Computing the k Densest Subgraphs of a Graph

Riccardo Dondi
Università degli Studi di Bergamo, Bergamo, Italy
riccardo.dondi@unibg.it

Danny Hermelin
Ben-Gurion University of the Negev, Be'er Sheva, Israel
hermelin@bgu.ac.il

Abstract

Computing cohesive subgraphs is a central problem in graph theory. While many formulations of cohesive subgraphs lead to NP-hard problems, finding a densest subgraph can be done in polynomial-time. As such, the densest subgraph model has emerged as the most popular notion of cohesiveness. Recently, the data mining community has started looking into the problem of computing k densest subgraphs in a given graph, rather than one, with various restrictions on the possible overlap between the subgraphs. However, there seems to be very little known on this important and natural generalization from a theoretical perspective.

In this paper we hope to remedy this situation by analyzing three natural variants of the k densest subgraphs problem. Each variant differs depending on the amount of overlap that is allowed between the subgraphs. In one extreme, when no overlap is allowed, we prove that the problem is NP-hard for $k \geq 3$, but polynomial-time solvable for $k \leq 2$. On the other extreme, when overlap is allowed without any restrictions and the solution subgraphs only have to be distinct, we show that the problem is fixed-parameter tractable with respect to k, and admits a PTAS for constant k. Finally, when a limited of overlap is allowed between the subgraphs, we prove that the problem is NP-hard for $k = 2$.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis; Mathematics of computing → Graph theory; Networks → Network algorithms

Keywords and phrases Algorithm Design, Network Mining and Analysis, Densest Subgraph, Algorithmic Aspects of Networks

Digital Object Identifier 10.4230/LIPIcs...

1 Introduction

Finding cohesive subgraphs is a central problem in the analysis of social networks [18], graph-mining [23, 24], group dynamics research [8], computational biology [10], and many other areas. The most basic and natural attempt at modeling cohesiveness is via the notion of cliques; however, this notion is too strict and rigid for most applications, and is also known to be computationally hard [15, 27].

While there are several alternative definitions for cohesiveness [17], a notion that has emerged as arguably the most popular is the *densest subgraph* model [1, 5, 11, 20, 23, 24, 26]. Here, the *density* of a graph is simply the edge-to-vertex ratio in the graph, and the densest subgraph is the (induced) subgraph that maximizes this ratio. As opposed to the maximum clique, finding a densest subgraph in a graph is polynomial-time solvable [13, 22]. This fact, along with the naturality of the concept, has lead the notion of density to nowadays be considered at the core of large scale data mining [4].

Recent contributions have shifted the interest from computing a single cohesive subgraph to computing a set of such subgraphs [5, 11, 20, 25], as this is naturally more desirable in most applications. The proposed approaches may allow (but not force) the subgraph to have a limited overlap, as many real-world cohesive groups may share some elements (for example hubs may belong to more than one community [19, 11]). The way the overlap is restricted varies among the different approaches. For instance, in [5], the notion of overlap is restricted via a constraint on the pairwise Jaccard coefficient of the subgraphs of the solution, while in [11] the total overlap is factored into the objective function.

1.1 Three natural variants

Despite the fact that the problem of computing the k densest subgraphs is quite natural, there seems to be very little known on this problem from a theoretical perspective. In this paper we aim to remedy this situation by considering three natural variants of the problem. The variants depend on how much overlap is allowed between the subgraphs. The objective function is the same across all three problems - the total sum of densities of the solution subgraphs.

In the most lenient variant, the only requirement is that the subgraphs are distinct (*i.e.* have different vertex sets). Thus, in this problem, a solution subgraph may be a subgraph, a supergraph, or have almost the exact same vertex set as another solution subgraph.

▷ Problem 1. k-Densest Subgraphs
Input: A graph G.
Output: k pairwise distinct subgraphs $G_1, \ldots, G_k$ of G.

On the other extreme no overlap is allowed whatsoever, and the requirement is that the vertex sets of the solution subgraphs be completely disjoint.

▷ Problem 2. k-Disjoint Densest Subgraphs
Input: A graph G.
Output: k pairwise disjoint subgraphs $G_1, \ldots, G_k$ of G.

Finally, between the two above problems lies the case where some overlap is allowed. Here we allow each solution subgraph to have a certain constant fraction of its vertices shared by other solution subgraphs.

▷ Problem 3. k-Overlapping Densest Subgraphs
Input: A graph G, and a constant $\alpha \in [0, 1]$.
Output: k pairwise distinct subgraphs $G_1, \ldots, G_k$ of G, $G_i = (V_i, E_i)$, such that for every $i \in \{1, \ldots, k\}$: G_i overlaps with any $G_j \neq G_i$ in at most $\alpha \cdot |V_i|$ vertices.

1.2 Our results

We analyze the complexity of each of the three problems discussed above. For the k-Disjoint Densest Subgraphs problem, where the solution subgraphs are required to be completely disjoint, we obtain the following interesting dichotomy:

▶ **Theorem 1.** k-Disjoint Densest Subgraphs *can be solved in $O(mn^3 \lg n)$ time for $k \leq 2$, and is NP-hard for $k \geq 3$.*

We find the case of $k = 2$ rather surprising, as it is quite rare these days to find a new natural graph problem that is polynomial-time solvable. The algorithm is an adaptation of the algorithm used for the case of $k = 1$ [13, 22], but with an addition of new ideas that allow the extension to follow through. The hardness result for the case of $k \geq 3$ is via a reduction from the 3-Clique Partition problem (see Section 3.2 for a formal definition).

Our second result concerns the k-Densest Subgraphs problem. The problem is known to be solvable in $O(n^k)$ time [7], which is quite slow even for moderate and small values of k. Here we show that the problem admits an efficient PTAS (or EPTAS). More precisely, we prove that:

▶ **Theorem 2.** *For any fixed k and ε, there is an algorithm that computes in $O(mn \lg n)$ time a $(1 - \frac{1}{\varepsilon})$-approximate solution for* k-Densest Subgraphs.

Moreover, we show that the problem is in fact fixed-parameter tractable when parameterized by the number of subgraphs k. That is, we show that the problem can be solved exactly by an algorithm of running in time $f(k) \cdot n^{O(1)}$, significantly improving the previously known $O(n^k)$ time algorithm [7].

▶ **Theorem 3.** k-Densest Subgraphs *can be solved in $O(2^k kmn^3 \lg n)$ time.*

Finally, in the last part of the paper we show that the k-Overlapping Densest Subgraphs problem is NP-hard already for $k = 2$, in contrast to k-Disjoint Densest Subgraphs. This hardness result is obtained via a reduction from the Minimum Bisection problem (see Section 6 for a formal definition).

▶ **Theorem 4.** 2-Overlapping Densest Subgraphs *is NP-hard.*

Some of the proofs are omitted due to page limit.

1.3 Related work

The Densest Subgraph problem, the problem of computing a densest subgraph in a given graph, is the special case of each of the three problems considered above when $k = 1$. This problem has been extensively studied in the literature, and we outline here only the main results: The problem is known to be polynomial-time solvable [13, 22], and it can be approximated within a factor of $\frac{1}{2}$ in linear time [3, 6]. Generalization of the problem to weighted graphs [13], as well as directed graphs [16], also turn out to be polynomial-time solvable. However, the Densest Subgraph problem becomes NP-hard when constraints on the number of vertices in the output graph are added [1, 2, 9, 14, 16].

2 Preliminaries

All graphs considered in this paper are simple, undirected, and without self-loops. Throughout the paper we let $G = (V, E)$ denote an input graph, and we let $n = |V|$ and $m = |E|$. For a vertex $v \in V$, we let $deg(v)$ denote the *degree* of v in G, *i.e.* $deg(v) = |\{u \in V : \{u, v\} \in E\}|$. The density of G is defined by $density(G) = m/n$, and in general, the density of a graph is the ratio between the number of edges and the number of vertices in the graph.

Given a subset of vertices $V_1 \subseteq V$, we denote by $G[V_1]$ the *subgraph* of G induced by V_1; formally, $G[V_1] = (V_1, E_1)$ where $E_1 = \{\{u, v\} \in E : u, v \in V_1\}$. Thus, a subgraph of G is determined completely by its subset of vertices. If $G[V_1]$ and $G[V_2]$ are both subgraphs of G, then we say that these subgraphs are *distinct* whenever $V_1 \neq V_2$. If $V_1 \cap V_2 = \emptyset$ then the two subgraphs are *disjoint*, and if $V_1 \subset V_2$, then $G[V_2]$ is a *proper supergraph* of $G[V_1]$.

