Finding Densest k-Connected Subgraphs

Francesco Bonchi[*1], David García-Soriano[†1], Atsushi Miyauchi[‡2], and Charalampos E. Tsourakakis[§1,3]

[1]ISI Foundation, Turin, Italy
[2]University of Tokyo, Tokyo, Japan
[3]Boston University, Boston, USA

July 3, 2020

Abstract

Dense subgraph discovery is an important graph-mining primitive with a variety of real-world applications. One of the most well-studied optimization problems for dense subgraph discovery is the densest subgraph problem, where given an edge-weighted undirected graph $G = (V, E, w)$, we are asked to find $S \subseteq V$ that maximizes the density $d(S)$, i.e., half the weighted average degree of the induced subgraph $G[S]$. This problem can be solved exactly in polynomial time and well-approximately in almost linear time. However, a densest subgraph has a structural drawback, namely, the subgraph may not be robust to vertex/edge failure. Indeed, a densest subgraph may not be well-connected, which implies that the subgraph may be disconnected by removing only a few vertices/edges within it. In this paper, we provide an algorithmic framework to find a dense subgraph that is well-connected in terms of vertex/edge connectivity. Specifically, we introduce the following problems: given a graph $G = (V, E, w)$ and a positive integer/real k, we are asked to find $S \subseteq V$ that maximizes the density $d(S)$ under the constraint that $G[S]$ is k-vertex/edge-connected. For both problems, we propose polynomial-time (bicriteria and ordinary) approximation algorithms, using classic Mader's theorem in graph theory and its extensions.

[*]francesco.bonchi@isi.it
[†]d.garcia.soriano@isi.it
[‡]miyauchi@mist.i.u-tokyo.ac.jp
[§]tsourolampis@gmail.com

1 Introduction

Dense subgraph discovery is an important graph-mining primitive with a variety of real-world applications [20]. Examples include detecting communities and spam link farms in the Web graph [11,19], extracting molecular complexes in protein–protein interaction networks [3, 45], finding experts in crowdsourcing systems [29], and real-time story identification from tweets [2].

One of the most well-studied optimization problems for dense subgraph discovery is the *densest subgraph problem*. Let $G = (V, E, w)$ be a simple undirected graph with edge weight $w : E \to \mathbb{R}_{>0}$, where $\mathbb{R}_{>0}$ is the set of positive reals. Throughout this paper, we assume that $|E| \geq 1$, edge-weighted graphs have only positive weights, and G is connected. For $S \subseteq V$, let $G[S]$ denote the subgraph induced by S, i.e., $G[S] = (S, E(S))$, where $E(S) = \{\{u, v\} \in E \mid u, v \in S\}$. The *density* of $S \subseteq V$ is defined as $d(S) = w(S)/|S|$, where $w(S)$ is the sum of edge weights of $G[S]$, i.e., $w(S) = \sum_{e \in E(S)} w(e)$. In the densest subgraph problem, given a graph $G = (V, E, w)$, we are asked to find $S \subseteq V$ that maximizes $d(S)$. An optimal solution to this problem is referred to as a *densest subgraph*.

Unlike most optimization problems for dense subgraph discovery such as the maximum clique problem [18], the densest subgraph problem is polynomial-time solvable. Indeed, there are some polynomial-time exact algorithms such as Goldberg's flow-based algorithm [21] and Charikar's linear-programming-based algorithm [8]. Moreover, it was shown by Charikar [8] that a simple greedy algorithm admits 1/2-approximation in almost linear time. Partially due to its solvability, the densest subgraph problem has been employed in many real-world applications.

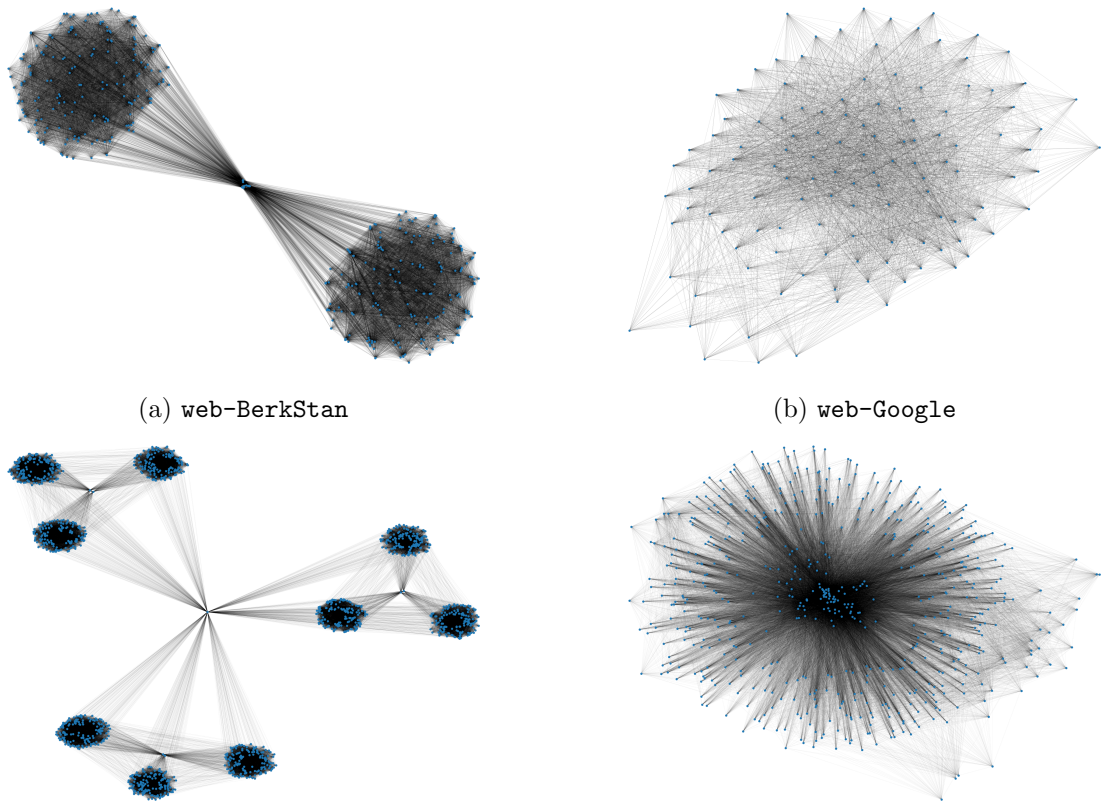
However, it can be seen that a densest subgraph has a structural drawback, that is, the subgraph may not be robust to vertex/edge failure. To see this, let us introduce some terminology. A vertex subset $S \subset V$ is called a *vertex separator* of G if its removal disconnects G, i.e., partitions G into at least two non-empty graphs between which there are no edges. Note that no clique has a vertex separator. An edge subset $F \subseteq E$ is called a *cut* of G if its removal disconnects G. The *weight* of a cut is defined to be the sum of weights of edges within it. The *vertex connectivity* of G, denoted by $\kappa(G)$, is the smallest cardinality of a vertex separator of G if G is not a clique and $|V|-1$ otherwise. The *edge connectivity* of G, denoted by $\lambda(G)$, is the smallest weight of a cut of G.

A densest subgraph does not necessarily have large vertex/edge connectivity, which means that the subgraph may be disconnected by removing only a few vertices/edges within it. For instance, consider an unweighted graph G (i.e., $w(e) = 1$ for every $e \in E$) consisting of two equally-sized large cliques that share only a few vertices or are connected by only a few edges. In both cases, the entire graph is a densest subgraph, but it is easily disconnected by removing the common vertices in the former case and the bridging edges in the latter case.

In this paper, we provide an algorithmic framework to find a dense subgraph that is well-connected in terms of vertex/edge connectivity. An (edge-weighted) graph G is said to be *k-vertex-connected* if $\kappa(G)$ is no less than k. On the other hand, an edge-weighted graph G is said to be *k-edge-connected* if $\lambda(G)$ is no less than k. Using these criteria, we introduce the following problems:

Problem 1 (Densest k-vertex-connected subgraph)**.** *Given an edge-weighted undirected graph $G = (V, E, w)$, where $w : E \to \mathbb{R}_{>0}$, and a positive integer $k \in \mathbb{Z}_{>0}$, the goal is to find $S \subseteq V$ that maximizes the density $d(S)$ subject to the constraint that the induced subgraph $G[S]$ is k-vertex-connected.*

Problem 2 (Densest k-edge-connected subgraph)**.** *Given an edge-weighted undirected graph $G = (V, E, w)$, where $w : E \to \mathbb{R}_{>0}$, and a positive real $k \in \mathbb{R}_{>0}$, the goal is to find $S \subseteq V$ that maximizes*



(a) `web-BerkStan` (b) `web-Google`

(c) `web-NotreDame` (d) `web-Stanford`

Figure 1: Densest subgraphs in real-world Web graphs.

the density $d(S)$ subject to the constraint that the induced subgraph $G[S]$ is k-edge-connected.

