\infty$ serves as a $1/2$-approximation to the densest subgraph that maximizes the p-density for $p=1$ [28], [38]. Inspired by this idea, we will show that the algorithm proposed for Case 3 ($p=-\infty$) in Section IV-D achieves a satisfactory approximation ratio in the situation $q\geq p=1$, and then generalize this theoretical result to the situation $q\geq 1\geq p>-\infty$.

Theorem 5. *When $q\geq p=1$, if $\|\mathbf{d}_v(S)\|_q$ is used as the score function in Algorithm 1, the subgraph returned by Algorithm 1 is a $\left(|L|^{1-\frac{1}{q}}+1\right)$-approximation to the optimal solution.*

Proof. Below, we streamline the shared deduction process in the proofs of Theorem 3 and Theorem 4. As $p=1$, $f_{q,p}(S)$ can be simplified to

$$f_{q,p}(S)=\frac{1}{|S|}\sum_{v\in S}\|\mathbf{d}_v(S)\|_q.$$

Let $S\subseteq V$ induce the optimal solution $\mathcal{G}[S]$ that maximizes $f_{q,1}(S)$. By the same deduction process as in the proof of Theorem 3, we have $\Delta_{q,p}(v,S)\geq f_{q,1}(S)$. As $p=1$,

$$\begin{aligned}
&\Delta_{q,p}(v,S)\\
=&\|\mathbf{d}_v(S)\|_q+\sum_{u\in N_v(S)}\|\mathbf{d}_u(S)\|_q-\|\mathbf{d}_u(S)-\boldsymbol{\delta}_{u,v}\|_q\\
\leq&\|\mathbf{d}_v(S)\|_q+\sum_{u\in N_v(S)}\|\boldsymbol{\delta}_{u,v}\|_q\\
\leq&\|\mathbf{d}_v(S)\|_q+|L|^{1-\frac{1}{q}}\|\mathbf{d}_v(S)\|_q\\
=&\left(1+|L|^{1-\frac{1}{q}}\right)\|\mathbf{d}_v(S)\|_q,
\end{aligned}$$

where the two inequalities follow from the triangle inequality of q-norms (Proposition 2) and Eq. (7) established in the proof of Theorem 3.

Suppose v is the first vertex in S removed by Algorithm 1 and $T \subseteq V$ is the vertex subset obtained during the peeling process before v is removed. We have $S \subseteq T$ and $\|\mathbf{d}_v(S)\|_q \leq \|\mathbf{d}_v(T)\|_q$. The relationship between $f_{q,1}(T)$ and $f_{q,1}(S)$ is derived as follows:

$$\begin{aligned} f_{q,1}(T) &= \frac{\sum_{u\in T}\|\mathbf{d}_u(T)\|_q}{|T|} \geq \|\mathbf{d}_v(T)\|_q \geq \|\mathbf{d}_v(S)\|_q \\ &\geq \frac{\Delta_{q,p}(v,S)}{1+|L|^{1-\frac{1}{q}}} \geq \frac{f_{q,1}(S)}{1+|L|^{1-\frac{1}{q}}}. \end{aligned}$$

Consequently, we have $\rho_{q,1}(T) \geq \left(1+|L|^{1-\frac{1}{q}}\right)^{-1}\rho_{q,1}(S)$. Since T is in the sequence of vertex subsets obtained by Algorithm 1, the result returned by Algorithm 1 is a $\left(1+|L|^{1-\frac{1}{q}}\right)$-approximation to the optimal solution $\mathcal{G}[S]$. □

The proof of Theorem 5 yields the following lemma as a byproduct, which lays the foundation for deriving guaranteed approximations for $-\infty < p < 1 \leq q$.

Lemma 5. *Let $S^*_{q,p} \subseteq V$ be the subset that maximizes $\rho_{q,p}(S)$ for $S \subseteq V$. It holds that $\rho_{q,1}(S^*_{q,1}) \leq \left(1+|L|^{1-1/q}\right)\rho_{q,-\infty}(S^*_{q,-\infty})$ when $q \geq 1$.*

Proof. See Appendix F of the full paper [36]. □

Based on Lemma 5, we establish an approximation algorithm for the situation $-\infty < p < 1 \leq q$.

Theorem 6. *When $-\infty < p < 1 \leq q$, Algorithm 1 using $\|\mathbf{d}_v(S)\|_q$ as the score function returns a $\left(|L|^{1-\frac{1}{q}}+1\right)$-approximation to the optimal solution.*

Proof. Let $S^*_{q,p} \subseteq V$ be the subset that maximizes $\rho_{q,p}(S)$ for $S \subseteq V$, and let $S_{q,p}$ be the set of vertices in the subgraph returned by Algorithm 1. In this case, when q is fixed, Algorithm 1 carries out the same peeling process regardless of p because the score function $\|\mathbf{d}_v(S)\|_q$ is independent of p. For $p < 1$ and $q \geq 1$, since $S_{q,p}$ is returned as the result, we have $\rho_{q,p}(S_{q,p}) \geq \rho_{q,p}(S_{q,-\infty})$. Then,

$$\rho_{q,p}(S_{q,p}) \geq \rho_{q,p}(S_{q,-\infty}) \geq \rho_{q,-\infty}(S_{q,-\infty}) = \rho_{q,-\infty}(S^*_{q,-\infty}),$$

where the second inequality follows from the monotonicity of $\rho_{q,p}(\cdot)$ described in Lemma 1, and the last equality is due to Theorem 4 (the algorithm returns the optimal solution when $p = -\infty$). Moreover,

$$\begin{aligned} \rho_{q,p}(S^*_{q,p}) \leq \rho_{q,1}(S^*_{q,p}) &\leq \rho_{q,1}(S^*_{q,1}) \\ &\leq \left(1+|L|^{1-\frac{1}{q}}\right)\rho_{q,-\infty}(S^*_{q,-\infty}), \end{aligned}$$

where the first inequality follows from Lemma 1, the second inequality is due to the optimality of $S^*_{q,1}$, and the last inequality is due to Lemma 5. Finally, we have

$$\rho_{q,p}(S^*_{q,p}) \leq \left(1+|L|^{1-\frac{1}{q}}\right)\rho_{q,p}(S_{q,p}).$$

Thus, the theorem holds. □

Time Complexity: Algorithm 1 employs the same score function $\|\mathbf{d}_v(S)\|_q$ for this case as for Case 3 ($p = -\infty$), yielding the same peeling process and certainly the same time complexity of $O(|L||E| + |E|\log|V|)$ for both cases.

F. Common Cases and Unexplored Case

Common Cases: As illustrated in Fig. 2, the situations studied in the previous subsections have overlappings. In this subsection, we will demonstrate the approximation performance of the proposed algorithms in these common situations.

(1) Case 1, Case 2, and Case 4 intersect at $q = p = 1$. In this situation,

$$\begin{aligned} \Delta_{q,p}(v,S) = \hat{\Delta}_{q,p}(v,S) &= \|\mathbf{d}_v(S)\|_1 + \sum_{u\in N_v(S)}\|\boldsymbol{\delta}_{u,v}\|_1 \\ &= \|\mathbf{d}_v(S)\|_1 + \sum_{l\in L} d_v(S,l) = 2\|\mathbf{d}_v(S)\|_1. \end{aligned}$$

for all $S \subseteq V$ and all $v \in S$. It implies that the score functions $\Delta_{q,p}(v,S)$, $\hat{\Delta}_{q,p}(v,S)$, and $\|\mathbf{d}_v(S)\|_q$ used in these cases are consistent, and they will lead to the same peeling process and therefore the same result. Theorems 1, 3, and 5 show that the algorithms designed for Case 1, Case 2, and Case 4 all have the approximation ratio of 2 in this situation.

(2) Case 1 and Case 2 also intersect at $q = p > 1$. As discussed at the end of Section IV-C, using $\hat{\Delta}_{q,p}(v,S)$ as the score function leads to an approximation ratio of $\left(1+p|L|^{1-\frac{1}{q}}\right)^{1/p}$, while using $\Delta_{q,p}(v,S)$ as the score function results in a better approximation ratio of $(p+1)^{1/p}$ because $|L|^{1-1/q} > 1$ for $|L| > 1$ and $q > 1$.

(3) Case 2 and Case 4 also intersect at $q > 1$ and $p = 1$. Theorems 3 and 5 show that using the score functions $\hat{\Delta}_{q,p}(v,S)$ and $\|\mathbf{d}_v(S)\|_q$ both lead to the approximation ratio of $(|L|^{1-1/q}+1)$ in this situation.