2.1 Goldberg's algorithm

As mentioned above, the Densest Subgraph problem can be solved in polynomial-time [13, 22]. The main idea is to reduce the problem to a series of min-cut computations. Picard and Queyranne's algorithm [22] requires $O(n)$ such computations, where n is the number of vertices in the input graph, while Goldberg's algorithm [13] improves this to $O(\lg n)$. Thus, since a single min-cut computation can be done in $O(mn)$ time via Orlin's algorithm [21], Goldberg's algorithm has $O(mn \lg n)$ running time. Furthermore, Goldberg also showed that one can compute in the same running-time a densest subgraph in a *vertex-weighted* graph; here, the density of a vertex-weight graph H on n vertices of total weight w and m edges is given by $density(H) = \frac{m+w}{n}$.

3 k-Disjoint Densest Subgraphs

In the following section we consider the k-Disjoint Densest Subgraphs problem. Recall that we are given a graph G on n vertices and m edges, and our goal is to compute k disjoint subgraphs of G with maximum total densities. We provide a proof for Theorem 1, split into two parts. In Section 3.1 we show that the 2-Disjoint Densest Subgraphs problem is solvable in $O(mn \lg n)$ time, while in Section 3.2 we prove that the problem is NP-hard for $k \geq 3$.

3.1 Tractability of 2-Disjoint Densest Subgraphs

Our algorithm for the 2-Disjoint Densest Subgraphs problem is an adaptation of Goldberg's algorithm for the Densest Subgraph problem discussed in Section 2.1 above. As such, it relies on reducing the problem to several minimum cut computations.

Let $G = (V, E)$ be the input graph of 2-Disjoint Densest Subgraphs with $|V| = n$ and $|E| = m$. Also, let $u \neq z \in V$ be two distinct vertices of G, and let d be some fixed density value. Below we present an algorithm which determines in $O(mn)$ time whether G has two disjoint subgraphs G_1 and G_2 with $density(G_1)+density(G_2) > d$, such that u belongs to G_1 and z belongs to G_2. By performing binary search on all possible values of d, and by iterating on all possible $u \neq z \in V$, this algorithm can be used to find two densest disjoint subgraphs in G, yielding a total run-time of $O(mn^3 \lg n)$ as stated in Theorem 1. Furthermore, using Goldberg's algorithm in a preprocessing step, we can assume that a densest subgraph in G has density at most d, and so our algorithm will only search for two non-empty graphs G_1 and G_2.

Our algorithm constructs from G and the two vertices u and z an edge-weighted directed graph $H_{u,z}$, with a source vertex s and a target vertex t. The minimal s,t-cut in $H_{u,z}$ will determine whether there exist two disjoint non-empty subgraphs in G, each including u and z, with total densities greater than d. The $O(mn)$ run-time of our algorithm will then follow from the fact that this construction can be carried out in linear time, and from the fact that a minimum cut in a graph can be computed in $O(mn)$ time [21].

3.1.1 The construction:

The directed graph $H_{u,z}$ will consist of a source vertex s, a target vertex t, and two disjoint copies v_a and v_b for each vertex v of G. We write $A = \{v_a : v \in V\}$ and $B = \{v_b : v \in V\}$. The arc set of H is defined as follows:

- An arc (s, v_a) with weight $w(s, v_a) = m$ for each $v_a \in A$, where $v \neq u$, and an arc (s, v_b) with weight $w(s, v_b) = m + d - deg(v)$ for each $v_b \in B$, where $v \neq u$.
- An arc (s, u_a) with weight $w(s, u_a) = \infty$ for $u_a \in A$ and an arc (s, u_b) with weight $w(s, v_b) = \infty$, for $u_b \in B$ (u_a, u_b are associated with vertex u).
- An arc (v_a, t) with weight $w(v_a, t) = m + d - deg(v)$ for each $v_a \in A$ where $v \neq z$, and an arc (v_b, t) with weight $w(v_b, t) = m$ for each $v_b \in B$ where $v \neq z$.
- An arc (z_a, t) with weight $w(z_a, t) = \infty$, and an arc (z_b, t) with weight $w(z_b, t) = \infty$ (z_a, z_b are associated with vertex z).
- Four arcs (u_a, v_a), (v_a, u_a), (u_b, v_b), (v_b, u_b), each of weight 1, for each edge $\{u, v\}$ of G.
- An arc (v_a, v_b) with weight ∞ for each vertex v of G.

Given two disjoint subsets of vertices X and Y in $H_{u,z}$, we use $w(X, Y)$ to denote the total weight of arcs outgoing from vertices in X to vertices in Y.

The main idea behind our construction is as follows: Consider an s,t-cut (S, T) in $H_{u,z}$. The cut (S, T) naturally partitions each of the two copies of V in $H_{u,z}$ into two (not necessarily non-empty) parts. Let $A_S = S \cap A$ and $A_T = T \cap A$ denote the two parts of A, and let $B_S = S \cap B$ and $B_T = T \cap B$ denote the two parts of B. Our goal is to have the two solution subgraphs of G encoded by A_S and B_T. Thus, from now on, for a given s,t-cut (S, T) of $H_{u,z}$, we let G_1 and G_2 respectively denote the subgraphs of G induced by $V_1 = \{v \in V : v_a \in A_S\}$ and $V_2 = \{v \in V : v_b \in B_T\}$. For $i \in \{1, 2\}$, we use d_i and n_i to respectively denote the density and number of vertices in G_i.

3.1.2 Analysis:

Consider an s,t-cut (S, T) in $H_{u,z}$, and let A_S, A_T, B_S, and B_T be defined as above. Then $S = \{s\} \cup A_S \cup B_S$ and $T = \{t\} \cup A_T \cup B_T$. As there is no arc between s and t, we can write the total weight $w(S, T)$ of (S, T) as

$$w(S,T) = w(\{s\} \cup A_S, \{t\} \cup A_T) + w(\{s\} \cup B_S, \{t\} \cup B_T) + w(A_S, B_T). \tag{1}$$

Below we calculate each of these three terms separately in the three different lemmas. Lemma 9 will then combine these three lemmas to provide us with the connection between minimum s,t-cuts in $H_{u,z}$ and the maximum total densities of two disjoint subgraphs in G.

▶ **Lemma 5.** *A cut (S, T) of $H_{u,z}$ is such that $w(S, T) < \infty$ iff $u_a, u_b \in S$ and $z_a, z_b \in T$.*

▶ **Lemma 6.** $w(\{s\} \cup A_S, \{t\} \cup A_T) = mn + 2n_1(d/2 - d_1)$.

Proof. The arcs between $\{s\} \cup A_S$ and $\{t\} \cup A_T$ can be partitioned into three sets: The arcs outgoing from s to A_T, the arcs from A_S to t, and the arcs outgoing from A_S to A_T. Now, if $A_S = \emptyset$, then $n_1 = |A_S| = 0$ and $|A_T| = n$, and we have $w(\{s\} \cup A_S, \{t\} \cup A_T) = w(\{s\}, A_T) = m|A_T| = mn$, and so the lemma holds.

Assume therefore that $A_S \neq \emptyset$. Accounting for all three sets of arcs discussed above we have

$$\begin{aligned} w(\{s\} \cup A_S, \{t\} \cup A_T) &= w(\{s\}, A_T) + w(A_S, \{t\}) + w(A_S, A_T) \\ &= m|A_T| + (m+d)|A_S| - \sum_{v_a \in A_S} deg(v) + |E(A_S, A_T)| \\ &= m|A| + d|A_S| - \sum_{v_a \in A_S} deg(v) + |E(A_S, A_T)|. \end{aligned}$$

Note that $\sum_{v_a \in A_S} deg(v) - |E(A_S, A_T)|$ is precisely twice the number of edges in G_1, the subgraph of G induced by $V_1 = \{v \in V : v_a \in A_S\}$. As A_S is non-empty, we can factor out the term $2|A_S| = 2n_1$, to obtain

$$m|A| + d|A_S| - \sum_{v_a \in A_S} deg(v) + |E(A_S, A_T)| = mn + 2n_1(d/2 - d_1),$$

and the lemma follows. ◀

Similarly to the previous lemma, we prove the following results.

▶ **Lemma 7.** $w(\{s\} \cup B_S, \{t\} \cup B_T) = mn + 2n_2(d/2 - d_2)$.