In the two-cliques example we discussed earlier, an optimal solution to Problems 1 and 2 with a sufficiently large value for k would be one of the cliques, which is robust to vertex/edge failure and nearly as dense as the densest subgraph (i.e., the entire graph). We observe that Problems 1 and 2 are meaningful for real-world data too; Figure 1 visualizes densest subgraphs of the four real-world Web graphs that are publicly available at SNAP (Stanford Network Analysis Project) [32] using a spring layout positioning.[1] As we can visually observe, small separators may exist in real-world densest subgraphs. Table 1 summarizes the detailed statistics. As can be seen, the densest subgraphs in `web-BerkStan` and `web-NotreDame` have surprisingly small vertex connectivity; in fact, they have vertex connectivity of twelve and one, respectively. Note that for both densest subgraphs, vertex connectivity is much smaller than the minimum degree of vertices, a trivial upper bound on that.

For both problems, we propose polynomial-time (bicriteria and ordinary) approximation algorithms. Let $w_{\max}$ and $w_{\min}$ denote the maximum and minimum weights, respectively, over all edges in G, i.e., $w_{\max} = \max_{e \in E} w(e)$ and $w_{\min} = \min_{e \in E} w(e)$.

[1]Graphs have been made simple undirected by ignoring the direction of edges, and by removing self-loops and multiple edges.

Table 1: Details of densest subgraphs $G[S^{\mathrm{DS}}]$ in four real-world Web graphs: $\delta(G[S^{\mathrm{DS}}])$ represents the minimum degree of vertices in $G[S^{\mathrm{DS}}]$, a trivial upper bound on $\kappa(G[S^{\mathrm{DS}}])$ and $\lambda(G[S^{\mathrm{DS}}])$.

Graph	$\lvert S^{\mathrm{DS}}\rvert$	$\lvert E(S^{\mathrm{DS}})\rvert$	$d(S)$	$\kappa(G[S^{\mathrm{DS}}])$	$\lambda(G[S^{\mathrm{DS}}])$	$\delta(G[S^{\mathrm{DS}}])$
`web-BerkStan`	392	40,535	103.41	12	201	201
`web-Google`	123	3,449	28.04	30	30	30
`web-NotreDame`	1,367	107,526	78.66	1	155	155
`web-Stanford`	597	35,456	59.39	60	60	60

Our first result is polynomial-time $\left(\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}}, 1/\gamma\right)$-bicriteria approximation algorithms with parameter $\gamma \in [1, 2]$ for Problems 1 and 2. That is, the algorithm for Problem 1/Problem 2 outputs $S \subseteq V$ having density at least the optimal value times $\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}}$ but only satisfies a (k/γ)-vertex/edge-connectivity constraint (rather than the original k-vertex/edge-connectivity constraint). Note that if we set $\gamma = 1$, we can obtain $\left(\frac{1}{4} \cdot \frac{w_{\min}}{w_{\max}}\right)$-approximation algorithms. The design of our algorithms is based on an elegant theorem in graph theory, proved by Mader [34]. The theorem states that any (unweighted) dense graph contains a highly vertex-connected subgraph wherein the minimum degree of vertices is greater than the density of the entire vertex set. We refer to this subgraph as a *Mader subgraph* and our algorithm finds a Mader subgraph of a densest subgraph of each maximal k-vertex-connected subgraph of G. It should be noted that to deal with edge-weighted graphs, we generalize Mader's theorem. Our generalized version cannot be directly obtained from the original statement of Mader's theorem, and is essential to derive the bicriteria approximation ratio for edge-weighted graphs.

Our second result is polynomial-time $\left(\frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}}\right)$-approximation algorithms for Problems 1 and 2, which improves the above approximation ratio of $\frac{1}{4} \cdot \frac{w_{\min}}{w_{\max}}$ derived directly from the bicriteria approximation ratio. Our algorithm for Problem 1/Problem 2 computes the most highly connected subgraph in terms of vertex/edge connectivity, which can be done using the algorithms in Matula [38]. In the analysis of the approximation ratio, we use a useful variant of Mader's theorem, recently proved by Bernshteyn and Kostochka [5].

Paper organization. The remainder of this paper is organized as follows. In Section 2, we review related work. In Section 3, we extend Mader's theorem to edge-weighted graphs and design an algorithm for finding a Mader subgraph. In Sections 4 and 5, we present our bicriteria and ordinary approximation algorithms, respectively. We conclude with some open problems in Section 6.

2 Related Work

Variations of the densest subgraph problem. Wu et al. [52] consider the problem of detecting a dense and connected subgraph in *dual networks*. A dual network is a pair of graphs $G = (V, E_G)$ and $H = (V, E_H)$ defined on the same vertex set V, which encode different types of connections using two edge sets E_G and E_H. Wu et al. [52] introduced the following problem: given a dual network (G, H), we are asked to find $S \subseteq V$ that maximizes $d(S)$ in G under the constraint that $H[S]$ is connected (i.e., 1-edge-connected). They proved that the problem is NP-hard and devised a scalable heuristic. Problem 2 with $k = 1$, i.e., the densest 1-edge-connected subgraph,

on unweighted graphs, can be seen as a special case of their problem wherein two graphs G and H are the same, i.e., $E_G = E_H$. It is easy to see that unlike the general form of their problem, the densest 1-edge-connected subgraph problem (on unweighted graphs) is polynomial-time solvable.

Two closely related papers are due to Tsourakakis [48] and Kawase and Miyauchi [30]. They aim to find a near-clique (which is robust to vertex/edge failure) by extending the densest subgraph problem. Tsourakakis [48] introduced the problem called the *k-clique densest subgraph problem*. In this problem, given an unweighted graph $G = (V, E)$, we are asked to find $S \subseteq V$ that maximizes the *k-clique density* $w_k(S)/|S|$, where $w_k(S)$ is the number of k-cliques (i.e., cliques with size k) in $G[S]$. Tsourakakis [48] showed that this problem (with constant k) remains polynomial-time solvable, and later, Mitzenmacher et al. [39] proposed a scalable algorithm that obtains a nearly-optimal solution. On the other hand, Kawase and Miyauchi [30] introduced the problem called the *f-densest subgraph problem with convex f*. In this problem, given an edge-weighted graph $G = (V, E, w)$, we are asked to find $S \subseteq V$ that maximizes $w(S)/f(|S|)$, where $f : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{\geq 0}$ is a monotonically non-decreasing function that satisfies $(f(x+2) - f(x+1)) - (f(x+1) - f(x)) \geq 0$ for any $x \in \mathbb{Z}_{\geq 0}$. This formulation generalizes the NP-hard optimal quasi-cliques problem due to Tsourakakis et al. [49, 50]. Kawase and Miyauchi [30] studied the hardness of the problem, and proposed a polynomial-time approximation algorithm. Although the above two problems contribute to computing a dense subgraph that is robust to vertex/edge failure, they cannot explicitly impose k-vertex/edge connectivity.

There are also some variants that take into account the robustness to the uncertainty of input graphs. Zou [53] studied the densest subgraph problem on *uncertain graphs*. Uncertain graphs are a generalization of graphs, which can model the uncertainty of the existence of edges. More formally, an uncertain graph consists of an unweighted graph $G = (V, E)$ and a function $p : E \to [0, 1]$, where $e \in E$ is present with probability $p(e)$ whereas $e \in E$ is absent with probability $1 - p(e)$. In the problem introduced by Zou [53], given an uncertain graph $G = (V, E)$ with p, we are asked to find $S \subseteq V$ that maximizes the expected value of the density. Zou [53] observed that this problem can be reduced to the original densest subgraph problem, and designed polynomial-time exact algorithm using the reduction. Very recently, Tsourakakis et al. [51] introduced the problem called the *risk-averse DSD*. In this problem, given an uncertain graph $G = (V, E)$ with p, we are asked to find $S \subseteq V$ that has a large expected density and at the same time has a small *risk*. The risk of $S \subseteq V$ is measured by the probability that S is not dense on a given uncertain graph. They showed that the risk-averse DSD can be reduced to the densest subgraph problem with *negative* edge weights (which is NP-hard), and designed an efficient approximation algorithm based on the reduction.

Miyauchi and Takeda [41] considered the uncertainty of edge weights rather than the existence of edges. To model that, they assumed that they have an *edge-weight space* $W = \times_{e \in E}[l_e, r_e] \subseteq \times_{e \in E}[0, \infty)$ that contains the unknown true edge weight w. To evaluate the performance of $S \subseteq V$ without any concrete edge weight, they employed a well-known measure in the field of robust optimization, called the *robust ratio*. In their scenario, the robust ratio of $S \subseteq V$ under W is defined as the multiplicative gap between the density of S in terms of edge weight w' and the density of $S^*_{w'}$ in terms of edge weight w' under the worst-case edge weight $w' \in W$, where $S^*_{w'}$ is a densest subgraph of G with w'. Intuitively, $S \subseteq V$ with a large robust ratio has a density close to the optimal value even on G with the edge weight selected adversarially from W. Using the robust ratio, they formulated the *robust densest subgraph problem*, where given an unweighted graph $G = (V, E)$ and an edge-weight space $W = \times_{e \in E}[l_e, r_e] \subseteq \times_{e \in E}[0, \infty)$, we are asked to find

$S \subseteq V$ that maximizes the robust ratio under W. Miyauchi and Takeda [41] designed an algorithm that returns $S \subseteq V$ with a robust ratio of at least $\frac{1}{\max_{e \in E} \frac{r_e}{l_e}}$ under some mild condition.