Unexplored Case: Designing efficient algorithms with provable approximation guarantees for $p > -\infty$ and $q < 1$ remains an open problem. This is mainly because the triangle inequality of q-norms (Proposition 2) which underpins the approximation guarantees for Case 1 and Case 2 does not hold for $q < 1$. Moreover, the key relationship between the densities $\rho_{q,1}(S^*_{q,1})$ and $\rho_{q,-\infty}(S^*_{q,-\infty})$ to establish the approximation guarantee for Case 4 (as described in Lemma 5) also fails in this situation.

V. A Faster Approximation Algorithm

As discussed in Section IV-B and Section IV-C, during the peeling process of Algorithm 1 for handling Case 1 and Case 2 ($q, p \geq 1$), the removal of a vertex $v \in V$ affects the scores of vertices within a 2-hop distance from v. Updating these scores incurs substantial computational overhead, especially when the ML graph is dense. In this section, we extend the fast greedy peeling process for the densest subgraph problem maximizing the p-density [28] proposed by Chekuri and Torres [39] and introduce a more efficient implementation for $p, q \geq 1$ at the expense of an insignificant loss in the result accuracy.

Recalling Section IV-C, we demonstrated that when $p, q \geq 1$, utilizing $\hat{\Delta}q,p(v,S)$ as the score function leads to a

Algorithm 2 Fast implementation for MGDS

```
Input: An ML graph $\mathcal{G} = (V, E, L)$ and two parameters $q, p \geq 1$.
Output: The subgraph $\mathcal{G}[S]$ where $S \subseteq V$ that maximizes the $(q, p)$-density $\rho_{q,p}(S)$.
 1: $S_1 \leftarrow V; k \leftarrow 1$
 2: for $v \in V$ do
 3:     $s_v \leftarrow \hat{\Delta}_{q,p}(v, V)$                ▷ The approximate score value of $v$
 4:     $\hat{d}_v \leftarrow \|\mathbf{d}_v(V)\|_q$        ▷ The approximate $q$-norm of $v$'s degree vector
 5: for $i \leftarrow 1$ to $|V|$ do
 6:     $v \leftarrow \arg\min_{v \in S_i} s_v$
 7:     $S_{i+1} \leftarrow S_i - \{v\}$
 8:     if $\rho_{q,p}(S_i) > \rho_{q,p}(S_k)$ then
 9:         $k \leftarrow i$
        /* Update approximate scores of vertices */
10:     for $u \in N_v(S_i)$ do
11:         $s_u \leftarrow s_u - \left(\|\mathbf{d}_u(S_i)\|_q^p - \|\mathbf{d}_u(S_{i+1})\|_q^p\right)$    ▷ Update the first term
12:         $s_u \leftarrow s_u - p\|\hat{d}_v\|_q^{p-1}\|\boldsymbol{\delta}_{u,v}\|_q$             ▷ Update the second term
13:         if $\hat{d}_u > \left(1 + \frac{\epsilon}{p-1}\right)\|\mathbf{d}_u(S_{i+1})\|_q$ then
14:             for $w \in N_u(S_{i+1})$ do
15:                 $s_w \leftarrow s_w - p\left(\|\mathbf{d}_u(S_i)\|_q^{p-1} - \|\mathbf{d}_u(S_{i+1})\|_q^{p-1}\right)(\|\boldsymbol{\delta}_{w,u}\|_q)$
16:             $\hat{d}_u \leftarrow \|\mathbf{d}_u(S_{i+1})\|_q$
17: return $\mathcal{G}[S_k]$
```

$\left(1 + p|L|^{1-\frac{1}{q}}\right)^{1/p}$-approximation to the optimal solution. Building upon this, our efficient implementation employs a "lazy" approach to update the scores of vertices while ensuring a $(1+2\epsilon)$-approximation to the score for each vertex, denoted as $\widetilde{\Delta}_{q,p}(v, S)$, where $\epsilon \in [0, 1]$.

Theorem 7. *When $p, q \geq 1$, Algorithm 1 using $\widetilde{\Delta}_{q,p}(v, S)$ as the score function returns a $\left((1+2\epsilon)\left(1 + p|L|^{1-1/q}\right)\right)^{1/p}$-approximation to the optimal solution.*

Proof. The proof follows by slightly adapting the proof of Theorem 3. See Appendix F of the full paper [36]. □

Now, the challenge lies in maintaining an approximate score for each vertex with minimal updates. Recall that in Eq. (5), the score $\hat{\Delta}_{q,p}(v, S)$ of a vertex v with respect to a set S has two terms: the first reflects v's contribution to $g_{q,p}(S)$, while the second term bounds the decreases of the contributions of v's neighbors to $g_{q,p}(S)$ when v is removed. In fact, direct changes to v's degrees affect the first term, whereas changes to the neighbors' degrees impact the second term. We adopt the following "lazy" update strategy: changes to v's degrees are immediately reflected in the first term, while the second term is updated only when the degree of some neighbor u undergoes a substantial change. We here use the q-norm of u's degree vector to characterize the overall degree of u across all layers. By maintaining a $\left(1 + \frac{\epsilon}{p-1}\right)$-approximation of $\|\mathbf{d}_u(S)\|_q$, we update the second term if this approximate value changes. We present the following lemma to ensure that $\widetilde{\Delta}_{q,p}(v, S) \leq (1 + 2\epsilon)\hat{\Delta}_{q,p}(v, S)$.

Lemma 6. *When $p \geq 1$, it holds that $(\gamma\|\mathbf{d}_u(S)\|_q)^{p-1} \leq (1+2\epsilon)\|\mathbf{d}_u(S)\|_q^{p-1}$, where $\epsilon \in [0, 1]$ and $\gamma = 1 + \frac{\epsilon}{p-1}$.*

Proof. See Appendix G of the full paper [36]. □

Algorithm 2 outlines this efficient implementation. Theorem 7 and Lemma 6 ensure the correctness of Algorithm 2.

TABLE III: The statistics of ML graphs used in experiments.

Dataset	$\|V\|$	$\|E\|$	$\|L\|$	$\max_{v \in V} d_v$
FAO	214	268,339	364	189
Sacchcere	6,570	247,152	7	3,254
Homo	18,190	153,922	7	9,724
Amazon	410,236	8,132,506	4	3,260
Higgs	456,631	13,706,153	4	70,357
Friendfeed	505,104	18,673,521	3	75,071
DBLP-small	513,627	1,015,808	10	360
DBLP-large	1,824,674	10,169,845	22	2,119
ObamaInIsrael	2,279,535	3,827,964	3	37,241
StackOverflow	2,584,164	38,125,417	24	44,065

Time Complexity: The time overhead of Algorithm 2 is primarily caused by the for loop in lines 14–15, which updates scores of the 2-hop neighbors of the removed vertex. For any vertex u, lines 14–15 are triggered at most $O(\log_{1+\frac{\epsilon}{p-1}}\|\mathbf{d}_u(V)\|_q)$ times, with each run using $O(d_u|L|\log|V|)$ time. Consequently, the total time overhead for running lines 14–15, also the time complexity of Algorithm 2, is $O(\sum_{u \in V} d_u|L|\log|V| \cdot \log_{1+\frac{\epsilon}{p-1}}|L||V|) = O\left(\frac{|E|\log|V|\log|L||V|}{\log(1+\frac{\epsilon}{p-1})}\right)$.

VI. Experiments

A. Experimental Setup

1) Algorithms: We compare our vertex-oriented density $\rho_{q,p}(\cdot)$ with two representative and advanced layer-oriented density measures: the multilayer density $\delta_{\beta}^{MLD}(\cdot)$ and the p-mean multilayer density $\delta_{\beta,p}^{MLD}(\cdot)$. The algorithms designed for maximizing these density measures are evaluated: **(1)** **`MGDS`**: the algorithm (Algorithm 1) for finding the generalized densest ML subgraph under our density $\rho_{q,p}(\cdot)$. **`MGDS-C`**x represents the version of **`MGDS`** proposed to solve Case x of the problem, where $x \in \{1, 2, 3, 4\}$. **(2)** **`MGDS-Fast`**: the fast implementation of **`MGDS`** (Algorithm 2) for Case 1 and Case 2. **`MGDS-Fast-C`**x represents the version of **`MGDS-Fast`** for Case x, where $x \in \{1, 2\}$. **(3)** **`FC`**: the FirmCore-based algorithm [26] for finding the densest ML subgraph maximizing the multilayer density $\delta_{\beta}^{MLD}(\cdot)$. **(4)** **`GFC`**: the generalized-FirmCore-based algorithm [25] for finding the densest ML subgraph under the density $\delta_{\beta,p}^{MLD}(\cdot)$.