▶ **Lemma 8.** *$w(A_S, B_T) = 0$ iff $V_1 \cap V_2 = \emptyset$, and otherwise $w(A_S, B_T) = \infty$.*

3.1.3 Summary:

Our algorithm for 2-Disjoint Densest Subgraphs can now be described as follows: On given input (G, d), the algorithm first constructs the directed graph $H_{u,z}$, and then it computes the weight of a minimum cut in G. It determines that G has two disjoint subgraphs with total densities greater than d, one containing vertex u and one containing vertex z, only if $H_{u,z}$ has an s,t-cut with total weight less than $2mn$. Given u and z, the running-time of this algorithm is $O(mn)$ using the recent max-flow/min-cut algorithm of Orlin [21], accounting also for the $O(m+n)$ time required for constructing $H_{u,z}$. The overall running-time of the algorithnm is therefore $O(mn^3)$, by iterating for each pair of vertices $u, z \in V$. The correctness of the algorithm follows from the lemma below.

▶ **Lemma 9.** *$w(S,T) < 2mn$ iff $V_1, V_2 \neq \emptyset$, $V_1 \cap V_2 = \emptyset$, and $d_1 + d_2 > d$.*

Proof. First, according to Lemma 5, $w(S,T) < \infty$ iff $V_1, V_2 \neq \emptyset$. Moreover, according to Lemma 8, $w(S,T) < \infty$ iff $V_1 \cap V_2 = \emptyset$. In this case, by Lemma 6 and Lemma 7, we have

$$\begin{aligned} w(S,T) &= w(\{s\} \cup A_S, \{t\} \cup A_T) + w(\{s\} \cup B_S, \{t\} \cup B_T) + w(A_S, B_T) \\ &= 2mn + 2n_1(d/2 - d_1) + 2n_2(d/2 - d_2). \end{aligned}$$

Assume w.l.o.g. that $n_2 \geq n_1$. Then

$$\begin{aligned} w(S,T) &= 2mn + 2n_1(d/2 - d_1) + 2n_2(d/2 - d_2) \\ &\geq 2mn + 2n_1(d/2 - d_1 + d/2 - d_2) = 2mn + 2n_1(d - (d_1 + d_2)). \end{aligned}$$

The lemma then follows as $2n_1(d - (d_1 + d_2)) < 0$ iff $d_1 + d_2 > d$. ◀

3.2 k-Disjoint Densest Subgraphs for $k \geq 3$

We next consider the k-Disjoint Densest Subgraphs problem for $k \geq 3$. We first show that 3-Disjoint Densest Subgraphs is NP-hard, and then we extend this result to larger values of k. Our hardness result for $k = 3$ relies on a reduction from the 3-Clique Partition partition problem, which is well known to be NP-hard [15].

▷ Problem 4. 3-Clique Partition
Input: A graph $G = (V, E)$.
Output: a partition of V into three sets V_1, V_2, and V_3 such that $G[V_i]$, $1 \leq i \leq 3$, is a clique.

The connection between 3-Clique Partition and 3-Disjoint Densest Subgraphs is given in the following lemma.

▶ **Lemma 10.** *Let $G = (V, E)$ be a graph. Then the vertex set of G can be partitioned into three cliques iff G has three disjoint subgraphs with total densities at least $(|V| - 3)/2$.*

Lemma 10 proves that 3-Disjoint Densest Subgraphs is NP-hard, as it shows that 3-Clique Partition reduces to the problem. To extend this reduction to larger values of k, one simply needs to add $k-3$ cliques of size $|V|$, and require the total densities of the solution subgraphs to be at least $(k-3)(|V|-1)/2 + (|V|-3)/2$. We omit details for the sake of brevity.

4 An EPTAS for k-Densest Subgraphs

In the following section we describe our EPTAS for k-Densest Subgraphs. Let (G, k) denote a given instance of k-Densest Subgraphs, and let $\varepsilon > 0$ be a given constant. Our goal is to compute in $O(mn \lg n)$ time k distinct subgraphs $G_1, \ldots, G_k$ of G with densities $d_1, \ldots, d_k$ such that $\sum_i d_i \geq (1 - \frac{1}{\varepsilon}) \cdot OPT$, where OPT is the total sum of densities of the k densest subgraphs in G. Recall that $k = O(1)$.

Below we first provide a description of our algorithm, followed by an analysis of its running time, and an analysis of it's approximation ratio guarantee. Since the function $(\frac{n-2k}{n})^k$ tends to 1 as n grows to infinity, we will henceforth assume that n is sufficiently large so that the following inequality holds:

$$\left(\frac{n-2k}{n}\right)^k \geq \left(1 - \frac{1}{\varepsilon}\right). \tag{2}$$

If n does not satisfy this inequality, we can solve the problem optimally via brute force in $O(1)$ time.

4.1 The algorithm

We say that a subgraph $G_i = (V_i, E_i)$ of G is *small* if $|V_i| \leq \varepsilon - 1$. Our algorithm proceeds in a certain way so long that all subgraphs computed so far are small; once a subgraph which is not small is computed, the algorithm proceeds in a different manner. The first subgraph $G_1 = (V_1, E_1)$ is computed using Goldberg's algorithm, so G_1 is a densest subgraph in G.

Suppose that we have computed subgraphs $G_1, \ldots, G_i$ for some $1 \leq i \leq k-1$, and all these subgraphs are small. The subgraph G_{i+1} is taken to be a densest graph out of all of the following possible candidates:

- A densest subgraph in $G - \{v_1, \ldots, v_i\}$ for some $v_1 \in V_1, \ldots, v_i \in V_i$.
- A densest strict supergraph of G_j in G for some $j \in \{1, \ldots, i\}$.

Note that some of the candidates of the second type above can be graphs in $\{G_1, \ldots, G_i\}$; such graphs are naturally excluded from being candidates for the subgraph G_{i+1}.

Suppose that we have computed subgraphs $G_1, \ldots, G_i$ for some $1 \leq i \leq k-1$, and $G_i = (V_i, E_i)$ is not small. Then in this case G_i can either be big or huge. We say that G_i is *big* if $|V_i| \leq n - k - i$, and otherwise it is *huge*. If G_i is big, we choose arbitrary distinct vertices $v_{i+1}, \ldots, v_k \in V \setminus V_i$ and set G_j to be the graph induced by $V_i \cup \{v_j\}$ for $j \in \{i+1, \ldots, k\}$. Note that since V_i is not huge, there are enough distinct vertices in $V \setminus V_i$. Also note that as G_i is the only big subgraph in $G_1, \ldots, G_i$, it is not a subgraph of any of these graphs and so all subgraphs G_j are distinct from all subgraphs computed so far.

If G_i is huge, then the graphs $G_{i+1}, \ldots, G_k$ are computed by iteratively removing minimal degree vertices in G_i. Note that as G_i is huge and all graphs $G_1, \ldots, G_{i-1}$ are small, we are guaranteed that subgraphs computed in this way are distinct from those we have computed so far.

4.2 Run-time analysis

Before analyzing the run-time of our algorithm, we begin with the following lemma:

▶ **Lemma 11.** *Let H_0 be a strict subgraph of G, and let H be a densest strict supergraph of H_0 in G. If $density(H) \leq density(H_0)$, then there is an algorithm that computes H in $O(mn)$ time given H_0 as input.*

Proof. Given $H_0 = (V_0, E_0)$ as input, the algorithm uses Goldberg's algorithm to compute a densest subgraph $H_1 = (V_1, E_1)$ in the vertex-weighted graph $G^* = G - V_0$, with vertex weights defined by $w(v) = |N_G(v) \cap V_0|$ for each vertex v of G^*. It then returns the graph

$H = H_0 \cup H_1$ as a solution. Clearly, this can be done in $O(mn)$ time, and H is a strict supergraph of H_0 in G. We claim that H_1 is indeed a densest among all supergraphs of H_0.