In addition to the above, there are many other problem variations. The most well-studied variants are size restricted ones [1, 6, 13, 31]. For example, in the *densest k-subgraph problem* [13], given an edge-weighted graph $G = (V, E, w)$ and a positive integer $k \in \mathbb{Z}_{>0}$, we are asked to find $S \subseteq V$ that maximizes $d(S)$ subject to the constraint $|S| = k$. It is known that such a restriction makes the problem much harder; indeed, the densest k-subgraph problem is NP-hard and the best known approximation ratio is $\Omega(1/n^{1/4+\epsilon})$ for any $\epsilon > 0$ [6]. The densest subgraph problem has also been extended to more general computation models and graph structures. As for computation models, to cope with the dynamics of real-world graphs, some literature has considered dynamic settings [12,26], and moreover, to model the limited computation resources in reality, some literature has considered streaming settings [2, 4, 7]. As for graph structures, the problem has been defined on hypergraphs [26, 40] and multilayer networks [17].

Vertex and edge connectivity. In the *vertex connectivity problem*, we are asked to compute $\kappa(G)$ for a given graph $G = (V, E)$. For this problem, Gabow [16] developed an $O(|V|(\kappa(G)^2 \cdot \min\{|V|^{3/4}, \kappa(G)^{3/2}\} + \kappa(G)|V|))$-time algorithm, which also computes a corresponding *minimum vertex separator* $S \subset V$. This is one of the current fastest deterministic algorithms for the problem, although there are various randomized algorithms (e.g., see [14, 24, 33, 43]). Note that there are linear-time algorithms that decide whether G is 2-vertex-connected and 3-vertex-connected, respectively, due to Tarjan [47] and Hopcroft and Tarjan [25].

Another important problem related to vertex connectivity is to compute the family of maximal k-vertex-connected subgraphs, which will be solved in our bicriteria approximation algorithm for Problem 1. For $S \subseteq V$ and $k \in \mathbb{Z}_{>0}$, the induced subgraph $G[S]$ is called a *maximal k-vertex-connected subgraph* if $G[S]$ is k-vertex-connected and no superset of S has this property. For this task, the first polynomial-time algorithm is given by Matula [37]. Note that maximal k-vertex-connected subgraphs may overlap each other; the design of the algorithm by Matula [37] is based on the fact that the maximum total number of maximal k-vertex-connected subgraphs is $O(|V|)$ [37]. Later, Makino [36] designed an $O(|V| \cdot T)$-time algorithm, where T is the computation time required to find a vertex separator of size at most $k - 1$. Combined with the above vertex connectivity algorithm by Gabow [16], the algorithm by Makino [36] yields the running time of $O(|V|^2(k^2 \cdot \min\{|V|^{3/4}, k^{3/2}\} + k|V|))$. For some special k, there are some existing algorithms that have better running time. For $k = 2$ and 3, there are linear-time algorithms by Tarjan [47] and Hopcroft and Tarjan [25], respectively. For any constant k, Henzinger et al. [22] presented an $O(|V|^3)$-time algorithm.

In the *(global) minimum cut problem*, given an edge-weighted graph $G = (V, E, w)$, we are asked to find the minimum weight cut $F \subseteq E$. For this problem, Nagamochi and Ibaraki [42] designed an $O(|V|(|E|+|V| \log |V|))$-time algorithm. Later, Stoer and Wagner [46] and Frank [15] independently presented a very simple algorithm that still has the same running time. For simple unweighted graphs, the seminal work by Karger [27] provides a randomized (Monte Carlo) algorithm that runs in nearly-linear, $O(|E| \log^3 |V|)$, time. As this algorithm does not necessarily return the right answer, Karger [27] posed an open question to find a nearly-linear-time deterministic algorithm. In a recent breakthrough, Kawarabayashi and Thorup [28] answered the question; they developed a deterministic algorithm for simple unweighted graphs that runs in $O(|E| \log^{12} |V|)$ time. Very recently, Henzinger et al. [23] improved the running time to $O(|E| \log^2 |V| \log \log^2 |V|)$ time, which

is better even than that of the randomized algorithm by Karger [27].

As in the vertex connectivity case, computing the family of maximal k-edge-connected subgraphs is also an important problem, which will be solved in our bicriteria approximation algorithm for Problem 2. For $S \subseteq V$ and $k \in \mathbb{R}_{>0}$, the induced subgraph $G[S]$ is called a *maximal k-edge-connected subgraph* if $G[S]$ is k-edge-connected and no superset of S has this property. The problem can be solved using any minimum cut algorithm as follows: if the weight of the minimum cut of the graph is less than k, divide the graph into two subgraphs along with the cut and then repeat the procedure on the resulting subgraphs. For edge-weighted graphs, we can directly obtain an $O(|V|^2(|E| + |V| \log |V|))$-time algorithm using one of the above minimum cut algorithms by Nagamochi and Ibaraki [42], Stoer and Wagner [46], and Frank [15]. To the best of our knowledge, there is no existing algorithm that has a better running time. For simple unweighted graphs, we can again directly obtain an $O(|E||V| \log^2 |V| \log \log^2 |V|)$-time algorithm using the above minimum cut algorithm by Henzinger et al. [23]. Unlike the weighted case, for some special k, there are some existing algorithms that have a better running time. For $k = 2$, there is a linear-time algorithm by Tarjan [47]. For any constant k, Henzinger et al. [22] presented an $O(|V|^2 \log |V|)$-time algorithm, and more recently, Chechik et al. [9] provided an $O(\sqrt{|V|}(|E| + |V| \log |V|))$-time algorithm. The latter algorithm is efficient particularly for sparse graphs; indeed, the latter is better than the former when $|E| = o(|V|^{3/2} \log |V|)$. Very recently, for any $k \in \mathbb{Z}_{>0}$, Forster et al. [14] developed a randomized (Las Vegas) algorithm that has expected running time $O(k^3|V|^{3/2} \log |V| + k|E| \log^2 |V|)$, which is faster than the algorithm by Chechik et al. [9] (for general $k \in \mathbb{Z}_{>0}$).

3 Mader's theorem and Mader subgraph

In this section, we extend Mader's theorem to edge-weighted graphs and design an algorithm for finding a Mader subgraph.

3.1 Mader's Theorem on Edge-Weighted Graphs

Mader's theorem [34] is a foundational theorem in graph theory. The precise statement is as follows:

Theorem 1 (Mader [34]; see also Theorem 1.4.3 in Diestel [10])**.** *Let $G = (V, E)$ be an unweighted graph and let d be a positive integer. If G has density at least d, then G has a $(\lfloor d/2 \rfloor + 1)$-vertex-connected subgraph wherein the minimum degree of vertices is greater than d.*

A straightforward application of Theorem 1 to edge-weighted graphs would yield the following result. Let $G = (V, E, w)$ be an edge-weighted graph, let d be a positive real, and assume that G has density at least d. Now consider an unweighted graph $G' = (V, E)$ defined on the same vertex set V and edge set E. As G' has the density of at least $d/w_{\max}$ (i.e., at least $\lfloor d/w_{\max} \rfloor$), by Theorem 1, we see that G' has a $\left(\left\lfloor \frac{\lfloor d/w_{\max} \rfloor}{2} \right\rfloor + 1\right)$-vertex-connected subgraph wherein the minimum degree of vertices is greater than $\lfloor d/w_{\max} \rfloor$. Therefore, we can deduce that G has a $\left(\left\lfloor \frac{\lfloor d/w_{\max} \rfloor}{2} \right\rfloor + 1\right)$-vertex-connected subgraph wherein the minimum weighted degree of vertices is greater than $w_{\min} \lfloor d/w_{\max} \rfloor$. However, this is weaker than what we need to prove the approximation guarantee of our algorithms, as we discuss in Section 4.4.