2) Datasets: The experiments were conducted on 10 real-world ML graphs, with the statistics listed in Table III. The FAO dataset is sourced from [40]. The datasets DBLP-large and StackOverflow are obtained from the KONECT Project [41] and are transformed into ML graphs by organizing edges into different layers according to their timestamps. The other ML graphs are obtained from [24].

3) Environments: The experiments were conducted on a server with an Intel Xeon Gold 5218R processor and 754GB of RAM, running 64-bit Ubuntu 22.04. All the algorithms were implemented in C++ and compiled with GCC 9.4.0, utilizing `-O3` optimization.

B. Effectiveness Evaluation

1) **Characteristics of the Generalized Densest ML Subgraphs:** When q is fixed and p increases, we observe a

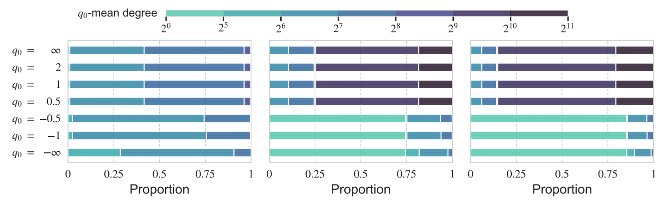
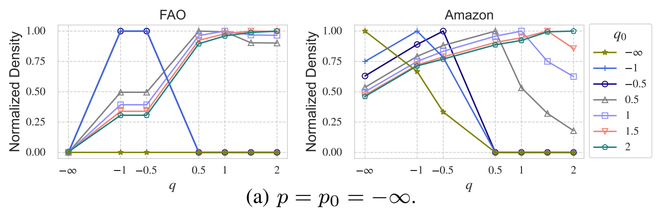
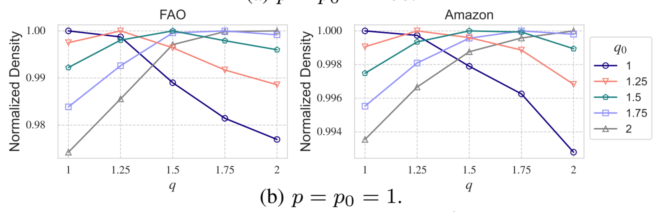

(a) $q = -\infty$. (b) $q = 1$. (c) $q = 2$.

Fig. 4: The distribution of the q_0-mean degrees of vertices in the subgraph output by **MGDS** on Friendfeed for $p = -\infty$.


(a) $p = p_0 = -\infty$.


(b) $p = p_0 = 1$.

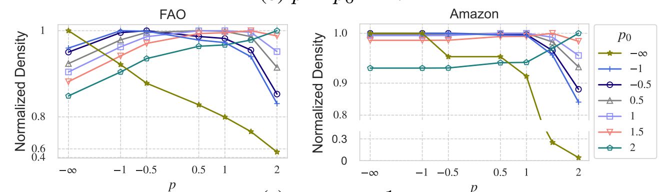

(c) $q = q_0 = 1$.

Fig. 5: Comparison between the results output by **MGDS** under different settings of q and p on FAO and Amazon in terms of the (q_0, p_0)-densities for various q_0 and p_0.

consistent increase in size and decrease in edge density of the ML subgraphs output by **MGDS**. This aligns with the findings reported in [28] on the densest subgraph maximizing the p-density in a single-layer graph.

We next examine the dense ML subgraphs produced by **MGDS** when fixing p and varying q. Fig. 4 depicts the distribution of the q_0-mean degrees of vertices in the subgraph obtained for $p = -\infty$ and $q \in \{-\infty, 1, 2\}$ on Friendfeed. We observe a correlation between q and q_0: as q increases (or decreases resp.), the resulting subgraph encompasses more vertices with high q_0-mean degrees for larger (or smaller resp.) q_0. This fully illustrates how q adjusts the preference on the vertex degrees across layers. When adopting a larger q, the q-mean degree captures higher degrees in the degree vector, and the resulting subgraph tends to contain vertices with sufficiently large degrees on certain layers, which exhibit high q_0-mean degrees for $q_0 \geq 0.5$. However, due to the skewness in vertex degrees across layers, these vertices may exhibit very low degrees on other layers, which we consider noisy or insignificant for those vertices. Consequently, the q_0-mean degrees drop significantly when q_0 decreases from 0.5 to -0.5. When $q = -\infty$, the resulting subgraph covers vertices with sufficient minimum degrees across layers, which shows high q_0-mean degrees for small q_0.

2) **Performance of the MGDS Framework:** To demonstrate the effectiveness of our MGDS framework, we conduct a comparative analysis of the results obtained under different settings of q and p and evaluate their quality with various (q_0, p_0)-densities. As illustrated in Fig. 5, each line representing a generalized density measure typically displays a distinct pattern. It suggests that our MGDS framework is able to uncover meaningful dense ML subgraphs with diverse characteristics. Furthermore, we see consistent trends from the figure: when varying q (or p), the peak density for each (q, p_0)-density (or (q_0, p)-density) measure occurs at $q = q_0$ (or $p = p_0$), and the density shows monotonicity on both sides of this point. It verifies that applying the MGDS framework for different q and p optimizes different regimes of the generalized densest ML subgraph problem.

3) **Comparison with the Layer-oriented Densest ML Subgraphs:** We begin by reporting the densities of the subgraphs output by **MGDS**, **FC**, and **GFC** on Sacchcere for $p = q = 1$ and $\beta = 2$ in Table IV. Unsurprisingly, each dense subgraph competes with the others in terms of the density measure it optimizes. From a layer-oriented perspective, the multilayer density $\delta_2^{MLD}(\cdot)$ of the subgraph identified by **FC** is determined by the four layers with the maximum average degree densities (shown in bold), while the remaining layers are considered noisy and ignored. The subgraph output by **MGDS** exhibits comparable density in each layer selected by **FC**. It suggests the effectiveness of our vertex-oriented density in handling noisy layers. The subgraph output by **GFC** has relatively lower per-layer densities. It is due to its emphasis on maximizing the 2-mean of the degrees of vertices, which may lead to the inclusion of neighbors with low degrees. We offer another vertex-oriented-perspective comparison between these subgraphs. Due to space limitations, we include it in Appendix H2 of the full paper [36].

TABLE IV: The densities of the subgraphs output by **MGDS**, **FC**, and **GFC** on Sacchcere.

Method	Per-layer Density	$\rho_{1,1}(\cdot)$	$\delta_2^{MLD}(\cdot)$	$\delta_{2,1}^{MLD}(\cdot)$
MGDS	$[\mathbf{19.3}, \mathbf{26.1}, 7.4, \mathbf{38.9}, 0.9, 0.7, \mathbf{76.1}]$	**24.2**	308.2	517.4
FC	$[\mathbf{23.9}, \mathbf{23.8}, 11.2, \mathbf{27.4}, 1.2, 0.8, \mathbf{59.8}]$	21.1	**380.8**	528.9
GFC	$[\mathbf{20.1}, \mathbf{14.5}, 10.6, \mathbf{14.7}, 0.7, 0.5, \mathbf{39.6}]$	14.4	264.3	**568.3**

We then assess the average degree of these subgraphs, defined as the ratio of the total number of edges to the product of the number of vertices and layers in the subgraph. Our results are reported in Fig. 6. We can see that the average degree of the subgraph output by **MGDC** is slightly higher than that identified by **FC**. We attribute this to the adoption of the q-mean degree in **MGDS**, which aggregates the vertex degree information from every layer, while **FC** disregards information it deems insignificant. The advantage of **MGDC** is notable on graphs DBLP-small and DBLP-large. In these graphs, **MGDC** extracts subgraphs with exceptionally high single-layer

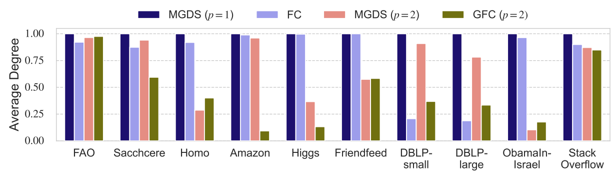


Fig. 6: The average degrees of subgraphs obtained by **MGDS**, **FC**, and **GFC**.