Let $H' = (V', E')$ be any strict supergraph of H_0 ($V_0 \subset V'$), and let $H_2 = (V_2, E_2)$ be the subgraph of G induced by $V_2 = V' \setminus V_0$. Our goal is to show that H is at least as dense as H' in G. Let $n_i = |V_i|$ and $m_i = |E_i| + \sum_{v \in H_i} w(v)$ for $i \in \{1, 2\}$. Then the density of H_1 and H_2 in the vertex weighted graph G^* is $d_1 = m_1/n_1$ and $d_2 = m_2/n_2$ respectively. Also, by letting $n_0 = |V_0|$ and $m_0 = |E_0|$, the density of H_0 in G is given by $density(H_0) = m_0/n_0$. Furthermore, observe that by the definition of the vertex weight function in G^*, we have

$$density(H) = \frac{|E_0| + |E_1| + |E(V_0, V_1)|}{|V_0| + |V_1|} = \frac{|E_0| + |E_1| + \sum_{v \in V_1} |N(v) \cap V_0|}{|V_0| + |V_1|} = \\ = \frac{|E_0| + |E_1| + \sum_{v \in V_1} w(v)}{|V_0| + |V_1|} = \frac{m_0 + m_1}{n_0 + n_1},$$

and similarly, $density(H') = (m_0 + m_2)/(n_0 + n_2)$. Below we argue that $density(H)$ is at least as large as $density(H')$.

By standard algebra, we have

$$\begin{aligned} density(H) \geq density(H') & \quad\Longleftrightarrow \\ \frac{m_0 + m_1}{n_0 + n_1} \geq \frac{m_0 + m_2}{n_0 + n_2} & \quad\Longleftrightarrow \\ m_0 n_2 + m_1(n_0 + n_2) \geq m_0 n_1 + m_2(n_0 + n_1) & \quad\Longleftrightarrow \\ m_1 n_2 + m_0(n_2 - n_1) \geq m_2 n_1 + n_0(m_2 - m_1) & \end{aligned}$$

The last inequality is implied by the following inequalities: $m_1 n_2 \geq m_2 n_1$ and $\frac{m_0}{n_0} \geq \frac{m_2 - m_1}{n_2 - n_1}$.

For the first inequality, observe that $d_1 = m_1/n_1 \geq d_2 = m_2/n_2$ as H_1 is a densest subgraph in G^*; this directly implies $m_1 n_2 \geq m_2 n_1$.

For second inequality, by the assumption in the lemma we can prove the following claim:

▷ Claim 12. $density(H_0) \geq d_1$.

Thus, we have

$$\frac{m_0}{n_0} = density(H_0) \geq d_1 = \frac{d_1(n_2 - n_1)}{n_2 - n_1} \geq \frac{d_2 n_2 - d_1 n_1}{n_2 - n_1} = \frac{m_2 - m_1}{n_2 - n_1},$$

and so also the second inequality holds, thus concluding the proof. ◀

Now, first observe that G_1 is computed in $O(mn \lg n)$ time, which is the running time of Goldberg's algorithm. Next, note that if some subgraph G_i is big or huge, then the remaining graphs $G_{i+1}, \ldots, G_k$ can easily be computed in $O(m + n)$ time. Consider then a small subgraph G_i for some $i \leq k - 1$. Then, by construction, all subgraphs $G_1, \ldots, G_i$ are small, and so we have $|V_1| \cdots |V_i| = O(1)$. The subgraph G_{i+1} is computed by first computing candidates of two different types. For the first type we need to invoke Goldberg's algorithm $|V_1| \cdots |V_i| = O(1)$ times, so this requires $O(mn \lg n)$ time. For the second type, we need to invoke the algorithm in Lemma 11 above $i = O(1)$ times, and so this also requires $O(mn \lg n)$ time. In total, we compute each subgraph G_i in $O(mn \lg n)$ time, which gives a similar run-time for the entire algorithm since $k = O(1)$.

4.3 Approximation-ratio analysis

Let $G_1^*, \ldots, G_k^*$ be top k densest subgraphs in G, with densities $d_1^* \geq d_2^* \geq \cdots \geq d_k^*$. We analyze the approximation ratio guaranteed by our algorithm by comparing the density of each subgraph $G_i = (V_i, E_i)$ computed by the algorithm with d_i^*. For G_1 this is easy. Since G_1^* is a densest subgraph in G, and G_1 is the graph computed by Goldberg's algorithm, we have:

▶ **Lemma 13.** *$density(G_1) = d_1^*$.*

For the remaining graphs, our analysis splits into three cases depending on the type of graph previously computed by the algorithm.

▶ **Lemma 14.** *If G_i is small, for $i < k$, then $density(G_{i+1}) = d_{i+1}^*$.*

Proof. The optimal subgraph $G_{i+1}^* = (V_{i+1}^*, E_{i+1}^*)$ is either a supergraph of some graph in $G_1, \ldots, G_i$, or $V_j \setminus V_{i+1}^* \neq \emptyset$ for each $j \in \{1, \ldots, i\}$. Since the candidates for G_{i+1} considered by our algorithm in case G_i is small cover both these cases, the lemma follows. ◀

Note that Lemma 13 and Lemma 14 together imply that if all subgraphs computed by the algorithm are small, then $density(G_i) = d_i^*$ for each $i \in \{1, \ldots, k\}$, and our algorithm computes an optimal solution. Furthermore, the first big or huge subgraph it computes also has optimal densities. The next two lemmas deal with the remaining subgraphs that are computed after computing a big or huge subgraph.

▶ **Lemma 15.** *Suppose G_i, for $i < k$, is the first big subgraph computed by the algorithm. Then $density(G_j) \geq (1 - \frac{1}{\varepsilon}) \cdot d_j^*$ for each $j \in \{i+1, \ldots, k\}$.*

Proof. Let $n_i = |V_i|$ and $m_i = |E_i|$. By Lemma 13 and Lemma 14 we know that $m_i/n_i = d_i^*$. Furthermore, as G_i is big, we have $n_i > \varepsilon - 1$, or written differently $n_i/(\varepsilon - 1) > 1$. Now, as each G_j has $n_i + 1$ vertices and at least m_i edges, we have

$$density(G_j) \geq \frac{m_i}{n_i + 1} > \frac{m_i}{n_i + n_i/(\varepsilon - 1)} = \frac{\varepsilon - 1}{\varepsilon} \cdot \frac{m_i}{n_i} = (1 - \frac{1}{\varepsilon}) \cdot d_i^* \geq (1 - \frac{1}{\varepsilon}) \cdot d_j^*$$

◀

▶ **Lemma 16.** *Suppose G_i, for $i < k$, is the first huge subgraph computed by the algorithm. Then $density(G_j) \geq (1 - \frac{1}{\varepsilon}) \cdot d_j^*$ for each $j \in \{i+1, \ldots, k\}$.*

Proof. Let $n_i = |V_i|$ and $m_i = |E_i|$. Since G_i is huge we know that $n_i > n - k$, and again by Lemmas 13 and 14 we know that $m_i/n_i = d_i^*$. Let $v \in V_i$ be a vertex of minimum degree in G_i. Consider the subgraph G_{i+1}, constructed from G_i by removing the vertex $v \in V_i$ with minimum degree. Then the degree of v cannot exceed the average degree in G_i, and so $deg(v) \leq 2m_i/n_i$. Thus, the density of G_i can be bounded by:

$$density(G_{i+1}) = \frac{m_i - deg(v)}{n_i - 1} \geq \frac{m_i - 2m_i/n_i}{n_i - 1} = \frac{n_i - 2}{n_i - 1} \cdot d_i^* > \frac{n - k - 2}{n} \cdot d_i^*.$$

Extending this argument, it can be seen that the density of G_{i+j}, for any $j \in \{1, \ldots, k - i\}$, is bounded from below by $\left(\frac{n-k-j-1}{n}\right)^j \cdot d_i^*$. The lemma then directly follows from Equation 2. ◀

Summarizing, due to Lemmas 13, 14, 15, and 16, we know that $density(G_i) \geq (1 - \frac{1}{\varepsilon}) \cdot d_i^*$ for all $i \in \{1, \ldots, k\}$, and so in total we have:

$$\sum_{i=1}^{k} density(G_i) \geq \sum_{i=1}^{k} (1 - \frac{1}{\varepsilon}) \cdot d_i^* = (1 - \frac{1}{\varepsilon}) \cdot OPT.$$

This completes the proof of Theorem 2.

5 k-Densest Subgraphs **in FPT Time**

We next show that k-Densest Subgraphs is solvable in $O(2^k kmn^3 \lg n)$ time, *i.e.* that it is fixed-parameter tractable in k. Recall that our goal is to compute k subgraphs $G_1, \ldots, G_k$ of $G = (V, E)$ whose total density is maximal, and our only constraint is that these subgraphs need to be distinct.