Here we provide a stronger version for edge-weighted graphs. Specifically, we prove the following theorem:

Theorem 2. *Let $G = (V, E, w)$ be an edge-weighted graph and let d be a positive real. If G has density at least d, then G has a $\left(\left\lfloor \frac{\lceil d/w_{\max} \rceil}{2} \right\rfloor + 1\right)$-vertex-connected subgraph wherein the minimum weighted degree of vertices is greater than d.*

Proof. Let $H = (S, E(S))$ be a subgraph of G with the minimum number of vertices that satisfies

$$|S| \geq \lceil d/w_{\max} \rceil \quad \text{and} \quad w(S) > d(V)\left(|S| - \frac{\lceil d/w_{\max} \rceil}{2}\right). \tag{1}$$

There exists such a subgraph H because G itself satisfies the above condition. In fact, since $d(V) \geq d$ holds, there exists a vertex with the weighted degree of at least $2d$, implying that the number of neighbors of such a vertex is at least $\lceil 2d/w_{\max} \rceil$, thus $|V| \geq \lceil 2d/w_{\max} \rceil + 1 > \lceil d/w_{\max} \rceil$ holds, and $w(V) = d(V)|V| > d(V)\left(|V| - \frac{\lceil d/w_{\max} \rceil}{2}\right)$. Suppose that $|S| = \lceil d/w_{\max} \rceil$. Then we have

$$\begin{aligned} w(S) &> d(V)\left(|S| - \frac{\lceil d/w_{\max} \rceil}{2}\right) = \frac{d(V)\lceil d/w_{\max} \rceil}{2} \\ &\geq \frac{w_{\max}(d/w_{\max})\lceil d/w_{\max} \rceil}{2} > w_{\max}\binom{\lceil d/w_{\max} \rceil}{2} = w_{\max}\binom{|S|}{2} \geq w(S), \end{aligned}$$

a contradiction. Therefore, we see that $|S| \geq \lceil d/w_{\max} \rceil + 1$. Suppose also that there exists a vertex v in H whose weighted degree is at most $d(V)$ in H. Let $H' = (S', E(S'))$ be a subgraph constructed by removing v from H. Then we have

$$\begin{aligned} &|S'| = |S| - 1 \geq \lceil d/w_{\max} \rceil \quad \text{and} \\ &w(S') \geq w(S) - d(V) > d(V)\left(|S| - \frac{\lceil d/w_{\max} \rceil}{2} - 1\right) = d(V)\left(|S'| - \frac{\lceil d/w_{\max} \rceil}{2}\right). \end{aligned}$$

This means that H' also satisfies condition (1), which contradicts the minimality of H. Therefore, we see that every vertex in H has weighted degree greater than $d(V) \geq d$ in H.

From now on, we show that H is $\left(\left\lfloor \frac{\lceil d/w_{\max} \rceil}{2} \right\rfloor + 1\right)$-vertex-connected. Suppose, for contradiction, that there exists $T \subseteq S$ with $|T| \leq \left\lfloor \frac{\lceil d/w_{\max} \rceil}{2} \right\rfloor$ whose removal separates H into two non-empty subgraphs $H[S_1]$ and $H[S_2]$ so that there are no edges between them. For any vertex $v \in S_1$, its neighbors in H are all contained in $S_1 \cup T$. As v has weighted degree greater than $d(V) \geq d$ in H, the number of neighbors of v in $S_1 \cup T$ is at least $\lceil d/w_{\max} \rceil$, thus $|S_1 \cup T| \geq \lceil d/w_{\max} \rceil + 1$. From the minimality of H, we see that the subgraph $H[S_1 \cup T]$ does not satisfy condition (1), which implies that

$$w(S_1 \cup T) \leq d(V)\left(|S_1 \cup T| - \frac{\lceil d/w_{\max} \rceil}{2}\right)$$

holds. Applying the same argument to S_2, we also have

$$w(S_2 \cup T) \leq d(V)\left(|S_2 \cup T| - \frac{\lceil d/w_{\max} \rceil}{2}\right).$$

Algorithm 1: `Peel`(G, d)

Input : $G = (V, E, w)$ and $d \in \mathbb{R}_{>0}$
Output: Subgraph of G or `Null`

```
S ← V;
while True do
    v_min ← argmin_{v∈S} deg_S(v);
    if deg_S(v_min) > d then
        return G[S];
    S ← S \ {v_min};
return Null;
```

Combining these two inequalities, we have

$$\begin{aligned} w(S) &\leq w(S_1 \cup T) + w(S_2 \cup T) \\ &\leq d(V)(|S_1 \cup T| + |S_2 \cup T| - \lceil d/w_{\max} \rceil) \\ &= d(V)(|S_1| + |T| + |S_2| + |T| - \lceil d/w_{\max} \rceil) \\ &\leq d(V) \left(|S| - \frac{\lceil d/w_{\max} \rceil}{2} \right), \end{aligned}$$

which contradicts that H satisfies condition (1). □

3.2 Algorithm for Finding a Mader Subgraph

We design an algorithm `Mader_subgraph` that extracts a Mader subgraph, i.e., the subgraph whose existence is guaranteed by Theorem 2. To this end, we first present a simple subprocedure, which we call `Peel`. For an edge-weighted graph $G = (V, E, w)$ and a positive real d, the procedure `Peel` returns the maximal subgraph of G wherein the minimum weighted degree of vertices is greater than d if such a subgraph exists and `Null` otherwise. Specifically, `Peel` iteratively removes a vertex with the minimum weighted degree in the currently remaining graph while the minimum weighted degree is no greater than d. Note that this procedure is similar to the procedure to find a *k-core*. For reference, we describe the entire procedure in Algorithm 1, where $\deg_S(v)$ for $S \subseteq V$ and $v \in S$ denotes the weighted degree of v in $G[S]$. This algorithm can be implemented to run in $O(|E| + |V| \log |V|)$ time, as mentioned in the literature [40].

Using Algorithm 1, we present `Mader_subgraph` in Algorithm 2, where the notation $V(H')$ denotes the vertex set of subgraph H' of G. Here we briefly explain the behavior of the algorithm. Let G^* be a Mader subgraph of a given edge-weighted graph G. The algorithm keeps a family of subgraphs $\mathcal{H}$ in which exactly one subgraph contains G^* as its subgraph. In each iteration, the algorithm tests whether a subgraph in $\mathcal{H}$ is a Mader subgraph or not, and if not, the algorithm divides the subgraph into strictly smaller pieces and add (a part of) them to $\mathcal{H}$. The algorithm repeats this operation until it finds a Mader subgraph. It should be noted that our algorithm is based on Matula's algorithm [38, Algorithm A], which finds the most highly connected subgraph in terms of vertex connectivity, i.e., $H \in \operatorname{argmax}\{\kappa(H) \mid H \text{ is a subgraph of } G\}$.

The following theorem verifies the validity of `Mader_subgraph`. The proof strategy is similar to that for Matula [38, Theorem 3].

Algorithm 2: `Mader_subgraph`(G)

Input : $G = (V, E, w)$
Output: Subgraph of G

$H \leftarrow$ `Peel`$(G, d(V))$;
$\tau \leftarrow \left\lfloor \frac{\lceil d(V)/w_{\max} \rceil}{2} \right\rfloor + 1$;
$\mathcal{H} \leftarrow$ the family of the maximal connected subgraphs of H that have at least $\tau + 1$ vertices;
if *there exists a clique K in $\mathcal{H}$* **then**
  return K;
while *True* **do**
  $H' \leftarrow$ an arbitrary element of $\mathcal{H}$;
  $C \leftarrow$ the minimum vertex separator of H';
  if $|C| \geq \tau$ **then**
    return H';
  $\mathcal{S} \leftarrow$ the family of the vertex sets of the maximal connected subgraphs of $G[V(H') \setminus C]$;
  $\mathcal{H}' \leftarrow \emptyset$;
  for *each $S \in \mathcal{S}$* **do**
    if *`Peel`$(G[S \cup C], d(V))$ has at least $\tau + 1$ vertices* **then**
      $\mathcal{H}' \leftarrow \mathcal{H}' \cup \{$`Peel`$(G[S \cup C], d(V))\}$;
  if *there exists a clique K in $\mathcal{H}'$* **then**
    return K;
  $\mathcal{H} \leftarrow (\mathcal{H} \setminus \{H'\}) \cup \mathcal{H}'$;

Theorem 3. *For a given edge-weighted graph $G = (V, E, w)$, Algorithm 2 outputs a Mader subgraph of G in $O(|V|^{19/4})$ time.*

Proof. It is easy to see that if the algorithm terminates, its output is a Mader subgraph of G. Thus, in what follows, we analyze the time complexity of the algorithm.