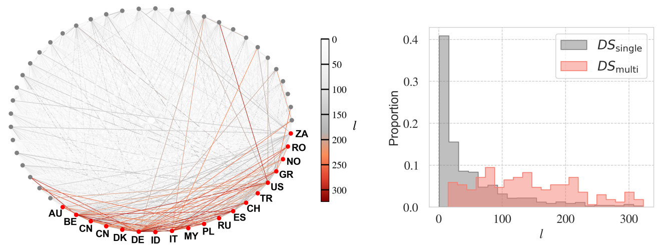


Fig. 7: Comparison between the densest subgraphs identified from FAO and its flattened single-layer version.

densities of 37.2 and 143.0, respectively. In contrast, **FC** identifies subgraphs with average densities of 0.79 and 2.5 on selected layers. This finding suggests that the vertex-oriented density is preferable when considering overall vertex degrees across layers, while layer-oriented densities may be preferred when the vertices are required to be closely connected in specific layers. When $p = 2$, both the subgraphs found by **MGDS** and **GFC** show a decrease in the average degree across most graphs. This is because they prioritize vertices with large q-mean degrees or degrees in selected layers, neglecting the inclusion of low-degree neighbors.

4) **A Case Study on FAO:** The ML graph FAO describes the worldwide food import/export relationships among countries or their subdivisions in 2010, with each layer representing transactions of a specific food product [40]. We apply our MGDS framework to the graph with $q = 1$ and $p = -\infty$ and obtain a dense ML subgraph, denoted as DS_{multi}, containing 37 vertices. Among these, 31 (83.8%) match the top-37 regions with the largest trade value in 2010 [42]. Furthermore, we extract the densest subgraph DS_{single} maximizing the p-density of the flattened single-layer version of graph FAO where vertices are connected if they are neighbors in at least one layer. DS_{single} contains a substantial number of 105 vertices covering all vertices in DS_{multi}. We depict it in Fig. 7, with the vertices included in DS_{multi} highlighted in red and labeled with the standard region code. Edge colors represent the number l of layers where the edge appears. We see that edges occur on a large number of layers, representing transactions of a diverse range of products, often connecting vertices in DS_{multi}. This diversity reflects prosperous trade transactions and higher trade values. On the right side of Fig. 7, we present the detailed distribution of l. S_{single} contains a large proportion of edges present in only a few layers. This suggests that the dense subgraph extracted from the flattened single-layer graph, which combines information from different layers, may not accurately reflect meaningful results.

C. Efficiency Evaluation

In the efficiency tests, we report the results obtained under representative settings of q and p for each solved case. Without otherwise stated, we use $\epsilon = 0.2$ for **MGDS-Fast**.

1) **Efficiency of the Densest ML Subgraph Detection Algorithms:** We compare the running time of our MGDS framework with layer-oriented densest ML subgraph detection algorithms. Fig. 8 depicts the results, where **MGDS-C3-min** corresponds to the extreme case of Case 3 for $q = p = -\infty$. We make the following observations:

(1) The greedy peeling framework exhibits distinct running time in dealing with the four studied cases. **MGDS-C1** and **MGDS-C2** takes notably longer running time than **MGDS-C3** and **MGDS-C4**, especially on denser graphs like Higgs and Friendfeed. This is primarily due to Case 1 and Case 2 requiring the update of scores for a 2-hop neighborhood of the removed vertex during the peeling process. Benefiting from the linear-heap-based implementation, **MGDS-C3-min** consistently achieves the shortest running time.

(2) **MGDS-Fast** significantly outperforms **MGDS**, achieving an average speedup of 134.6$\times$ for Case 1 and 86.3$\times$ for Case 2. The only exception is observed on graph FAO, where **MGDS-C2** runs faster than **MGDS-Fast-C2**. We attribute this to the large number of layers in graph FAO, which increases the computational cost for maintaining the approximate q-norm of vertices' degree vectors.

(3) **MGDS-C1** and **GFC** shows comparable running time. Akin to **MGDS-C1**, **GFC** adopt a peeling process to compute the generalized FirmCore, which also involves updating the information of the 2-hop neighbors of the removed vertex. However, **GFC** becomes notably slower than **MGDS-C1** on graphs with a lot of layers, such as FAO and StackOverflow. This happens because **GFC** executes the peeling process as many times as the number of layers, while **MGDS-C1** performs just a single peeling process. A similar situation can be observed for **MGDS-C3** (**MGDS-C4**) and **FC**.

2) **Effectiveness of the Fast Implementation:** Fig. 9 reports the running time and (normalized) generalized density of the output of **MGDS** and **MGDS-Fast**. **MGDS-Fast-C**x with $\epsilon = 0.1$ achieves a substantial speedup over **MGDS-C**x, up to 217.4$\times$ and 301.1$\times$ on the tested graphs for cases $x = 1$ and 2, respectively. As ϵ increases from 0.1 to 0.5, the efficiency improvements become insignificant or remain consistent. Regarding the quality of the results, except for graph DBLP-Large, where the generalized density outputted by **MGDS-Fast-C1** drops to under 0.85 of that obtained by **MGDS-C1**, the density degradation on other graphs is no more than 0.05. Moreover, it's surprising that **MGDS-Fast-C2** consistently produces the same results as **MGDS-C2** when increasing ϵ from 0.1 to 0.5. These observations indicate the effectiveness of **MGDS-Fast** and its potential to replace **MGDS** in practical scenarios.

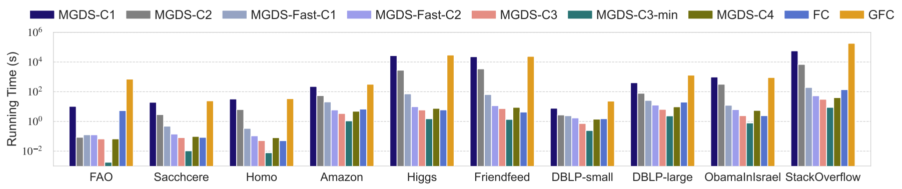


Fig. 8: Running time of different densest ML subgraph detection algorithms.

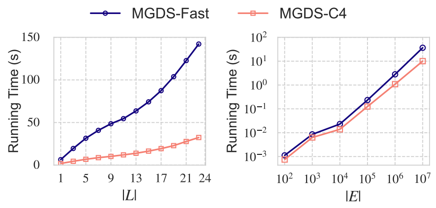
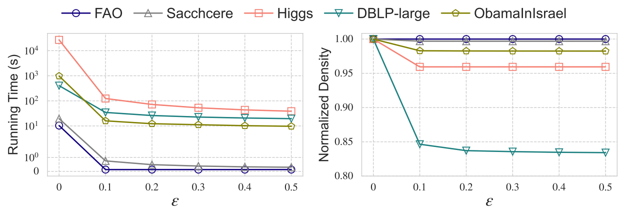


Fig. 10: The running time of **MGDS-Fast** and **MGDS-C4** on StackOverflow.


(a) **MGDS-C1** V.S. **MGDS-Fast-C1**

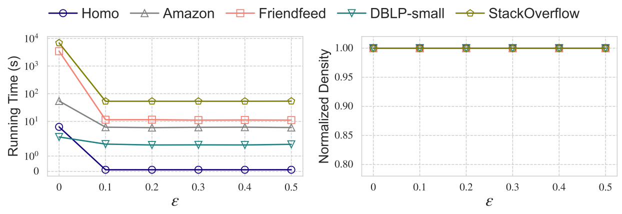

(b) **MGDS-C2** V.S. **MGDS-Fast-C2**

Fig. 9: Comparison between **MGDS** and **MGDS-Fast** for varying ϵ, where $\epsilon = 0$ corresponds to **MGDS**.

3) **Scalability:** We test the running time of **MGDS-Fast** and **MGDS-C4** using different versions of graph StackOverflow obtained by selecting a variable number of layers and subsets of edges. For **MGDS-Fast**, we set $q = p = 2$, which caters to both Case 1 and Case 2. Additionally, since **MGDS-C3** and **MGDS-C4** share the same peeling process for a fixed q, we opt for the slightly slower **MGDS-C4** as a representative. We report the results in Fig. 10. There is a clear trend that both algorithms scale gracefully with the increasing number of layers or edges, confirming their abilities to handle large-scale ML graphs. Notably, both **FC** and **GFC** for finding layer-oriented densest ML subgraphs scale at least quadratically with the number of layers.