Similarly to Section 4, our approach here is to iteratively compute G_1, then G_2, and so forth, where we start from a densest subgraph G_1 of G. In what follows, we assume we have already computed the subgraphs $G_1 = (V_1, E_1), \ldots, G_\ell = (V_\ell, E_\ell)$, for $\ell \in \{1, \ldots, k-1\}$, and our goal is to compute a densest subgraph $G_{\ell+1} = (V_{\ell+1}, E_{\ell+1})$ among all subgraphs in G distinct from $G_1, \ldots, G_\ell$. Let $V^* = \bigcup_{i=1}^{\ell} V_i$. We consider the following two cases:

1. There is some vertex $v \in V_{\ell+1}$ that is not in V^*, *i.e.* $V_{\ell+1} \nsubseteq V^*$.
2. $V_{\ell+1}$ is contained completely in V^*, *i.e.* $V_{\ell+1} \subseteq V^*$.

We compute a densest subgraph in each one of these cases, and then take the densest of the two. The first case where $V_{\ell+1} \nsubseteq V^*$ is easy: we iterate through all vertices $v \in V \setminus V^*$ and compute a densest subgraph of G that includes v, and then take the densest of all these subgraphs (each of them being distinct from $G_1, \ldots, G_\ell$).

▶ **Lemma 17.** *Let $v \in V$. A densest subgraph of G that includes v can be computed in $O(mn \lg n)$ time.*

▶ **Lemma 18.** *If $V_{\ell+1} \nsubseteq V^*$ then $G_{\ell+1}$ can be computed in $O(mn^2 \lg n)$ time.*

The second case where $V_{\ell+1} \subseteq V^*$ requires more details. We say that a non-empty subset $\mathcal{C} \subseteq \{V_1, \ldots, V_\ell\}$ *covers* $V_{\ell+1}$ if $V_{\ell+1} \subseteq V_{\mathcal{C}} = \bigcup_{V_i \in \mathcal{C}} V_i$, and it is a *minimal cover* if $V_{\ell+1} \nsubseteq V_{\mathcal{C}'}$ for any proper subset $\mathcal{C}' \subset \mathcal{C}$. Our approach is to compute for each non-empty subset $\mathcal{C} \subseteq \{V_1, \ldots, V_\ell\}$, a densest subgraph of G for which $\mathcal{C}$ is a minimal cover.

▶ **Lemma 19.** *Let $\mathcal{C} \subseteq \{V_1, \ldots, V_\ell\}$, and suppose that $\mathcal{C}$ is a minimal cover of $V_{\ell+1}$. If $V_{\ell+1} \neq V_{\mathcal{C}}$, then there are two vertices $v_{in}, v_{out} \in V_{\mathcal{C}}$ such that $v_{in} \in V_{\ell+1}$ and $v_{out} \notin V_{\ell+1}$, and there is no subset $V_i \in \mathcal{C}$ with $v_{in} \in V_i$ and $v_{out} \notin V_i$.*

Proof. Suppose that $V_{\ell+1} \neq V_{\mathcal{C}}$, and so $V_{\ell+1} \subset V_{\mathcal{C}}$. It follows that there exists a vertex $v_{out} \in V_{\mathcal{C}} \setminus V_{\ell+1}$. Consider the subset $\mathcal{C}' \subset \mathcal{C}$ which includes all vertex subsets in $\mathcal{C}$ that include the vertex v_{out}, *i.e.* $\mathcal{C}' = \{V_i \in \mathcal{C} : v_{out} \in V_i\}$. Note that $\mathcal{C}'$ is indeed a proper subset of $\mathcal{C}$, as v_{out} belongs to some graph in $\mathcal{C}$. If $\mathcal{C}' = \emptyset$, then v_{out} belongs to every subset $V_i \in \mathcal{C}$, and the lemma holds. If $\mathcal{C}' \neq \emptyset$, there must be some vertex $v_{in} \in V_{\ell+1} \setminus V_{\mathcal{C}'}$ by the minimality of $\mathcal{C}$, since otherwise $\mathcal{C}'$ would cover $V_{\ell+1}$. ◀

▶ **Lemma 20.** *If $V_{\ell+1} \subseteq V^*$ then G_ℓ can be computed in $O(2^k mn^3 \lg n)$ time.*

Thus, taking the densest of the subgraph given by Lemma 18 and the subgraph given by Lemma 20 gives us a densest subgraph in G which is distinct from $\{G_1, \ldots, G_\ell\}$ in $O(2^k mn^3 \lg n)$ time. In this way, we can compute k densest distinct subgraphs of G in $O(2^k kmn^3 \lg n)$ time, and so Theorem 3 holds.

6 2-Overlapping Densest Subgraphs

In this section we prove that 2-Overlapping Densest Subgraphs is NP-hard. We prove this result by giving a reduction from Minimum Bisection (which is known to be NP-hard [12]), defined as follows.

▷ Problem 5. Minimum Bisection
Input: A graph $G_B = (V_B, E_B)$ with $|V_B|$ even.
Output: A partition of V_B into $V_{B,1}$ and $V_{B,2}$ such that $|V_{B,1}| = |V_{B,2}|$ and $|E(V_{B_1}, V_{B,2})| \leq h$.

Given $G_B = (V_B, E_B)$, with $|V_B| = n$, an input graph of Minimum Bisection, we construct in polynomial time an instance $G = (V, E)$ of 2-Overlapping Densest Subgraphs as follows. The graph G consists of a clique $G_c = (V_c, E_c)$ of size n and of a copy of G_B. Moreover, each vertex of G_B is connected to each vertex of G_c. Finally, we let $\alpha = \frac{2}{3}$. Thus if $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are two solution subgraphs of the instance, we require that $\frac{|V_1 \cap V_2|}{|V_1|} \leq \frac{2}{3}$ and $\frac{|V_1 \cap V_2|}{|V_2|} \leq \frac{2}{3}$.

We start by proving the following lemma that will be useful later on.

▶ **Lemma 21.** *Let $G[V']$ be a subgraph of G of density d. Consider a set of vertices X, disjoint from V' such that $d_x = \frac{|E(X) \cup E(X,V')|}{|X|}$. If $d_x \geq d$, then $G[V' \cup X]$ has density at least d.*

We next prove that there is an optimal solution where both subgraphs have all vertices of G_c included in them.

▶ **Lemma 22.** *Let $G_B = (V_B, E_B)$ be an input graph of* Minimum Bisection *and let G be the corresponding instance of* 2-Overlapping Densest Subgraphs*. Consider a solution $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ of* 2-Overlapping Densest Subgraphs *on instance G. Then, we can compute in polynomial time a solution $G[Y_1]$, $G[Y_2]$ of* 2-Overlapping Densest Subgraphs *such that $V_c \subseteq Y_1, Y_2$ and $density(G[Y_1]) + density(G[Y_2]) \geq density(G_1) + density(G_2)$.*

Proof. Starting from $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, we compute in polynomial time a solution $G[Y_1]$, $G[Y_2]$ of 2-Overlapping Densest Subgraphs as follows.

Preliminary Step.

Let $X = V_1 \cap V_2 \cap V_B$. While there exists $u \in X \neq \emptyset$ and V_c is not contained in V_i, with $1 \leq i \leq 2$, remove u from V_i and add a vertex of $V_c \setminus V_i$, to V_i.

After the preliminary step, either X is empty or both V_1 and V_2 contain V_c. Notice that the size of V_1 and V_2 is not changed and that $|V_1 \cap V_2|$ is not increased, hence $\frac{|V_1 \cap V_2|}{|V_1|} \leq \frac{2}{3}$ and $\frac{|V_1 \cap V_2|}{|V_2|} \leq \frac{2}{3}$. Moreover, the density of G_1 and G_2 is not decreased as each vertex $v \in V_c$ is connected to each vertex of $V \setminus \{v\}$. If $V_c \subseteq V_1, V_2$, by defining $Y_1 = V_1$ and $Y_2 = V_2$, the lemma holds. Consider the case that $V_c \not\subseteq V_i$, for some i with $1 \leq i \leq 2$, and notice that by construction of the preliminary step $V_1 \cap V_2 \cap V_B = \emptyset$. For each G_i, with $1 \leq i \leq 2$, if $V_c \not\subseteq V_i$ compute $G[Y_i]$ as follows. First, define $Y_i = V_i$, with $1 \leq i \leq 2$. Then apply the following two steps.

Step 1.