Specifically, we prove that Algorithm 2 runs in $O(|V|^{19/4})$ time. The time complexity of the algorithm except for the while-loop is given by $O(|E| + |V| \log |V|)$ due to the time complexity of the procedure `Peel`. We can show that the time complexity of the while-loop is given by $O(|V|^{19/4})$. To see this, we analyze the time complexity of each iteration and the number of iterations. The time complexity of each iteration is dominated by that required to compute the minimum vertex separator C of H'. As reviewed in Section 2, the current best algorithm completes this task in $O(|V(H')|(|C|^2 \cdot \min\{|V(H')|^{3/4}, |C|^{3/2}\} + |C||V(H')|))$ time. Hence, the time complexity of each iteration is bounded by $O(|V|^{15/4})$. Next we show that the number of iterations of the while-loop is bounded by $|V|$. Let G^* be a Mader subgraph of G, that is, G^* is a τ-vertex-connected subgraph of G wherein the minimum weighted degree of vertices is greater than $d(V)$. It is easy to see that exactly one subgraph in $\mathcal{H}$ contains G^* as its subgraph in any iteration of the while-loop. Here we define the *surplus* of $\mathcal{H}$ as

$$s(\mathcal{H}) = \sum_{H \in \mathcal{H}} (|V(H)| - \tau - 1).$$

For the initial $\mathcal{H}$, we have $s(\mathcal{H}) \leq |V| - \tau - 1$. Note that $s(\mathcal{H}) \geq 0$ holds in any iteration. Let us consider an arbitrary iteration in which the algorithm does not terminate. Let $\mathcal{S}' = \{S \in \mathcal{S} \mid |V(\texttt{Peel}(G[S \cup C], d(V)))| \geq \tau + 1\}$. If $|\mathcal{S}'| \leq 1$ holds, then H' is simply deleted or replaced by a subgraph with at most $|V(H')| - 1$ vertices, in the updated $\mathcal{H}$. Thus, the surplus decreases by at least one in the iteration. Assume that $|\mathcal{S}'| \geq 2$. Then we have

$$\begin{aligned}
\sum_{H \in \mathcal{H}'} (|V(H)| - \tau - 1) &= \sum_{S \in \mathcal{S}'} (|V(\texttt{Peel}(G[S \cup C], d(V)))| - \tau - 1) \\
&\leq \sum_{S \in \mathcal{S}'} (|V(G[S \cup C])| - \tau - 1) \\
&\leq |V(H')| + (|\mathcal{S}'| - 1)(|C| - \tau) - \tau - |\mathcal{S}'| \\
&< |V(H')| - \tau - 2,
\end{aligned}$$

where the last inequality follows from $|\mathcal{S}'| \geq 2$ and $|C| < \tau$. Note that $|C| < \tau$ holds because the algorithm has not yet terminated in the iteration. The above inequality implies that the surplus decreases by at least two in the iteration. Therefore, the number of iterations of the while-loop is bounded by $|V| - \tau < |V|$. □

4 Bicriteria Approximation Algorithms

In this section, we first design a polynomial-time $\left(\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}}, 1/\gamma\right)$-bicriteria approximation algorithm with parameter $\gamma \in [1, 2]$ for Problem 1, and then present a corresponding result for Problem 2.

4.1 Algorithm for Problem 1

For a given edge-weighted graph $G = (V, E, w)$, our algorithm first finds the family of maximal k-vertex-connected subgraphs $\{G[S_1], \ldots, G[S_p]\}$ using Makino's algorithm [36] combined with Gabow's vertex connectivity algorithm [16], which takes $O(|V|^2(k^2 \cdot \min\{|V|^{3/4}, k^{3/2}\} + k|V|))$ time. Note that if there is no k-vertex-connected subgraph found, our algorithm returns `INFEASIBLE` because the instance is actually infeasible.

For each $i = 1, \ldots, p$, the algorithm initializes S_i^* as S_i. Then the algorithm finds a densest subgraph S_i^{DS} (without any constraint) in $G[S_i]$. This can be done in polynomial time using Charikar's linear-programming-based algorithm for the densest subgraph problem [8]. After that, if $k \leq \gamma \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1\right)$ holds, then the algorithm employs as S_i^* the vertex set of a Mader subgraph of $G[S_i^{\mathrm{DS}}]$, i.e., the vertex set of a $\left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max}(G[S_i^{\mathrm{DS}}]) \rceil}{2} \right\rfloor + 1\right)$-vertex-connected subgraph in $G[S_i^{\mathrm{DS}}]$ wherein the minimum weighted degree of vertices is greater than $d(S_i^{\mathrm{DS}})$, using the procedure `Mader_subgraph` (Algorithm 2). Here $w_{\max}(G[S_i^{\mathrm{DS}}])$ denotes the maximum weight of edges in $G[S_i^{\mathrm{DS}}]$. Note that $w_{\max}(G[S_i^{\mathrm{DS}}]) \leq w_{\max}$ holds. For $G[S_i^{\mathrm{DS}}]$, `Mader_subgraph` runs in $O(|S_i^{\mathrm{DS}}|^{19/4}) = O(|V|^{19/4})$ time.

Finally, the algorithm outputs the densest subset among $\{S_1^*, \ldots, S_p^*\}$. For reference, we summarize the entire procedure in Algorithm 3. As the maximum total number of maximal k-vertex-connected subgraphs is $O(|V|)$ [37], the overall running time of Algorithm 3 is given by $O(|V|(|V|^{19/4} + T_{\mathrm{DS}}))$, where T_{DS} is the computation time required to find a densest subgraph in (any subgraph of) G. Note that as mentioned above, T_{DS} is polynomial in $|V|$ and $|E|$. Moreover,

Algorithm 3: Bicriteria approximation algorithm with parameter $\gamma \in [1, 2]$ for Problem 1

Input : $G = (V, E, w)$ and $k \in \mathbb{Z}_{>0}$
Output: $S \subseteq V$ or `INFEASIBLE`

Find the family of maximal k-vertex-connected subgraphs $\{G[S_1], \ldots, G[S_p]\}$;
if *there is no k-vertex-connected subgraph found* **then**
  return `INFEASIBLE`;
else
  for $i = 1, \ldots, p$ **do**
    $S_i^* \leftarrow S_i$;
    Find a densest subgraph S_i^{DS} (without any constraint) in $G[S_i]$;
    if $k \leq \gamma \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$ **then**
      $S_i^* \leftarrow$ The vertex set of `Mader_subgraph`$(G[S_i^{\mathrm{DS}}])$;
  return $S \in \operatorname{argmax}_{S \in \{S_1^*, \ldots, S_p^*\}} d(S)$;

for unweighted graphs, Goldberg's flow-based algorithm [21] gives $T_{\mathrm{DS}} = O(|E||V|)$, using Orlin's maximum-flow algorithm [44].

4.2 Analysis

Using our generalized Mader's theorem (Theorem 2), we provide the bicriteria approximation ratio of Algorithm 3:

Theorem 4. *For any $\gamma \in [1, 2]$, Algorithm 3 is a polynomial-time $\left(\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}}, 1/\gamma\right)$-bicriteria approximation algorithm for Problem 1.*

Proof. We first show that the output of Algorithm 3 is (k/γ)-vertex-connected. To this end, it suffices to confirm (k/γ)-vertex-connectivity of $G[S_i^*]$ for each $i = 1, \ldots, p$. Fix $i \in \{1, \ldots, p\}$. If $k \leq \gamma \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$ does not hold, we are done since $G[S_i^*]$ is given by $G[S_i]$, which is k-vertex-connected (thus (k/γ)-vertex-connected). Consider the case where $k \leq \gamma \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$ holds. Applying Theorem 2 to $G[S_i^{\mathrm{DS}}]$ with setting $d = d(S_i^{\mathrm{DS}})$, we see that $G[S_i^{\mathrm{DS}}]$ has a $\left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max}(G[S_i^{\mathrm{DS}}]) \rceil}{2} \right\rfloor + 1 \right)$-vertex-connected subgraph, which is (k/γ)-vertex-connected. Algorithm 3 employs such a subset as S_i^*.

We next analyze the first term of the bicriteria approximation ratio. It suffices to show that for each $i = 1, \ldots, p$, the subset S_i^* has density at least $\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}}$ times the optimal value of Problem 1 on $G[S_i]$. Fix $i \in \{1, \ldots, p\}$. Clearly, the optimal value of Problem 1 on $G[S_i]$, which we denote by OPT_i, is at most $d(S_i^{\mathrm{DS}})$.

We first consider the case where $k \leq \gamma \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$ does not hold. In this case, Algorithm 3 just employs S_i as S_i^*. As $G[S_i]$ is k-vertex-connected, each vertex has weighted

degree of at least $w_{\min} \cdot k > \gamma \cdot w_{\min} \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$; thus, the density of S_i is greater than

$$\gamma \cdot w_{\min} \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right) /2 \geq \gamma \cdot w_{\min} \left(\frac{d(S_i^{\mathrm{DS}})/w_{\max}}{2} - \frac{1}{2} + 1 \right) /2$$
$$> \frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}} \cdot d(S_i^{\mathrm{DS}}) \geq \frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}} \cdot \mathsf{OPT}_i,$$

which means $\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}}$-approximation.

We next consider the case where $k \leq \gamma \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$ holds. Applying Theorem 2 to $G[S_i^{\mathrm{DS}}]$ with setting $d = d(S_i^{\mathrm{DS}})$, we see that $G[S_i^{\mathrm{DS}}]$ has a $\left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max}(G[S_i^{\mathrm{DS}}]) \rceil}{2} \right\rfloor + 1 \right)$-vertex-connected subgraph wherein the minimum weighted degree of vertices is greater than $d(S_i^{\mathrm{DS}})$. Algorithm 3 employs such a subset as S_i^*. As each vertex has weighted degree greater than $d(S_i^{\mathrm{DS}})$, the density of S_i^* is greater than $d(S_i^{\mathrm{DS}})/2 \geq \mathsf{OPT}_i/2$, which means 1/2-approximation (thus $\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}}$-approximation). □

From the proof, we see that if the if-condition of Algorithm 3 holds, the output admits 1/2-approximation, irrespective of edge weights. Moreover, it should be noted that setting $\gamma = 1$ in the theorem, we can obtain an ordinary $\frac{1}{4} \cdot \frac{w_{\min}}{w_{\max}}$-approximation algorithm for Problem 1. In Section 5, we present an algorithm with a better approximation ratio.