VII. Related Work

Densest Subgraphs in Simple Graphs: Over decades of study, researchers have accumulated a wealth of density measures, algorithms for finding the densest subgraphs, and related theoretical results [5], [8], [16], [22], [28], [33], [43]–[46]. In recent years, there has been a notable trend towards introducing general density frameworks that incorporate many existing density measures and provide solutions for finding the densest subgraphs under a family of density measures [28], [30], [45], [46].

Most of the densest subgraph problems are polynomially solvable via reductions to the maximum flow problem [22], [33], [46]–[48] or linear programming approaches [31], [47], [48]. However, these methods often become impractical when dealing with large-scale graphs. As a practical and efficient alternative, the greedy peeling framework is widely adopted to produce approximate solutions for maximizing a broad range of density measures [6], [29]–[33].

Densest Subgraphs in ML Graphs: We have made a detailed introduction to existing density measures [20], [21], [24]–[27] for ML graphs and present their limitations in Section II. There are also studies exploring finding the densest subgraphs in temporal networks [49] and uncertain graphs [50], which can be viewed as ML graphs with edges labeled by timestamps and probabilities, respectively. For further details, please see [16].

Connections with Cohesive Subgraphs: Cohesive subgraph models like k-core [51], k-truss [52], k-nucleus [53], and cliques [54] impose constraints on relationships between basic graph components of the subgraph like vertices, edges, and triangles. Most cohesive subgraphs demonstrate a hierarchical structure [55] and are typically dense, yet they do not guarantee the maximum density. The k-core model shows close connections with the densest subgraphs. In simple graphs, the k-core with the maximum k maximizes the minimum degree of vertices, equivalent to the subgraph maximizing the p-density for $p = -\infty$ [28], [38]. Furthermore, there is a large body of research focused on developing core-based approximations for the densest subgraphs under various density measures [24]–[26], [56] or preprocessing approaches to expedite the exact densest subgraph search [38], [46], [57].

VIII. Conclusions

Existing density measures for ML graphs can be formalized into a layer-oriented density framework, which poses challenges in devising unified, scalable, and effective solutions for identifying the densest ML subgraphs. The (q, p)-density generalizes single-layer density measures in a vertex-oriented approach. It provides flexibility in adjusting the aggregation of vertex degrees across layers and the q-mean degrees of vertices, offering a general framework for capturing dense patterns of various characteristics. The vertex-oriented generalized densest ML subgraph problem can be largely solved by the greedy peeling framework with approximation guarantees and time complexity results outlined in Table I. Extensive experiments demonstrate the effectiveness of our (q, p)-density and the proposed algorithms in uncovering diverse types of dense subgraphs within large-scale ML graphs.

References

[1] M. E. Dickison, M. Magnani, and L. Rossi, *Multilayer social networks*. Cambridge University Press, 2016.

[2] A. Aleta, S. Meloni, and Y. Moreno, "A multilayer perspective for the analysis of urban transportation systems," *Scientific reports*, vol. 7, no. 1, p. 44359, 2017.

[3] H. Hu, X. Yan, Y. Huang, J. Han, and X. J. Zhou, "Mining coherent dense subgraphs across massive biological networks for functional discovery," *Bioinformatics*, vol. 21, no. suppl_1, pp. i213–i221, 2005.

[4] Z. Hammoud and F. Kramer, "Multilayer networks: aspects, implementations, and application in biomedicine," *Big Data Analytics*, vol. 5, no. 1, p. 2, 2020.

[5] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "Fraudar: Bounding graph fraud in the face of camouflage," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 895–904.

[6] J. Jiang, Y. Li, B. He, B. Hooi, J. Chen, and J. K. Z. Kang, "Spade: a real-time fraud detection framework on evolving graphs," *Proceedings of the VLDB Endowment*, vol. 16, no. 3, pp. 461–469, 2022.

[7] A. Angel, N. Sarkas, N. Koudas, and D. Srivastava, "Dense subgraph maintenance under streaming edge weight updates for real-time story identification," *Proceedings of the VLDB Endowment*, vol. 5, no. 6, pp. 574–585, 2012.

[8] A. Gionis and C. E. Tsourakakis, "Dense subgraph discovery: Kdd 2015 tutorial," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 2313–2314.

[9] V. E. Lee, N. Ruan, R. Jin, and C. C. Aggarwal, "A survey of algorithms for dense subgraph discovery." *Managing and mining graph data*, vol. 40, pp. 303–336, 2010.

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

[11] J. Chen and Y. Saad, "Dense subgraph extraction with application to community detection," *IEEE Transactions on knowledge and data engineering*, vol. 24, no. 7, pp. 1216–1230, 2010.

[12] E. Fratkin, B. T. Naughton, D. L. Brutlag, and S. Batzoglou, "Motifcut: regulatory motifs finding with maximum density subgraphs," *Bioinformatics*, vol. 22, no. 14, pp. e150–e157, 2006.

[13] A. Gionis, F. P. Junqueira, V. Leroy, M. Serafini, and I. Weber, "Piggybacking on social networks," in *VLDB 2013-39th International Conference on Very Large Databases*, vol. 6, no. 6, 2013, pp. 409–420.

[14] B. Saha, A. Hoch, S. Khuller, L. Raschid, and X.-N. Zhang, "Dense subgraphs with restrictions and applications to gene annotation graphs," in *Research in Computational Molecular Biology: 14th Annual International Conference, RECOMB 2010, Lisbon, Portugal, April 25-28, 2010. Proceedings 14*. Springer, 2010, pp. 456–472.

[15] P. Rozenshtein, A. Anagnostopoulos, A. Gionis, and N. Tatti, "Event detection in activity networks," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2014, pp. 1176–1185.

[16] T. Lanciano, A. Miyauchi, A. Fazzone, and F. Bonchi, "A survey on the densest subgraph problem and its variants," *ACM Computing Surveys*, 2023.

[17] D. Liu and Z. Zou, "gcore: Exploring cross-layer cohesiveness in multilayer graphs," *Proceedings of the VLDB Endowment*, vol. 16, no. 11, pp. 3201–3213, 2023.

[18] J. Pei, D. Jiang, and A. Zhang, "On mining cross-graph quasi-cliques," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 2005, pp. 228–238.

[19] D. Luo, Y. Bian, Y. Yan, X. Liu, J. Huan, and X. Zhang, "Local community detection in multiple networks," in *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, 2020, pp. 266–274.

[20] V. Jethava and N. Beerenwinkel, "Finding dense subgraphs in relational graphs," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part II 15*. Springer, 2015, pp. 641–654.

[21] K. Semertzidis, E. Pitoura, E. Terzi, and P. Tsaparas, "Finding lasting dense subgraphs," *Data Mining and Knowledge Discovery*, vol. 33, pp. 1417–1445, 2019.

[22] A. V. Goldberg, "Finding a maximum density subgraph," 1984.

[23] M. Sozio and A. Gionis, "The community-search problem and how to plan a successful cocktail party," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 939–948.

[24] E. Galimberti, F. Bonchi, F. Gullo, and T. Lanciano, "Core decomposition in multilayer networks: Theory, algorithms, and applications," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 14, no. 1, pp. 1–40, 2020.

[25] A. Behrouz and F. Hashemi, "Generalized densest subgraph in multiplex networks," in *International Conference on Complex Networks and Their Applications*. Springer, 2023, pp. 49–61.

[26] F. Hashemi, A. Behrouz, and L. V. Lakshmanan, "Firmcore decomposition of multilayer networks," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1589–1600.

[27] M. Charikar, Y. Naamad, and J. Wu, "On finding dense common subgraphs," *arXiv preprint arXiv:1802.06361*, 2018.

[28] N. Veldt, A. R. Benson, and J. Kleinberg, "The generalized mean densest subgraph problem," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 1604–1614.

[29] D. Boob, Y. Gao, R. Peng, S. Sawlani, C. Tsourakakis, D. Wang, and J. Wang, "Flowless: Extracting densest subgraphs without flow computations," in *Proceedings of The Web Conference 2020*, 2020, pp. 573–583.