While there exists a vertex $u \in V_B$ not in $Y_1 \cup Y_2$, if $v \in V_c \setminus Y_i$, for some i with $1 \leq i \leq 2$, add u and v to Y_i. Notice that each iteration of Step 1 increases of 1 the overlapping between

Y_1 and Y_2, and of two the size of Y_i. Hence $\frac{|Y_1 \cap Y_2|}{|Y_1|} \leq \frac{2}{3}$ and $\frac{|Y_1 \cap Y_2|}{|Y_2|} \leq \frac{2}{3}$. Moreover, notice that after Step 1 each vertex of V_B belongs to exactly one of Y_1, Y_2. Indeed, if there exist a vertex $u \in V_B \setminus (Y_1 \cup Y_2)$, then $V_c \subseteq Y_1, Y_2$, hence $|Y_1| = |Y_2| = \frac{3}{2}n$ and by construction Y_1 and Y_2 must be disjoint.

Step 2.

Assume that $|Y_1 \cap V_B| < |Y_2 \cap V_B|$ and hence that $V_c \not\subseteq Y_1$. We assume that $V_c \subseteq Y_2$, since $|Y_2| \geq \frac{3}{2}n$. Then until $|Y_1 \cap V_B| = |Y_2 \cap V_B|$, a vertex $u \in Y_2 \cap V_B$ is removed from Y_2, u and a vertex $v \in V_c \setminus Y_1$ are added to Y_1. When $|Y_1 \cap V_B| = |Y_2 \cap V_B| = \frac{n}{2}$, since by construction $Y_1 \cap V_B$ and $Y_2 \cap V_B$ are disjoint, we add all the vertices of V_c to Y_i, and we obtain a solution with an overlapping of $\frac{2}{3}$ for both $G[Y_1]$ and $G[Y_2]$.

In the following claim, we show that the density of the computed solution is not decreased.

▷ Claim 23. $density(G[Y_i]) \geq density(G_i)$, for each i with $1 \leq i \leq 2$.

By Claim 23, since by construction $Y_1 \cap Y_2 = V_c$, we conclude that the proof holds. ◀

From Lemma 22 we can easily prove the following result.

▶ **Corollary 24.** *Consider an optimal solution $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ of* 2-Overlapping Densest Subgraphs *over instance G. Then $V_1 \cup V_2 = V$.*

Next we prove that a solution of 2-Overlapping Densest Subgraphs consists of two graphs whose overlap is exactly V_c.

▶ **Lemma 25.** *Let $G_B = (V_B, E_B)$ be an input graph of* Minimum Bisection *and let G be the corresponding instance of* 2-Overlapping Densest Subgraphs. *Consider a solution $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ of* 2-Overlapping Densest Subgraphs *on instance G that satisfies Lemma 22 and Corollary 24. Then $(V_1 \cap V_2) = V_c$ and $|V_1| = |V_2|$.*

We can conclude the reduction with the following proof.

▶ **Theorem 26.** *Let $G_B = (V_B, E_B)$ be an input graph of* Minimum Bisection *and let G be the corresponding instance of* 2-Overlapping Densest Subgraphs. *Then there exists a solution of* Minimum Bisection *on G_B that cuts h edges if and only if there exists a solution of* 2-Overlapping Densest Subgraphs *on instance G having density $\frac{2n^2-n+m-h}{\frac{3}{2}n}$.*

7 Conclusion

This paper introduces and studies three natural variants for computing the k densest subgraphs of a given graph, a central problem in graph data mining In the variant where no overlap is allowed, we prove that the problem is polynomial-time solvable for $k \leq 2$ and NP-hard for $k \geq 3$. For the variant when the graphs are required only to be distinct, we show that the problem is fixed-parameter tractable with respect to k, and admits a PTAS for $k = O(1)$. When a limited of overlap is allowed between the subgraphs, we prove that the the problem is NP-hard for $k = 2$. This later result can be extended to $k \geq 2$, but the details are somewhat technical, and are omitted to the full version of the paper.

From a theoretical perspective, the most interesting problem that is left open by our paper is whether the basic variant, k-Densest Subgraphs, is NP-hard for unbounded k. However, we feel that for most practical settings, the number k of solution subgraphs should be significantly smaller than the size n of the network. Thus, we feel that examining the three variants of k-Densest Subgraphs on specific social network models might be more interesting from a practical point of view.

References

1 Reid Andersen and Kumar Chellapilla. Finding dense subgraphs with size bounds. In Konstantin Avrachenkov, Debora Donato, and Nelly Litvak, editors, *Algorithms and Models for the Web-Graph, 6th International Workshop, WAW 2009, Barcelona, Spain, February 12-13, 2009. Proceedings*, volume 5427 of *Lecture Notes in Computer Science*, pages 25–37. Springer, 2009. doi:10.1007/978-3-540-95995-3_3.

2 Yuichi Asahiro, Refael Hassin, and Kazuo Iwama. Complexity of finding dense subgraphs. *Discrete Applied Mathematics*, 121(1-3):15–26, 2002. doi:10.1016/S0166-218X(01)00243-8.

3 Yuichi Asahiro, Kazuo Iwama, Hisao Tamaki, and Takeshi Tokuyama. Greedily finding a dense subgraph. In Rolf G. Karlsson and Andrzej Lingas, editors, *Algorithm Theory - SWAT '96, 5th Scandinavian Workshop on Algorithm Theory, Reykjavík, Iceland, July 3-5, 1996, Proceedings*, volume 1097 of *Lecture Notes in Computer Science*, pages 136–148. Springer, 1996. doi:10.1007/3-540-61422-2_127.

4 Bahman Bahmani, Ravi Kumar, and Sergei Vassilvitskii. Densest subgraph in streaming and mapreduce. *PVLDB*, 5(5):454–465, 2012. doi:10.14778/2140436.2140442.

5 Oana Denisa Balalau, Francesco Bonchi, T.-H. Hubert Chan, Francesco Gullo, and Mauro Sozio. Finding subgraphs with maximum total density and limited overlap. In Xueqi Cheng, Hang Li, Evgeniy Gabrilovich, and Jie Tang, editors, *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015*, pages 379–388. ACM, 2015. doi:10.1145/2684822.2685298.

6 Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In Klaus Jansen and Samir Khuller, editors, *Approximation Algorithms for Combinatorial Optimization, Third International Workshop, APPROX 2000, Proceedings*, volume 1913 of *Lecture Notes in Computer Science*, pages 84–95. Springer, 2000. doi:10.1007/3-540-44436-X.

7 Riccardo Dondi, Mohammad Mehdi Hosseinzadeh, Giancarlo Mauri, and Italo Zoppis. Top-k overlapping densest subgraphs: Approximation and complexity. In *Proceedings of the 20th Italian Conference on Theoretical Computer Science, ICTCS 2019, Como, Italy, September 9-11, 2019*, pages 110–121, 2019.

8 Lata Dyaram and T. J. Kamalanabhan. Unearthed: The other side of group cohesiveness. *Journal of Social Sciences*, 10(3):185–190, 2005.

9 Uriel Feige, Guy Kortsarz, and David Peleg. The dense k-subgraph problem. *Algorithmica*, 29(3):410–421, 2001. doi:10.1007/s004530010050.

10 Eugene Fratkin, Brian T. Naughton, Douglas L. Brutlag, and Serafim Batzoglou. Motifcut: regulatory motifs finding with maximum density subgraphs. *Bioinformatics*, 22(14):156–157, 2006. doi:10.1093/bioinformatics/btl243.

11 Esther Galbrun, Aristides Gionis, and Nikolaj Tatti. Top-k overlapping densest subgraphs. *Data Min. Knowl. Discov.*, 30(5):1134–1165, 2016. doi:10.1007/s10618-016-0464-z.

12 Michael R Garey and David S Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* WH Freeman & Co., 1979.

13 Andrew V. Goldberg. Finding a maximum density subgraph. Technical report, Berkeley, CA, USA, 1984.

14 Doron Goldstein and Michael Langberg. The dense k subgraph problem. *CoRR*, abs/0912.5327, 2009. arXiv:0912.5327.

15 Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972.

16 Samir Khuller and Barna Saha. On finding dense subgraphs. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris E. Nikoletseas, and Wolfgang Thomas, editors, *Automata, Languages and Programming, 36th International Colloquium, ICALP 2009, Rhodes, Greece, July 5-12, 2009, Proceedings, Part I*, volume 5555 of *Lecture Notes in Computer Science*, pages 597–608. Springer, 2009. doi:10.1007/978-3-642-02927-1_50.