4.3 Algorithm for Problem 2 and Analysis

Here we present a bicriteria approximation algorithm for Problem 2, which is an edge-connectivity counterpart of Algorithm 3. For a given edge-weighted graph $G = (V, E, w)$, our algorithm first finds the family of maximal k-edge-connected subgraphs $\{G[S_1], \ldots, G[S_p]\}$. As reviewed in Section 2, this can be done in $O(|V|^2(|E| + |V| \log |V|))$ time using one of the minimum cut algorithms by Nagamochi and Ibaraki [42], Stoer and Wagner [46], and Frank [15] as a subroutine. If G is simple unweighted, the time complexity reduces to $O(|E||V| \log^2 |V| \log \log^2 |V|)$ using the minimum cut algorithm by Henzinger et al. [23].

In the processing of $G[S_i]$ for each $i = 1, \ldots, p$, the algorithm computes a *variant* of a Mader subgraph of $G[S_i^{\mathrm{DS}}]$, i.e., a $w_{\min} \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$-edge-connected subgraph in $G[S_i^{\mathrm{DS}}]$ wherein the minimum weighted degree of vertices is greater than $d(S_i^{\mathrm{DS}})$. The existence of such a subgraph is guaranteed by a corollary of Theorem 2, which we will present later. Recall that Algorithm 3 uses the procedure `Mader_subgraph`. On the other hand, the above variant can be computed using the strategy employed by the algorithms for computing the family of maximal k-edge-connected subgraphs, presented in Section 2. Specifically, the strategy in our scenario is as follows: if the weight of the minimum cut of $G[S_i^{\mathrm{DS}}]$ is less than $w_{\min} \left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1$, divide the graph into two subgraphs along with the cut and then repeat the procedure on the resulting subgraphs (until it finds the variant of a Mader subgraph). It should be noted that in order to satisfy the minimum weighted degree condition, our algorithm needs to conduct the procedure `Peel` every time before it processes a new subgraph. For reference, the pseudocode of our algorithm is given in Algorithm 4.

Here we evaluate the running time of Algorithm 4. It is easy to see that the above algorithm for finding the variant of a Mader subgraph still has the same running time as that of algorithms for computing the family of maximal k-edge-connected subgraphs. Therefore, the time complexity

Algorithm 4: Bicriteria approximation algorithm with parameter $\gamma \in [1, 2]$ for Problem 2

Input : $G = (V, E, w)$ and $k \in \mathbb{R}_{>0}$
Output: $S \subseteq V$ or `INFEASIBLE`

Find the family of maximal k-edge-connected subgraphs $\{G[S_1], \ldots, G[S_p]\}$;
if *there is no k-edge-connected subgraph found* **then**
  return `INFEASIBLE`;
else
  for $i = 1, \ldots, p$ **do**
    $S_i^* \leftarrow S_i$;
    Find a densest subgraph S_i^{DS} (without any constraint) in $G[S_i]$;
    if $k \leq \gamma \cdot w_{\min} \left(\left\lfloor \frac{\lceil d(S_i^{DS})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$ **then**
      $S_i^* \leftarrow$ The vertex set of a $w_{\min} \left(\left\lfloor \frac{\lceil d(S_i^{\mathrm{DS}})/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$-edge-connected subgraph in $G[S_i^{\mathrm{DS}}]$ wherein the minimum weighted degree of vertices is greater than $d(S_i^{\mathrm{DS}})$;
  return $S \in \operatorname{argmax}_{S \in \{S_1^*, \ldots, S_p^*\}} d(S)$;

of the processing of each $G[S_i]$ is bounded by $O(T_{\mathrm{DS}}(S_i) + |S_i|^2(|E(S_i)| + |S_i| \log |S_i|))$, where $T_{\mathrm{DS}}(S_i)$ is the computation time required to find a densest subgraph in $G[S_i]$. Recalling that maximal k-edge-connected subgraphs do not overlap for any k, we see that the time complexity of the entire for-loop is bounded by $O(T_{\mathrm{DS}}(G) + |V|^2(|E| + |V| \log |V|))$, which also bounds the overall running time of Algorithm 4. For simple unweighted graphs, we have the running time of $O(|V|^3 + |E||V| \log^2 |V| \log \log^2 |V|)$.

Finally we analyze the theoretical performance guarantee of Algorithm 4. It is easy to see that any (edge-weighted) k-vertex-connected graph G is $w_{\min}k$-edge-connected, which gives the following corollary to Theorem 2:

Corollary 1. *Let $G = (V, E, w)$ be an edge-weighted graph and let d be a positive real. If G has density at least d, then G has a $w_{\min} \left(\left\lfloor \frac{\lceil d/w_{\max} \rceil}{2} \right\rfloor + 1 \right)$-edge-connected subgraph wherein the minimum weighted degree of vertices is greater than d.*

Using this corollary, we can derive the bicriteria approximation ratio of Algorithm 4:

Theorem 5. *For any $\gamma \in [1, 2]$, Algorithm 4 is a polynomial-time $\left(\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}}, 1/\gamma \right)$-bicriteria approximation algorithm for Problem 2.*

The proof is similar to that of Theorem 4, and is omitted.

4.4 Remarks on Theorem 2

Here we explain that our generalized Mader's theorem (i.e., Theorem 2) is essential to derive the bicriteria approximation ratio given in Theorems 4 and 5. To this end, recall that the straightforward application of the original Mader's theorem to edge-weighted graphs derives the following statement: Let $G = (V, E, w)$ be an edge-weighted graph and let d be a positive real. If G has

Algorithm 5: Approximation algorithm for Problem 1

Input : $G = (V, E, w)$ and $k \in \mathbb{Z}_{>0}$
Output: $S \subseteq V$ or `INFEASIBLE`

$H \leftarrow \operatorname{argmax}\{\kappa(H) \mid H \text{ is a subgraph of } G\}$;
if $\kappa(H) \geq k$ **then**
  return the vertex set of H;
else
  return `INFEASIBLE`;

density at least d, then G has a $\left(\left\lfloor \frac{\lfloor d/w_{\max} \rfloor}{2} \right\rfloor + 1\right)$-vertex-connected subgraph wherein the minimum weighted degree of vertices is greater than $w_{\min}\lfloor d/w_{\max} \rfloor$.

Obviously, the above statement is weaker than Theorem 2. Indeed, vertex connectivity of $\left\lfloor \frac{\lceil d/w_{\max} \rceil}{2} \right\rfloor + 1$ in Theorem 2 has decreased to $\left\lfloor \frac{\lfloor d/w_{\max} \rfloor}{2} \right\rfloor + 1$, which is only a slight deterioration, but the minimum weighted degree of d in Theorem 2 has significantly decreased to $w_{\min}\lfloor d/w_{\max} \rfloor$. It is easy to see that to prove Theorems 4 and 5, vertex connectivity of $\left\lfloor \frac{\lfloor d/w_{\max} \rfloor}{2} \right\rfloor + 1$ is sufficient, but the minimum weighted degree of $w_{\min}\lfloor d/w_{\max} \rfloor$ is insufficient. In fact, in the last paragraph of the proof of Theorem 4, by using the decreased minimum weighted degree, we can only guarantee that the density of S_i^* is greater than $\frac{w_{\min}\lfloor d(S_i^{\mathrm{DS}})/w_{\max} \rfloor}{2} \geq \frac{w_{\min}\lfloor \mathsf{OPT}_i/w_{\max} \rfloor}{2}$ (rather than $d(S_i^{\mathrm{DS}})/2 \geq \mathsf{OPT}_i/2$ in the proof). Note that $\frac{w_{\min}\lfloor \mathsf{OPT}_i/w_{\max} \rfloor}{2}$ may be less than $\frac{\gamma}{4} \cdot \frac{w_{\min}}{w_{\max}} \cdot \mathsf{OPT}_i$, meaning that the decreased minimum weighted degree is insufficient to prove the theorem. We can see the same issue in the proof of Theorem 5.

5 Approximation Algorithms

In this section, we design a polynomial-time $\left(\frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}}\right)$-approximation algorithm for Problem 1, which improves the approximation ratio of $\frac{1}{4} \cdot \frac{w_{\min}}{w_{\max}}$ that is immediately derived by Algorithm 3. Then we present its counterpart result for Problem 2.

5.1 Algorithm for Problem 1

Our algorithm first computes the most highly connected subgraph in terms of vertex connectivity, i.e., $H \in \operatorname{argmax}\{\kappa(H) \mid H \text{ is a subgraph of } G\}$. This can be done using Matula's algorithm [38, Algorithm A]. Then our algorithm simply returns the subgraph if its vertex connectivity is no less than k and `INFEASIBLE` otherwise. Our algorithm is described in pseudocode as Algorithm 5.