[30] C. Chekuri, K. Quanrud, and M. R. Torres, "Densest subgraph: Supermodularity, iterative peeling, and flow," in *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2022, pp. 1531–1555.

[31] M. Charikar, "Greedy approximation algorithms for finding dense components in a graph," in *International workshop on approximation algorithms for combinatorial optimization*. Springer, 2000, pp. 84–95.

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

[33] C. Tsourakakis, "The k-clique densest subgraph problem," in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 1122–1132.

[34] V. Batagelj and M. Zaversnik, "An o (m) algorithm for cores decomposition of networks," *arXiv preprint cs/0310049*, 2003.

[35] F. Hashemi, A. Behrouz, and L. V. Lakshmanan, "Firmcore decomposition of multilayer networks," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1589–1600.

[36] D. Liu, Z. Zou, and R.-A. Wang. (2024) Unveiling densest multilayer subgraphs via greedy peeling (technique report). [Online]. Available: https://github.com/SSSuperDan/MGDS/blob/main/MGDS.pdf

[37] L. Chang and L. Qin, *Cohesive subgraph computation over large sparse graphs: algorithms, data structures, and programming techniques*. Springer, 2018.

[38] Y. Fang, K. Yu, R. Cheng, L. V. Lakshmanan, and X. Lin, "Efficient algorithms for densest subgraph discovery," *arXiv preprint arXiv:1906.00341*, 2019.

[39] C. Chekuri and M. R. Torres, "On the generalized mean densest subgraph problem: complexity and algorithms," *arXiv preprint arXiv:2306.02172*, 2023.

[40] M. De Domenico, V. Nicosia, A. Arenas, and V. Latora, "Structural reducibility of multilayer networks," *Nature communications*, vol. 6, no. 1, p. 6864, 2015.

[41] J. Kunegis, "Konect: the koblenz network collection," in *Proceedings of the 22nd international conference on world wide web*, 2013, pp. 1343–1350.

[42] "Food Products Exports, Imports, Tariffs by country 2010 — WITS Data — wits.worldbank.org," https://wits.worldbank.org/CountryProfile/en/Country/WLD/Year/2010/TradeFlow/EXPIMP/Partner/by-country/Product/16-24_FoodProd\#, [Accessed 31-03-2024].

[43] U. Feige, D. Peleg, and G. Kortsarz, "The dense k-subgraph problem," *Algorithmica*, vol. 29, pp. 410–421, 2001.

[44] L. Qin, R.-H. Li, L. Chang, and C. Zhang, "Locally densest subgraph discovery," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 965–974.

[45] C. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarli, "Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees," in *Proceedings of the 19th ACM SIGKDD international*

conference on Knowledge discovery and data mining, 2013, pp. 104–112.

[46] Y. Xu, C. Ma, Y. Fang, and Z. Bao, "Efficient and effective algorithms for generalized densest subgraph discovery," *Proceedings of the ACM on Management of Data*, vol. 1, no. 2, pp. 1–27, 2023.

[47] Y. Kawase and A. Miyauchi, "The densest subgraph problem with a convex/concave size function," *Algorithmica*, vol. 80, pp. 3461–3480, 2018.

[48] S. Khuller and B. Saha, "On finding dense subgraphs," in *International colloquium on automata, languages, and programming*. Springer, 2009, pp. 597–608.

[49] P. Rozenshtein, F. Bonchi, A. Gionis, M. Sozio, and N. Tatti, "Finding events in temporal networks: segmentation meets densest subgraph discovery," *Knowledge and Information Systems*, vol. 62, no. 4, pp. 1611–1639, 2020.

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

[51] S. B. Seidman, "Network structure and minimum degree," *Social networks*, vol. 5, no. 3, pp. 269–287, 1983.

[52] J. Cohen, "Trusses: Cohesive subgraphs for social network analysis," *National security agency technical report*, vol. 16, no. 3.1, 2008.

[53] A. E. Sariyuce, C. Seshadhri, A. Pinar, and U. V. Catalyurek, "Finding the hierarchy of dense subgraphs using nucleus decompositions," in *Proceedings of the 24th International Conference on World Wide Web*, 2015, pp. 927–937.

[54] R. Luce and A. D. Perry, "A method of matrix analysis of group structure," 1949.

[55] Y. Zhang, L. Qin, F. Zhang, and W. Zhang, "Hierarchical decomposition of big graphs," in *2019 IEEE 35Th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 2064–2067.

[56] F. Bonchi, A. Khan, and L. Severini, "Distance-generalized core decomposition," in *proceedings of the 2019 international conference on management of data*, 2019, pp. 1006–1023.

[57] 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 *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, 2020, pp. 1051–1066.

[58] Y. T. Lee, A. Sidford, and S. C.-w. Wong, "A faster cutting plane method and its implications for combinatorial and convex optimization," in *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. IEEE, 2015, pp. 1049–1065.

APPENDIX

A. Proof of Lemma 2

Before proving Lemma 2, we first present the following proposition.

Proposition 5. *Let $p \geq 1$ and $a, b, c, d \in \mathbb{R}_{0+}$. If $a \leq b, c \leq d$, and $b + c \leq a + d$, we have $b^p + c^p \leq a^p + d^p$.*

Proof. Without loss of generality, let $b \leq c$. If either $a = b$ or $c = d$ holds, the proposition is obvious. When $a < b$ and $c < d$, let $h(x) = x^p$. The mean value theorem guarantees that there exist $x \in (a, b)$ and $y \in (c, d)$ satisfying that $h'(x) = \frac{h(b)-h(a)}{b-a}$ and $h'(y) = \frac{h(d)-h(c)}{d-c}$. As $h'(x)$ is increasing when $p \geq 1$, we have $h'(x) \leq h'(y)$, that is, $\frac{b^p-a^p}{b-a} \leq \frac{d^p-c^p}{d-c}$. Because $d - c \geq b - a$, we must have $b^p + c^p \leq a^p + d^p$. □

The proof of Lemma 2 is given as below.

Proof. By the definition of supermodular functions (Definition 4), it is sufficient to prove that for any $S, T \subseteq V$

$$g(S) + g(T) \leq g(S \cap T) + g(S \cup T). \tag{9}$$

We extend the notion of vertex degrees: Given a set $R \subseteq V$, we define $d_v(R, l) = 0$ if $v \notin R$. Then, $g(S)$ can be rewritten as

$$g(S) = \sum_{v \in V} \left(\sum_{l \in L} d_v(S, l)^q \right)^{p/q}.$$

Let E_l be the set of edges adjacent to v on the lth layer. We define

$$\begin{aligned} A_l &= \{(i,j) \in E_l | i \in S, j \in S\}, \\ B_l &= \{(i,j) \in E_l | i \in T, j \in T\}, \\ C_l &= \{(i,j) \in E_l | i \in S \cap T, j \in S \cap T\}, \\ D_l &= \{(i,j) \in E_l | i \in S \cup T, j \in S \cup T\}. \end{aligned}$$

It follows that

$$\begin{aligned} d_v(S, l) &= |A_l|, \\ d_v(T, l) &= |B_l|, \\ d_v(S \cap T, l) &= |C_l| = |A_l \cap B_l|, \\ d_v(S \cup T, l) &= |D_l| \geq |A_l \cup B_l|. \end{aligned}$$

Let $\Omega(R) = |R|^q$. When $q \geq 1$, $\Omega(\cdot)$ is supermodular. Therefore,

$$\begin{aligned} d_v(S,l)^q + d_v(T,l)^q = |A_l|^q + |B_l|^q &= \Omega(A_l) + \Omega(B_l) \\ &\leq \Omega(A_l \cap B_l) + \Omega(A_l \cup B_l) \\ &\leq |C_l|^q + |D_l|^q \\ &= d_v(S \cap T, l)^q + d_v(S \cup T, l)^q. \end{aligned}$$

By summing the above inequality for all $l \in L$, we have

$$\begin{aligned} &\sum_{l \in L} d_v(S,l)^q + \sum_{l \in L} d_v(T,l)^q \\ &\qquad \leq \sum_{l \in L} d_v(S \cap T, l)^q + \sum_{l \in L} d_v(S \cup T, l)^q. \end{aligned}$$

In addition, due to the definition of $d_v(\cdot)$ and the monotonically increasing nature of the power function $h(x) = x^q$ for $x \in \mathbb{R}_{0+}$ and $q \geq 1$, we easily have

$$\begin{aligned} \sum_{l \in L} d_v(S \cap T, l)^q \leq \sum_{l \in L} d_v(S,l)^q, \ \sum_{l \in L} d_v(T,l)^q \\ \leq \sum_{l \in L} d_v(S \cup T, l)^q. \end{aligned}$$