17 Christian Komusiewicz. Multivariate algorithmics for finding cohesive subnetworks. *Algorithms*, 9(1):21, 2016.

18 Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. Trawling the web for emerging cyber-communities. *Computer Networks*, 31(11-16):1481–1493, 1999. doi:10.1016/S1389-1286(99)00040-7.

19 Jure Leskovec, Kevin J. Lang, Anirban Dasgupta, and Michael W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009. doi:10.1080/15427951.2009.10129177.

20 Muhammad Anis Uddin Nasir, Aristides Gionis, Gianmarco De Francisci Morales, and Sarunas Girdzijauskas. Fully dynamic algorithm for top-k densest subgraphs. In Ee-Peng Lim, Marianne Winslett, Mark Sanderson, Ada Wai-Chee Fu, Jimeng Sun, J. Shane Culpepper, Eric Lo, Joyce C. Ho, Debora Donato, Rakesh Agrawal, Yu Zheng, Carlos Castillo, Aixin Sun, Vincent S. Tseng, and Chenliang Li, editors, *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017*, pages 1817–1826. ACM, 2017. doi:10.1145/3132847.3132966.

21 James B. Orlin. Max flows in o(nm) time, or better. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 765–774. ACM, 2013. URL: https://doi.org/10.1145/2488608.2488705, doi:10.1145/2488608.2488705.

22 Jean-Claude Picard and Maurice Queyranne. A network flow solution to some nonlinear 0-1 programming problems, with applications to graph theory. *Networks*, 12(2):141–159, 1982. URL: https://doi.org/10.1002/net.3230120206, doi:10.1002/net.3230120206.

23 Mauro Sozio and Aristides Gionis. The community-search problem and how to plan a successful cocktail party. In Bharat Rao, Balaji Krishnapuram, Andrew Tomkins, and Qiang Yang, editors, *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, July 25-28, 2010*, pages 939–948. ACM, 2010. doi:10.1145/1835804.1835923.

24 Nikolaj Tatti and Aristides Gionis. Density-friendly graph decomposition. In Aldo Gangemi, Stefano Leonardi, and Alessandro Panconesi, editors, *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, pages 1089–1099. ACM, 2015. doi:10.1145/2736277.2741119.

25 Elena Valari, Maria Kontaki, and Apostolos N. Papadopoulos. Discovery of top-k dense subgraphs in dynamic graph collections. In Anastasia Ailamaki and Shawn Bowers, editors, *Scientific and Statistical Database Management - 24th International Conference, SSDBM 2012, Chania, Crete, Greece, June 25-27, 2012. Proceedings*, volume 7338 of *Lecture Notes in Computer Science*, pages 213–230. Springer, 2012.

26 Zhaonian Zou. Polynomial-time algorithm for finding densest subgraphs in uncertain graphs. In *Proceedings of Internation Workshop on Mining and Learning with Graphs*, 2013.

27 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(1):103–128, 2007. doi:10.4086/toc.2007.v003a006.

Appendix

Proof of Lemma 5

▶ **Lemma 5.** *A cut (S,T) of $H_{u,z}$ is such that $w(S,T) < \infty$ iff $u_a, u_b \in S$ and $z_a, z_b \in T$.*

Proof. Observe that by our construction, one of u_a, u_b is in T (one of z_a,z_b is in S, respectively) iff one of the ∞-weight arcs (s, u_a) or (s, u_b) (one of the arcs (z_a, t) or (z_b, t), respectively) belongs to the cut (S,T), and $w(S,T) = \infty$. ◀

Proof of Lemma 7

▶ **Lemma 7.** $w(\{s\} \cup B_S, \{t\} \cup B_T) = mn + 2n_2(d/2 - d_2)$.

Proof. Similar to the previous lemma, if $B_T = \emptyset$ we have $w(\{s\} \cup B_S, \{t\} \cup B_T) = mn$, and otherwise

$$\begin{aligned} w(\{s\} \cup B_S, \{t\} \cup B_T) &= w(B_S, \{t\}) + w(\{s\}, B_T) + w(B_S, B_T) \\ &= m|B_S| + (m+d)|B_T| - \sum_{v_b \in B_T} deg(v) + |E(B_S, B_T)| \\ &= m|B| + d|B_T| - \sum_{v_b \in B_T} deg(v) + |E(B_S, B_T)| \\ &= mn + 2n_2(d/2 - d_2). \end{aligned}$$

◀

Proof of Lemma 8

▶ **Lemma 8.** *$w(A_S, B_T) = 0$ iff $V_1 \cap V_2 = \emptyset$, and otherwise $w(A_S, B_T) = \infty$.*

Proof. By construction of $H_{u,v}$, the only arcs between A_S and B_T are arcs (v_a, v_b) with weight ∞ between two copies of some vertex v of G. Thus, the two vertex subsets $V_1 = \{v \in V : v_a \in A_S\}$ and $V_2 = \{v \in V : v_b \in B_T\}$ in G are disjoint iff $w(A_S, B_T) = 0$, and otherwise $w(A_S, B_T) = \infty$. ◀

Proof of Lemma 10

▶ **Lemma 10.** *Let $G = (V, E)$ be a graph. Then the vertex set of G can be partitioned into three cliques iff G has three disjoint subgraphs with total densities at least $(|V| - 3)/2$.*

Proof. Suppose V can be partitioned into three sets $V = V_1 \cup V_2 \cup V_3$, where $G_i = G[V_i]$ is a clique for each $i \in \{1, 2, 3\}$. Then as G_i is a clique, we have $density(G_i) = (|V_i| - 1)/2$, and so in total we have

$$\sum_{i=1}^{3} density(G_i) = \sum_{i=1}^{3} \frac{|V_i| - 1}{2} = \frac{|V| - 3}{2}.$$

Conversely, suppose G has three disjoint subgraphs G_1, G_2, and G_3 with total densities at least $(|V| - 3)/2$. Let n_i denote the number of vertices in G_i for each $i \in \{1, 2, 3\}$. Then the density of G_i is maximized when G_i is a clique, in which case its density is $(n_i - 1)/2$, and so $density(G_i) \leq (n_i - 1)/2$. Thus,

$$\sum_{i=1}^{3} density(G_i) \leq \sum_{i=1}^{3} (n_i - 1)/2 \leq (|V| - 3)/2$$

where equality is attained only in the case where each G_i is indeed a clique and $V_1 \cup V_2 \cup V_3 = V$. ◀

Proof of Claim 11

▷ Claim 11. $density(H_0) \geq d_1$.

Proof. Since $density(H_0) \geq density(H)$, it holds that:

$$\begin{aligned} density(H_0) &\geq density(H) && \iff \\ \frac{m_0}{n_0} &\geq \frac{m_0 + m_1}{n_0 + n_1} && \iff \\ m_0 n_1 &\geq m_1 n_0 && \iff \\ \frac{m_0}{n_0} &\geq \frac{m_1}{n_1} && \iff \\ density(H_0) &\geq d_1, \end{aligned}$$

thus concluding the proof. ◀

Proof of Lemma 17

▶ **Lemma 17.** *Let $v \in V$. A densest subgraph of G that includes v can be computed in $O(mn \lg n)$ time.*

Proof. Let $w_v : V \to \mathbb{N}$ be the weight function defined by $w_v(v) = n$, and $w_v(u) = 1$ for all vertices $u \neq v$. Then any subgraph of G that does not include v has weighted density less than n, and any subgraph that includes v has weight density at least n. It follows that computing a densest subgraph of G that includes v can be done by a single application of Goldberg's algorithm in $O(mn \lg n)$ time on G weighted by w_v. ◀

Proof of Lemma 20

▶ **Lemma 20.** *If $V_{\ell+1} \subseteq V^*$ then G_ℓ can be computed in $O(2^k mn^3 \lg n)$ time.*

Proof. We iterate over all possible $2^\ell - 1$ non-empty subsets $\mathcal{C} \subseteq \{V_1, \ldots, V_\ell\}$. For each subset $\mathcal{C}$, we iterate over all $O(n^2)$ vertices $v_{in}, v_{out} \in V_\mathcal{C}$ and compute a densest subgraph in $G[V_\mathcal{C} \setminus \{v_{out}\}]$ that includes v_{in} (using the algorithm in Lemma 17). This requires $O(mn^3 \lg n)$ time in total. Out of all subgraphs computed this way, along with all subgraphs of the form $G[V_\mathcal{C}]$, we choose the densest subgraph which is distinct from $\{G_1, \ldots, G_\ell\}$. As $G_{\ell+1}$ is a densest subgraph in $G[V_\mathcal{C} \setminus \{v_{out}\}]$ that includes v_{in}, for the minimal cover $\mathcal{C}$ of $V_{\ell+1}$ and some $v_{in}, v_{out} \in V_\mathcal{C}$ (according to Lemma 19), this algorithm is indeed guaranteed to find a subgraph of G with density at least $density(G_{\ell+1})$. ◀