Matula [38] showed that the time complexity of the algorithm for computing the most highly connected subgraph in terms of vertex connectivity is given by $O(|V| \cdot T)$, where T is the computation time required to find a minimum vertex separator of G. If we consider Gabow's vertex connectivity algorithm [16], the time complexity becomes $O(|V|^2(\kappa(G)^2 \cdot \min\{|V|^{3/4}, \kappa(G)^{3/2}\} + \kappa(G)|V|))$. Clearly, Algorithm 5 has the same time complexity.

5.2 Analysis

From now on, we analyze the theoretical performance guarantee of Algorithm 5. To this end, we use the following theorem, which is a useful variant of Mader's theorem:

Theorem 6 (Bernshteyn and Kostochka [5])**.** *Let $G = (V, E)$ be an unweighted graph and let t be an integer with $t \geq 2$. If G satisfies $|V| \geq \frac{5}{2}t$ and $|E| > \frac{19}{12}t(|V| - t)$, then G has a $(t+1)$-vertex-connected subgraph.*

We provide the approximation ratio of Algorithm 5 in the following theorem:

Theorem 7. *Algorithm 5 is a polynomial-time $\left(\frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}}\right)$-approximation algorithm for Problem 1.*

Proof. Let $S \subseteq V$ be the output of Algorithm 5. Define

$$\kappa_{\max} = \max\{\kappa(H) \mid H \text{ is a subgraph of } G\}.$$

As we assumed that $|E| \geq 1$, we have $\kappa_{\max} \geq 1$. Recall that $H = G[S]$ is a $\kappa_{\max}$-vertex-connected subgraph. We denote by OPT the density of an optimal solution to Problem 1. Let $S_{\mathrm{DS}} \subseteq V$ be a densest subgraph (unconstrained) in G. As $d(S_{\mathrm{DS}}) \geq \mathsf{OPT}$, it suffices to show that $d(S) \geq \frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}} \cdot d(S_{\mathrm{DS}})$ holds. Let n_{DS} and m_{DS} denote the number of vertices and edges in $G[S_{\mathrm{DS}}]$, respectively.

Case I: $\kappa_{\max} = 1$**.** In this case, G is a forest; therefore, using the fact that $m_{\mathrm{DS}} \leq n_{\mathrm{DS}} - 1$, we have $d(S_{\mathrm{DS}}) = \frac{w(S_{\mathrm{DS}})}{n_{\mathrm{DS}}} \leq w_{\max} \cdot \frac{m_{\mathrm{DS}}}{n_{\mathrm{DS}}} < w_{\max}$. Any vertex subset (with size more than one) inducing a connected subgraph, including the output S, has density of at least

$$\frac{w_{\min}}{2} > \frac{6}{19} w_{\min} > \frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}} \cdot d(S_{\mathrm{DS}}).$$

Case II: $\kappa_{\max} \geq 2$**.** Let us define $t = \left\lfloor \frac{12}{19} \cdot \frac{m_{\mathrm{DS}}}{n_{\mathrm{DS}}} \right\rfloor$. As $m_{\mathrm{DS}} \leq \binom{n_{\mathrm{DS}}}{2}$ holds, we have $t < \frac{2}{5} n_{\mathrm{DS}}$, and thus $n_{\mathrm{DS}} > \frac{5}{2}t$. As for the value of m_{DS}, if $t \neq 0$, $m_{\mathrm{DS}} \geq \frac{19}{12} t n_{\mathrm{DS}} > \frac{19}{12} t(n_{\mathrm{DS}} - t)$ holds. Thus, by Theorem 6, if $t \geq 2$ holds, then the subgraph $G[S_{\mathrm{DS}}]$ has a $(t+1)$-vertex-connected subgraph, which is also a subgraph of G. Hence, we have $\kappa_{\max} \geq t + 1 \geq \frac{12}{19} \cdot \frac{m_{\mathrm{DS}}}{n_{\mathrm{DS}}}$. On the other hand, if $t < 2$ holds, then $\kappa_{\max} \geq 2 > \frac{12}{19} \cdot \frac{m_{\mathrm{DS}}}{n_{\mathrm{DS}}}$. In either case, noticing that the output S is $w_{\min}\kappa_{\max}$-edge-connected, we see that S has density at least

$$\frac{w_{\min} \cdot \kappa_{\max}}{2} \geq w_{\min} \cdot \frac{6}{19} \cdot \frac{m_{\mathrm{DS}}}{n_{\mathrm{DS}}} \geq \frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}} \cdot \frac{w(S_{\mathrm{DS}})}{n_{\mathrm{DS}}} \geq \frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}} \cdot d(S_{\mathrm{DS}}),$$

which completes the proof. □

5.3 Algorithm for Problem 2 and Analysis

Here we present an approximation algorithm for Problem 2, which is an edge-connectivity counterpart of Algorithm 5. Specifically, our algorithm first computes the most highly connected subgraph in terms of edge connectivity, i.e., $H \in \operatorname{argmax}\{\lambda(H) \mid H \text{ is a subgraph of } G\}$. This can be done using a simple recursive algorithm mentioned by Matula [38], which is similar to the algorithms for computing the family of maximal k-edge-connected subgraphs. Then our algorithm simply returns

Algorithm 6: Approximation algorithm for Problem 2

Input : $G = (V, E, w)$ and $k \in \mathbb{R}_{>0}$
Output: $S \subseteq V$ or `INFEASIBLE`

$H \leftarrow \operatorname{argmax}\{\lambda(H) \mid H \text{ is a subgraph of } G\}$;
if $\lambda(H) \geq k$ **then**
  return the vertex set of H;
else
  return `INFEASIBLE`;

the subgraph if its edge connectivity is no less than k and `INFEASIBLE` otherwise. For reference, we describe the entire procedure in Algorithm 6.

Matula [38] stated that the time complexity of the algorithm for computing the most highly connected subgraph in terms of edge connectivity is given by $O(|V| \cdot T)$, where T is the computation time required to find a minimum cut of G. If we consider one of the minimum cut algorithms by Nagamochi and Ibaraki [42], Stoer and Wagner [46], and Frank [15], the time complexity becomes $O(|V|^2(|E| + |V| \log |V|))$. If G is simple unweighted, the time complexity reduces to $O(|E||V| \log^2 |V| \log \log^2 |V|)$ using the minimum cut algorithm by Henzinger et al. [23]. Clearly, Algorithm 6 has the same time complexity.

Finally we analyze the theoretical performance guarantee of Algorithm 6. The following corollary is an edge-connectivity counterpart of Theorem 6:

Corollary 2. *Let $G = (V, E, w)$ be an edge-weighted graph and let t be an integer with $t \geq 2$. If G satisfies $|V| \geq \frac{5}{2}t$ and $|E| > \frac{19}{12}t(|V| - t)$, then G has a $w_{\min}(t + 1)$-edge-connected subgraph.*

Using this corollary, we can derive the approximation ratio of Algorithm 6:

Theorem 8. *Algorithm 6 is a polynomial-time $\left(\frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}}\right)$-approximation algorithm for Problem 2.*

The proof is similar to that of Theorem 7, and is omitted.

6 Open Problems

There are several directions for future research. The most interesting one is to design a polynomial-time algorithm that has a better (bicriteria or ordinary) approximation ratio. We wish to remark that assuming Mader's conjecture [35], which is a stronger version of Theorem 6, we can improve the approximation ratio of Algorithms 5 and 6, i.e., $\frac{6}{19} \cdot \frac{w_{\min}}{w_{\max}}$, to $\frac{1}{3} \cdot \frac{w_{\min}}{w_{\max}}$. However, Mader [35] also conjectured that the statement is best possible, making it unlikely to obtain an approximation ratio better than $\frac{1}{3} \cdot \frac{w_{\min}}{w_{\max}}$ via similar analysis. Another interesting direction is to investigate the computational complexity of Problems 1 and 2.

Acknowledgments

F.B., D.G-S., and C.T. acknowledge support from Intesa Sanpaolo Innovation Center, who had no role in study design, data collection and analysis, decision to publish, or preparation of the

manuscript. A.M. was supported by Grant-in-Aid for Research Activity Start-up (No. 17H07357) and Grant-in-Aid for Early-Career Scientists (No. 19K20218). This work was partially done while A.M. was at RIKEN AIP, Japan, and visitied ISI Foundation, Italy.

References

[1] R. Andersen and K. Chellapilla. Finding dense subgraphs with size bounds. In *WAW '09: Proceedings of the 6th Workshop on Algorithms and Models for the Web Graph*, pages 25–37, 2009.

[2] A. Angel, N. Sarkas, N. Koudas, and D. Srivastava. Dense subgraph maintenance under streaming edge weight updates for real-time story identification. In *VLDB '12: Proceedings of the 38th International Conference on Very Large Data Bases*, pages 574–585, 2012.