As $p/q \geq 1$, we can establish the following inequality by applying Proposition 5.

$$\left(\sum_{l \in L} d_v(S,l)^q \right)^{p/q} + \left(\sum_{l \in L} d_v(T,l)^q \right)^{p/q} \leq$$
$$\left(\sum_{l \in L} d_v(S \cap T, l)^q \right)^{p/q} + \left(\sum_{l \in L} d_v(S \cup T, l)^q \right)^{p/q}.$$

By summing the inequality for all $v \in V$, we have the desired result in Eq. (9). Thus, the lemma holds. □

B. An Exact Solution to Case 1

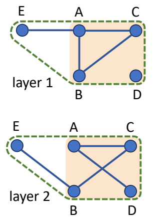
We outline a polynomial-time solution to the MGDS problem. Consider the decision version of the MGDS problem: is there a subset $S \subseteq V$ such that $f_{q,p}(S) \geq \theta$, where $\theta \geq 0$? The decision problem can be converted to the problem of finding the subset $S \subseteq V$ that maximizes

$$\Psi_{q,p}(S) = g_{q,p}(S) - \theta |S|.$$

Let $S^* = \arg\max_{S \subseteq V} \Psi_{q,p}(S)$. We have $f_{q,p}(S^*) \geq \theta$ if and only if $\Psi_{q,p}(S^*) \geq 0$. The supermodularity of $\Psi_{q,p}(\cdot)$ for $p \geq q \geq 1$ naturally follows from Lemma 2.

Lemma 7. *For $p \geq q \geq 1$, $\Psi_{q,p}(\cdot)$ is supermodular.*

Lemma 7 indicates that any polynomial-time algorithm for maximizing a supermodular objective function such as [58] can be used to find the subset $S \subseteq V$ maximizing $\Psi_{q,p}(S)$. Subsequently, the MGDS problem can be solved by performing a binary search over the range $[0, g_{q,p}(V)]$ of all possible values of θ to find the maximum θ^* of θ such that there is a subset $S^* \subseteq V$ satisfying $f_{q,p}(S^*) \geq \theta^*$. Since there are no more than $2^{|V|}$ distinct values of $f_{q,p}(S)$ for all $S \subseteq V$, a binary search returns θ^* within $|V|$ iterations. The subset S^* induces the densest subgraph.



Vertex	$S=\{A,B,C,D\}$			$S'=S-\{D\}$			$T=\{A,B,C,D,E\}$			$T'=T-\{D\}$		
	$d_v(S,1)$	$d_v(S,2)$	$\|\mathbf{d}_v(S)\|_3^2$	$d_v(S',1)$	$d_v(S',2)$	$\|\mathbf{d}_v(S')\|_3^2$	$d_v(T,1)$	$d_v(T,2)$	$\|\mathbf{d}_v(T)\|_3^2$	$d_v(T',1)$	$d_v(T',2)$	$\|\mathbf{d}_v(T')\|_3^2$
A	2	2	6.350	2	1	4.327	3	2	10.700	3	1	9.221
B	2	1	4.327	2	1	4.327	2	2	6.350	2	2	6.350
C	2	2	6.350	2	2	6.350	2	2	6.350	2	2	6.350
D	0	1	1.000	—	—		0	1	1.000	—	—	
E	—	—	—	—	—		1	1	1.587	1	1	1.587

Fig. 10: A counter-example demonstrating that the function $g_{q,p}(\cdot)$ is not supermodular in the case when $q \geq p \geq 1$.

C. Proof of Lemma 3

Proof. When $h = g_{q,p}$, c_h can be rewritten as

$$c_h = \max_{S\subseteq V} \frac{\sum_{v\in S}\Delta_{q,p}(v,S)}{g_{q,p}(S)}. \tag{10}$$

We will prove $c_h \leq p+1$ by showing that

$$\sum_{v\in S}\Delta_{q,p}(v,S) \leq (p+1)g_{q,p}(S)$$

for all $S\subseteq V$. For simplicity, let

$$E_v(S) = \sum_{l\in L} d_v(S,l)^q,$$
$$E_{v,u}(S) = \sum_{l\in L}\left(d_u(S,l)-\delta_{u,v,l}\right)^q,$$

where $\delta_{u,v,l}=1$ if v and u are adjacent on layer l, and $\delta_{u,v,l}=0$ otherwise. From the definition of $g_{q,p}(S)$ and $\Delta_{q,p}(v,S)$, we have

$$g_{q,p}(S) = \sum_{v\in S} E_v(S)^{p/q},$$
$$\Delta_{q,p}(v,S) = E_v(S)^{p/q} + \sum_{u\in N_v(S)} E_u(S)^{p/q} - E_{v,u}(S)^{p/q}.$$

Therefore,

$$\begin{aligned}
&\sum_{v\in S}\Delta_{q,p}(v,S)\\
=&\sum_{v\in S}\left(E_v(S)^{p/q} + \sum_{u\in N_v(S)} E_u(S)^{p/q} - E_{v,u}(S)^{p/q}\right)\\
=&g_{q,p}(S) + \sum_{v\in S}\sum_{u\in N_v(S)} E_u(S)^{p/q} - E_{v,u}(S)^{p/q}.
\end{aligned} \tag{11}$$

Next, we will show that the second term in Eq. (11) does not exceed $p\cdot g_{q,p}(S)$. As $q\geq 1$, the following inequality follows from Proposition 4 for any $l\in L$:

$$d_u(S,l)^q - \left(d_u(S,l)-\delta_{u,v,l}\right)^q \leq q\cdot\delta_{u,v,l}\cdot d_u(S,l)^{q-1}. \tag{12}$$

Summing Eq. (12) over all $l\in L$ establishes that

$$E_u(S) - E_{v,u}(S) \leq q\sum_{l\in L}\delta_{u,v,l}\cdot d_u(S,l)^{q-1}.$$

Because $p/q\geq 1$, it follows from Proposition 4 that

$$\begin{aligned}
&E_u(S)^{p/q} - E_{v,u}(S)^{p/q}\\
&\leq \frac{p}{q}\left(E_u(S)-E_{v,u}(S)\right)E_u(S)^{p/q-1}\\
&\leq p\left(\sum_{l\in L}\delta_{u,v,l}\cdot d_u(S,l)^{q-1}\right)E_u(S)^{p/q-1}\\
&= p\cdot E_u(S)^{p/q-1}\left(\sum_{l\in L}\delta_{u,v,l}\cdot d_u(S,l)^{q-1}\right).
\end{aligned}$$

Finally, we have

$$\begin{aligned}
&\sum_{v\in S}\sum_{u\in N_v(S)} E_u(S)^{p/q} - E_{v,u}(S)^{p/q}\\
\leq&\sum_{v\in S}\sum_{u\in N_v(S)} p\cdot E_u(S)^{p/q-1}\left(\sum_{l\in L}\delta_{u,v,l}\cdot d_u(S,l)^{q-1}\right)\\
=&p\sum_{u\in S}\sum_{v\in N_u(S)} E_u(S)^{p/q-1}\left(\sum_{l\in L}\delta_{u,v,l}\cdot d_u(S,l)^{q-1}\right)\\
=&p\sum_{u\in S}E_u(S)^{p/q-1}\sum_{l\in L}\sum_{v\in N_u(S,l)} d_u(S,l)^{q-1}\\
=&p\sum_{u\in S}E_u(S)^{p/q-1}\sum_{l\in L} d_u(S,l)^{q}\\
=&p\sum_{u\in S}E_u(S)^{p/q}\\
=&p\cdot g_{q,p}(S).
\end{aligned} \tag{13}$$

By Eq. (10), (11) and (13), we have $c_h\leq p+1$. □

D. The Function $g_{q,p}(\cdot)$ is not Supermodular When $q\geq p\geq 1$

Fig. 10 illustrates a counter-example demonstrating that $g_{q,p}(\cdot)$ is not supermodular for $q\geq p\geq 1$. Given the ML graph on the left, let $S=\{A,B,C,D\}$, $T=\{A,B,C,D,E\}$, $q=3$, and $p=2$. The key components for computing $g_{q,p}(S)$, $g_{q,p}(T)$, $g_{q,p}(S-\{D\})$, and $g_{q,p}(T-\{D\})$ are listed in the table on the right. Here, we represent $g_{q,p}(S)$ in the form of generalized norms, that is, $g_{q,p}(S)=\sum_{v\in S}\|\mathbf{d}_v(S)\|_q^p$. We have

$$g_{q,p}(S) = \sum_{v\in S}\|\mathbf{d}_v(S)\|_q^p = 18.026,$$
$$g_{q,p}(T) = \sum_{v\in T}\|\mathbf{d}_v(T)\|_q^p = 25.986,$$
$$g_{q,p}(S-\{D\}) = \sum_{v\in S-\{D\}}\|\mathbf{d}_v(S-\{D\})\|_q^p = 15.003,$$
$$g_{q,p}(T-\{D\}) = \sum_{v\in T-\{D\}}\|\mathbf{d}_v(T-\{D\})\|_q^p = 23.507.$$

Therefore, $g_{q,p}(S)-g_{q,p}(S-\{D\}) > g_{q,p}(T)-g_{q,p}(T-\{D\})$, which violates the definition of supermodular functions.