Proof of Lemma 21

▶ **Lemma 21.** *Let $G[V']$ be a subgraph of G of density d. Consider a set of vertices X, disjoint from V' such that $d_x = \frac{|E(X) \cup E(X,V')|}{|X|}$. If $d_x \geq d$, then $G[V' \cup X]$ has density at least d.*

Proof. Consider the density d' of $G[V' \cup X]$, defined as:

$$d' = \frac{|E(V')| + |E(X)| + |E(X, V')|}{|V'| + |X|}$$

Then

$$d' - d = \frac{|E(V')| + |E(X)| + |E(X, V')|}{|V'| + |X|} - \frac{E(V')}{|V'|} =$$

$$\frac{|V'|(|E(V')| + |E(X)| + |E(X, V')|) - (|V'| + |X|)E(V')}{|V'|(|V'| + |X|)} =$$

$$\frac{|V'||E(X)| + |V'||E(X, V')| - |X||E(V')|}{|V'|(|V'| + |X|)}$$

Now, since

$$d_x = \frac{|E(X)| + |E(X, V')|}{|X|} \geq d = \frac{|E(V')|}{|V'|}$$

it holds

$$\frac{|V'||E(X)| + |V'||E(X, V')| - |X||E(V')|}{|V'|(|V'| + |X|)} \geq 0$$

thus concluding the proof. ◀

Proof of Claim 23

▷ Claim 23. $density(G[Y_i]) \geq density(G_i)$, for each i with $1 \leq i \leq 2$.

Proof. Notice that by adding a vertex $u \in V_c$ to Y_i, the density of Y_i is not decreased with respect to the density of G_i, hence we assume in what follows that $V_c \subseteq Y_i$, with $1 \leq i \leq 2$. We start by showing an upper bound on the density of $G[Y_i]$, that is $density(Y_i) < n$. Notice that $|Y_i| \leq 2n$, hence the maximum density of $G[Y_i]$ (when $G[Y_i]$ is a clique of size $2n$) is $\frac{\frac{1}{2}2n(2n-1)}{2n} < n$.

Now, consider the vertices added by Step 1. Since each vertex $u \in V_B$ is adjacent to at least n vertices of Y_i (the vertices in V_c), it follows by Lemma 21 that the vertices added by Step 1 do not decrease the density of $G[Y_i]$ with respect to G_i.

Now, consider the vertices added by Step 2. Each vertex v removed from Y_2 and added to Y_1 corresponds to a vertex of $u \in V_c$ added to Y_1. The removal of vertex u decreases the density of Y_2 of at most $\frac{2n-1}{\frac{3}{2}n} = \frac{4}{3} - \frac{2}{3n}$. Vertices u and v to Y_1 contributes to the density of Y_1 for at least $\frac{2n-1+\frac{n}{2}}{\frac{3}{2}n} = \frac{5}{3} - \frac{2}{3n}$. It follows that the vertices added by Step 2 do not decrease the density of $G[Y_i]$ with respect to G_i.

◀

Proof of Corollary 24

▶ **Corollary 24.** *Let $G_B = (V_B, E_B)$ be an input graph of* Minimum Bisection *and let G be the corresponding instance of* 2-Overlapping Densest Subgraphs. *Consider an optimal solution $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ of* 2-Overlapping Densest Subgraphs. *Then $V_1 \cup V_2 = V$.*

Proof. The proof follows from Lemma 22, since $V_c \subseteq V_1, V_2$ and the density of Y_i, with $1 \leq i \leq 2$, is less than n (see the proof of Lemma 22). By Lemma 21 it follows that adding a vertex $u \notin V_1 \cup V_2$ to one of G_1 or G_2 does not decrease its density, since u it is connected to each vertex of V_c. ◀

Proof of Lemma 25

▶ **Lemma 25.** *Let $G_B = (V_B, E_B)$ be an input graph of* Minimum Bisection *and let G be the corresponding instance of* 2-Overlapping Densest Subgraphs*. Consider a solution $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ of* 2-Overlapping Densest Subgraphs *on instance G that satisfies Lemma 22 and Corollary 24. Then $(V_1 \cap V_2) = V_c$ and $|V_1| = |V_2|$.*

Proof. Consider a solution G_1, G_2 of 2-Overlapping Densest Subgraphs on instance G that satisfies Lemma 22 and Corollary 24. It follows that $V_c \subseteq V_1, V_2$ and that $V_1 \cup V_2 = V$. Assume that $V_1 \cap V_2 = V_c \cup X$, with $X \neq \emptyset$, $|V_1| \leq |V_2|$ and $|V_1 \cap V_B| = |X| + z$, where $z \leq \frac{n}{2}$, since $|V_1| \leq |V_2|$. It follows that

$$\frac{|V_1 \cap V_2|}{|V_1|} = \frac{n + |X|}{n + |X| + z} =$$

$$\frac{\frac{2}{3}(n + |X| + z) + \frac{1}{3}(n + |X|) - \frac{2}{3}z}{n + |X| + z} = \frac{2}{3} + \frac{\frac{1}{3}(n + |X|) - \frac{2}{3}z}{n + |X| + z}.$$

Since $z \leq \frac{n}{2}$, it follows that $\frac{2}{3}z \leq \frac{1}{3}n$, thus

$$\frac{\frac{1}{3}(n + |X|) - \frac{2}{3}z}{n + |X| + z} \geq \frac{\frac{1}{3}(n + |X|) - \frac{1}{3}n}{n + |X| + z} = \frac{\frac{1}{3}|X|}{n + |X| + z}$$

Since $|X| > 0$, $\frac{\frac{1}{3}|X|}{n+|X|+z} > 0$, hence

$$\frac{|V_1 \cap V_2|}{|V_1|} > \frac{2}{3}.$$

Thus $V_1 \cap V_2 = V_c$. Moreover, $|V_1| = |V_2| = \frac{3}{2}n$, otherwise the constraints $\frac{|V_1 \cap V_2|}{|V_1|} \leq \frac{2}{3}$ and $\frac{|V_1 \cap V_2|}{|V_2|} \leq \frac{2}{3}$ would not be satisfied. ◀

Proof of Theorem 26

▶ **Theorem 26.** *Let $G_B = (V_B, E_B)$ be an input graph of* Minimum Bisection *and let G be the corresponding instance of* 2-Overlapping Densest Subgraphs*. Then there exists a solution of* Minimum Bisection *on G_B that cuts h edges if and only if there exists a solution of* 2-Overlapping Densest Subgraphs *on instance G having density $\frac{2n^2 - n + m - h}{\frac{3}{2}n}$.*

Proof. Consider a solution $(V_{B,1}, V_{B,2})$ of Minimum Bisection and define a solution $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ of 2-Overlapping Densest Subgraphs on instance G as follows: $V_i = V_c \cup V_{B,i}$, with $1 \leq i \leq 2$. Now, denote by e_i the number edges having both endpoints in $V_{B,i}$. It holds that

$$density(G_1) + density(G_2) = \frac{n(n-1) + n^2 + e_1 + e_2}{\frac{3}{2}n}$$

and, since $h = m - (e_1 + e_2)$, the overall density is

$$density(G_1) + density(G_2) = \frac{2n^2 - n + m - h}{\frac{3}{2}n}$$

Consider now a solution $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ of 2-Overlapping Densest Subgraphs on instance G. By Lemma 22, Corollary 24, Lemma 25 we can assume that $V_c \subseteq V_1, V_2$, $V_1 \cap V_2 = V_c$ and $|V_1| = |V_2|$. This implies $|V_1| = |V_2| = \frac{3}{2}n$. Define a solution of Minimum Bisection as follows: $V_{B,i} = V_i \setminus V_c$, with $1 \leq i \leq 2$. If $m - h$ edges have both endpoints in either V_1 or V_2, then h edges are cut by $V_{B,1}$ and $V_{B,2}$, thus concluding the proof. ◀