[3] G. D. Bader and C. W. V. Hogue. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics*, 4(1):1–27, 2003.

[4] B. Bahmani, R. Kumar, and S. Vassilvitskii. Densest subgraph in streaming and mapreduce. In *VLDB '12: Proceedings of the 38th International Conference on Very Large Data Bases*, pages 454–465, 2012.

[5] A. Bernshteyn and A. Kostochka. On the number of edges in a graph with no $(k+1)$-connected subgraphs. *Discrete Mathematics*, 339(2):682–688, 2016.

[6] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: An $O(n^{1/4})$ approximation for densest k-subgraph. In *STOC '10: Proceedings of the 42nd ACM Symposium on Theory of Computing*, pages 201–210, 2010.

[7] S. Bhattacharya, M. Henzinger, D. Nanongkai, and C. E. Tsourakakis. Space- and time-efficient algorithm for maintaining dense subgraphs on one-pass dynamic streams. In *STOC '15: Proceedings of the 47th ACM Symposium on Theory of Computing*, pages 173–182, 2015.

[8] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. In *APPROX '00: Proceedings of the 3rd International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 84–95, 2000.

[9] S. Chechik, T. D. Hansen, G. F. Italiano, V. Loitzenbauer, and N. Parotsidis. Faster algorithms for computing maximal 2-connected subgraphs in sparse directed graphs. In *SODA '17: Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1900–1918, 2017.

[10] R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer-Verlag Berlin Heidelberg, 5th edition, 2016.

[11] Y. Dourisboure, F. Geraci, and M. Pellegrini. Extraction and classification of dense communities in the web. In *WWW '07: Proceedings of the 16th International Conference on World Wide Web*, pages 461–470, 2007.

[12] A. Epasto, S. Lattanzi, and M. Sozio. Efficient densest subgraph computation in evolving graphs. In *WWW '15: Proceedings of the 24th International Conference on World Wide Web*, pages 300–310, 2015.

[13] U. Feige, D. Peleg, and G. Kortsarz. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001.

[14] S. Forster, D. Nanongkai, L. Yang, T. Saranurak, and S. Yingchareonthawornchai. Computing and testing small connectivity in near-linear time and queries via fast local cut algorithms. In *SODA '20: Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2046–2065, 2020.

[15] A. Frank. On the edge-connectivity algorithm of nagamochi and ibaraki. *Laboratoire Artemis, IMAG, Universite J. Fourier*, 1994.

[16] H. N. Gabow. Using expander graphs to find vertex connectivity. *Journal of the ACM*, 53(5):800–844, 2006.

[17] E. Galimberti, F. Bonchi, and F. Gullo. Core decomposition and densest subgraph in multilayer networks. In *CIKM '17: Proceedings of the 26th ACM International Conference on Information and Knowledge Management*, pages 1807–1816, 2017.

[18] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., NY, 1979.

[19] D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs. In *VLDB '05: Proceedings of the 31st International Conference on Very Large Data Bases*, pages 721–732, 2005.

[20] A. Gionis and C. E. Tsourakakis. Dense subgraph discovery: KDD 2015 Tutorial. In *KDD '15: Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2313–2314, 2015.

[21] A. V. Goldberg. *Finding a maximum density subgraph*. University of California Berkeley, 1984.

[22] M. Henzinger, S. Krinninger, and V. Loitzenbauer. Finding 2-edge and 2-vertex strongly connected components in quadratic time. In *ICALP '15: Proceedings of the 42nd International Colloquium on Automata, Languages and Programming*, pages 713–724, 2015.

[23] M. Henzinger, S. Rao, and D. Wang. Local flow partitioning for faster edge connectivity. *SIAM Journal on Computing*, 49(1):1–36, 2020.

[24] M. R. Henzinger, S. Rao, and H. N. Gabow. Computing vertex connectivity: New bounds from old techniques. *Journal of Algorithms*, 34(2):222–250, 2000.

[25] J. E. Hopcroft and R. E. Tarjan. Dividing a graph into triconnected components. *SIAM Journal on Computing*, 2(3):135–158, 1973.

[26] S. Hu, X. Wu, and T.-H. H. Chan. Maintaining densest subsets efficiently in evolving hypergraphs. In *CIKM '17: Proceedings of the 26th ACM International Conference on Information and Knowledge Management*, pages 929–938, 2017.

[27] D. R. Karger. Minimum cuts in near-linear time. *Journal of the ACM*, 47(1):46–76, 2000.

[28] K.-I. Kawarabayashi and M. Thorup. Deterministic edge connectivity in near-linear time. *Journal of the ACM*, 66(1):Article No. 4, 2018.

[29] Y. Kawase, Y. Kuroki, and A. Miyauchi. Graph mining meets crowdsourcing: Extracting experts for answer aggregation. In *IJCAI '19: Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 1272–1279, 2019.

[30] Y. Kawase and A. Miyauchi. The densest subgraph problem with a convex/concave size function. *Algorithmica*, 80(12):3461–3480, 2018.

[31] S. Khuller and B. Saha. On finding dense subgraphs. In *ICALP '09: Proceedings of the 36th International Colloquium on Automata, Languages and Programming*, pages 597–608, 2009.

[32] J. Leskovec and A. Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, 2014.

[33] N. Linial, L. Lovász, and A. Wigderson. Rubber bands, convex embeddings and graph connectivity. *Combinatorica*, 8(1):91–102, 1988.

[34] W. Mader. Existenzn-fach zusammenhängender teilgraphen in graphen genügend großer kantendichte. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 37(1):86–97, 1972.

[35] W. Mader. Connectivity and edge-connectivity in finite graphs. In B. Bollobas, editor, *Surveys in Combinatorics (Proceedings of the Seventh British Combinatorial Conference)*, volume 38 of *London Mathematical Society Lecture Note Series*, pages 66–95, 1979.

[36] S. Makino. An algorithm for finding all the k-components of a digraph. *International journal of computer mathematics*, 24(3-4):213–221, 1988.

[37] D. W. Matula. Graph theoretic techniques for cluster analysis algorithms. In J. V. Ryzin, editor, *Classification and Clustering*, pages 95–129. Academic Press, 1977.

[38] D. W. Matula. k-blocks and ultrablocks in graphs. *Journal of Combinatorial Theory, Series B*, 24(1):1–13, 1978.

[39] M. Mitzenmacher, J. Pachocki, R. Peng, C. E. Tsourakakis, and S. C. Xu. Scalable large near-clique detection in large-scale networks via sampling. In *KDD '15: Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 815–824, 2015.

[40] A. Miyauchi, Y. Iwamasa, T. Fukunaga, and N. Kakimura. Threshold influence model for allocating advertising budgets. In *ICML '15: Proceedings of the 32nd International Conference on Machine Learning*, pages 1395–1404, 2015.

[41] A. Miyauchi and A. Takeda. Robust densest subgraph discovery. In *ICDM '18: Proceedings of the 18th IEEE International Conference on Data Mining*, pages 1188–1193, 2018.

[42] H. Nagamochi and T. Ibaraki. Computing edge-connectivity in multigraphs and capacitated graphs. *SIAM Journal on Discrete Mathematics*, 5(1):54–66, 1992.

[43] D. Nanongkai, T. Saranurak, and S. Yingchareonthawornchai. Breaking quadratic time for small vertex connectivity and an approximation scheme. In *STOC '19: Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 241–252, 2019.

[44] J. B. Orlin. Max flows in $O(nm)$ time, or better. In *STOC '13: Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 765–774, 2013.

[45] V. Spirin and L. A. Mirny. Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences of the United States of America*, 100(21):12123–12128, 2003.

[46] M. Stoer and F. Wagner. A simple min-cut algorithm. *Journal of the ACM*, 44(4):585–591, 1997.

[47] R. Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.

[48] C. E. Tsourakakis. The k-clique densest subgraph problem. In *WWW '15: Proceedings of the 24th International Conference on World Wide Web*, pages 1122–1132, 2015.

[49] C. E. Tsourakakis. Streaming graph partitioning in the planted partition model. In *COSN '15: Proceedings of the 2015 ACM Conference on Online Social Networks*, pages 27–35, 2015.

[50] C. E. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarli. Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees. In *KDD '13: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 104–112, 2013.

[51] C. E. Tsourakakis, T. Chen, N. Kakimura, and J. Pachocki. Novel dense subgraph discovery primitives: Risk aversion and exclusion queries. In *ECML-PKDD '19: Proceedings of the 2019 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2019. No page numbers.

[52] Y. Wu, X. Zhu, L. Li, W. Fan, R. Jin, and X. Zhang. Mining dual networks: Models, algorithms, and applications. *ACM Transactions on Knowledge Discovery from Data*, 10(4):40:1–40:37, 2016.

[53] Z. Zou. Polynomial-time algorithm for finding densest subgraphs in uncertain graphs. In *MLG '13: Proceedings of the 11th Workshop on Mining and Learning with Graphs*, 2013. No page numbers.