E. Proof of Lemma 4

Proof. Since $R \subseteq T$, for any vertex $v \in R$, $\mathbf{d}_v(R)$ is element-wise no larger than $\mathbf{d}_v(T)$, and thereby, $\|\mathbf{d}_v(R)\|_q \leq \|\mathbf{d}_v(T)\|_q$. Similarly, for $u \in N_v(R)$, $\|\mathbf{d}_u(R)\|_q \leq \|\mathbf{d}_u(T)\|_q$. According to Eq. (5), we easily have $\hat{\Delta}_{q,p}(v,R) \leq \hat{\Delta}_{q,p}(v,T)$, and the lemma holds. □

F. Proof of Lemma 5

Proof. Apparently, $\rho_{q,-\infty}(S) = |L|^{-1/q} \min \|\mathbf{d}_v(S)\|_q$ for $S \subseteq V$. Since $S^*_{q,-\infty}$ maximizes $\rho_{q,-\infty}(S)$ among all subsets $S \subseteq V$, we have

$$\begin{aligned}\rho_{q,-\infty}(S^*_{q,-\infty}) &\geq \max_{S\subseteq V} \rho_{q,-\infty}(S) \geq \rho_{q,-\infty}(S^*_{q,1}) \\ &= |L|^{-1/q} \min_{v\in S^*_{q,1}} \|\mathbf{d}_v(S^*_{q,1})\|_q.\end{aligned}$$

From the proof for Theorem 5, we can easily have

$$f_{q,1}(S^*_{q,1}) \leq \Delta_{q,p}(v, S^*_{q,1}) \leq \left(1+|L|^{1-\frac{1}{q}}\right) \|\mathbf{d}_v(S^*_{q,1})\|_q$$

for any $v \in S^*_{q,1}$. Then,

$$f_{q,1}(S^*_{q,1}) \leq \left(1+|L|^{1-\frac{1}{q}}\right) \min_{v\in S^*_{q,1}} \|\mathbf{d}_v(S^*_{q,1})\|_q.$$

Finally,

$$\begin{aligned}\rho_{q,1}(S^*_{q,1}) &= |L|^{-1/q} f_{q,1}(S^*_{q,1}) \\ &\leq |L|^{-1/q}\left(1+|L|^{1-\frac{1}{q}}\right) \min_{v\in S^*_{q,1}} \|\mathbf{d}_v(S^*_{q,1})\|_q \\ &\leq \left(1+|L|^{1-\frac{1}{q}}\right) \rho_{q,-\infty}(S^*_{q,-\infty}).\end{aligned}$$

Consequently, the lemma holds. □

Proof. The theorem can be proved by slightly modifying the proof for Theorem 3. Specifically, we maintain the inequality

$$\hat{\Delta}_{q,p}(v,S) \leq \widetilde{\Delta}_{q,p}(v,S) \leq (1+2\epsilon)\hat{\Delta}_{q,p}(v,S). \tag{14}$$

Then, Eq. 6 can be refined as follows:

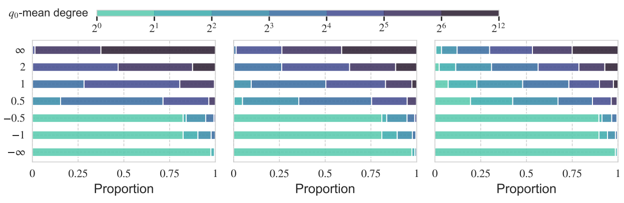
$$\begin{aligned}f_{q,p}(S^*) &\leq \hat{\Delta}_{q,p}(v,S) \leq \hat{\Delta}_{q,p}(v,T) \\ &\leq (1+2\epsilon)\frac{1}{|T|}\sum_{u\in T} \hat{\Delta}_{q,p}(u,T),\end{aligned}$$

where v is greedily selected in T that minimizes $\widetilde{\Delta}q,p(v,T)$. The last inequality holds because

$$\begin{aligned}\hat{\Delta}_{q,p}(v,T) \leq \widetilde{\Delta}_{q,p}(v,T) &\leq \frac{1}{|T|}\sum_{u\in T}\widetilde{\Delta}_{q,p}(u,T) \\ &\leq (1+2\epsilon)\frac{1}{|T|}\sum_{u\in T}\hat{\Delta}_{q,p}(u,T).\end{aligned}$$

TABLE V: Settings for the Efficiency Tests.

Case	**Tested Settings**
Case 1 ($1 \leq q \leq p$)	$q=1,\ p=2$
Case 2 ($1 \leq p \leq q$)	$q=2,\ p=1$
Case 3 ($p=-\infty$)	$q=1,\ p=-\infty$
Case 4 ($p \leq 1 \leq q$)	$q=1,\ p=-1$



(a) **MGDS**. (b) **FC**.

(c) **GFC**.

Fig. 11: Distribution of the q_0-mean degrees of vertices in the subgraphs output by **MGDS**, **FC**, and **GFC** on Sacchcere.

Further, according to Eq. 8, we have

$$f_{q,p}(S^*) \leq (1+2\epsilon)\left(1+p|L|^{1-\frac{1}{q}}\right) f_{q,p}(T),$$

and therefore

$$\rho_{q,p}(S^*) \leq \left((1+2\epsilon)\left(1+p|L|^{1-\frac{1}{q}}\right)\right)^{1/p} \rho_{q,p}(T).$$

Thus, the theorem holds. □

G. Proof of Lemma 6

Proof. For simplicity, let $d = \|\mathbf{d}_u(S)\|_q$. Consider the function $h(x) = \frac{(\xi(x)\cdot d)^{p-1}}{e^x}$, where $x \in [0,1]$ and $\xi(x) = 1 + \frac{x}{p-1}$. We have

$$h'(x) = \frac{\left(\frac{1}{\xi(x)} - 1\right)(\xi(x)\cdot d)^{p-1}}{e^x} \leq 0$$

for all $x \in [0,1]$. Thus, $h(x)$ is decreasing, so $h(\epsilon) \leq h(0) = d^{p-1}$. It follows that $(\gamma \cdot d)^{p-1} \leq e^\epsilon d^{p-1} \leq (1+2\epsilon)d^{p-1}$, where the last inequality is due to the fact that $e^\epsilon \leq 1+2\epsilon$. Thus, the lemma holds. □

H. Supplements to the Experiments

1) Settings for the Efficiency Tests. : Table V outlines the settings used in the efficiency tests.

2) A Supplementary Comparison between Vertex-oriented and Layer-oriented Densities: Following the settings in Section VI-B3, we present the distributions of the q_0-mean degrees of vertices in the dense subgraphs output by **MGDS**, **FC**, and **GFC** on graph Sacchcere in Fig. 11. The subgraphs obtained by **MGDS** and **FC** exhibit similarities: they prioritize vertices with high q_0-mean degrees for large q_0, while as q_0 approaches $-\infty$, the q_0-mean degrees of almost all vertices decrease to less than 5. This observation aligns with the findings in Tables IV that these two subgraphs exhibit large densities on four layers. For larger q_0 values, the q_0-mean degree captures the higher vertex degrees from these layers, while for smaller q_0 values, the lower vertex degrees from the noisy layers dominate. Additionally, the q_0-mean degrees of vertices in the subgraph obtained by **`GFC`** tend to be smaller compared to the others.