Structural Parameterization of Cluster Deletion

Giuseppe Italiano, Athanasios Konstantinidis, Charis Papadopoulos

Structural Parameterization of Cluster Deletion

Giuseppe F. Italiano[1][0000−0002−9492−9894]⋆, Athanasios L. Konstantinidis[1], and Charis Papadopoulos[2][0000−0001−5556−2981]⋆⋆

[1] LUISS University, Rome, Italy gitaliano@luiss.it akonstantinidis@luiss.it
[2] Department of Mathematics, University of Ioannina, Greece charis@uoi.gr

Abstract. In the WEIGHTED CLUSTER DELETION problem we are given a graph with non-negative integral edge weights and the task is to determine, for a target value k, if there is a set of edges of total weight at most k such that its removal results in a disjoint union of cliques. It is well-known that the problem is FPT parameterized by k, the total weight of edge deletions. In scenarios in which the solution size is large, naturally one needs to drop the constraint on the solution size. Here we study WEIGHTED CLUSTER DELETION where there is no bound on the size of the solution, but the parameter captures structural properties of the input graph. Our main contribution is to classify the parameterized complexity of WEIGHTED CLUSTER DELETION with three structural parameters, namely, vertex cover, twin cover and neighborhood diversity. We show that the problem is FPT when parameterized by the vertex cover, whereas it becomes paraNP-hard when parameterized by the twin cover or the neighborhood diversity. To illustrate the applicability of our FPT result, we use it in order to show that the unweighted variant of the problem, CLUSTER DELETION, is FPT parameterized by the twin cover. This is the first algorithm with single-exponential running time parameterized by the twin cover. Interestingly, we are able to achieve an FPT result parameterized by the neighborhood diversity that involves an ILP formulation. In fact, our results generalize the parameterized setting by the solution size, as we deduce that both parameters, twin cover and neighborhood diversity, are linearly bounded by the number of edge deletions.

Keywords: Cluster deletion problem · twin cover · neighborhood diversity

1 Introduction

Highly-connected parts of complex systems reveal clustering data that are important in numerous application fields such as computational biology [2] and

⋆ Partially supported by MUR, the Italian Ministry for University and Research, under PRIN Project AHeAD (Efficient Algorithms for HArnessing Networked Data).
⋆⋆ Supported by the Hellenic Foundation for Research and Innovation (H.F.R.I.) under the "First Call for H.F.R.I. Research Projects to support Faculty members and Researchers and the procurement of high-cost research grant", Project FANTA (eFficient Algorithms for NeTwork Analysis), number HFRI-FM17-431.

machine learning [1,25]. In graph-theoretic terms, those dense homogeneous sets are often identified as cliques. A core algorithmic theme that has received considerably interest is to modify a given graph as little as possible in order to reveal disjoint cliques. In the Cluster Deletion problem we seek to delete the minimum number of edges of a given graph such that the resulting graph is a vertex-disjoint union of cliques (cluster graph). Here we also consider its natural variant with weights on the edges of the graph, named weighted Cluster Deletion: each edge has an associated non-negative weight and the goal is to minimize the sum of the weights of the removed edges. It is known that the problems are NP-hard on general graphs [26] and settling their complexity status even on restricted settings has attracted several researchers.

With regards to parameterized complexity, the general result by Cai [6] shows that (weighted) Cluster Deletion is FPT parameterized by the number of deleted edges. Considering the same parameter, it is known that the problem admits a linear kernel [7] and recently, Cao et al. [8] devised a different, still linear, kernel. In particular, both variations of the problem admit several fast FPT algorithms parameterized by the solution size [3,8,27]. However, as with the principle of parameterized complexity, such algorithms are efficient whenever the considered parameter is rather small. Combined with the light of lower bounds refuting the existence of subexponential FPT algorithms [20], it seems reasonable to study different distance measures. If the remaining edges inside the cliques are used as a parameter then the unweighted variant of the problem has shown to be FPT and does not admit a polynomial kernel [18]. In contrast, by considering either the maximum degree or the diameter of the graph as a parameter of the problem, paraNP-hardness results occur. More specifically, Komusiewicz and Uhlmann [20] showed that Cluster Deletion is NP-hard on graphs of maximum degree 4, but it is polynomial-time solvable on graphs with maximum degree 3. Further, Cluster Deletion is NP-hard on P_5-free graphs [5,23], although there is a polynomial algorithm that computes an optimal solution on P_4-free graph [16]. Interestingly, Cluster Deletion is FPT when parameterized by the size of a minimum cluster vertex deletion set [21].

Naturally, the weighted variant of the problem may behave differently than the unweighted on the same class of graphs. For instance, weighted Cluster Deletion is NP-hard even on P_4-free graphs and split graphs [5]. Apart from some restricted subclasses of chordal graphs for which the problem can be solved in polynomial time [5], the weighted variant of the problem has received less interest when parameterized by distance measures other than the solution size. Our focus is to complement existing results and analyze both variations of the problem under graph structural parameters. To capture the powerfulness of such parameters, we consider generalizations of the vertex cover number. These type of parameterizations proved to be successful in a wide range of problems [4,14,15,24]. Here we exploit their impact towards the (weighted) Cluster Deletion problem.

Our results We consider parameterizations of the problem with respect to structural properties of the given graph. As (weighted) Cluster Deletion

admits fast algorithms parameterized by the solution size, it is natural to consider variations of the vertex cover number such as the twin cover and the neighborhood diversity. We note that both notions constitute generalizations of the vertex cover number [15,24], though there is no relation among them.

We first show that both parameters are linearly upper-bounded in the number of edge deletions required to obtain a cluster graph. Thus we explore further venues to attack Cluster Deletion, since the unweighted and weighted variations of the problem were already known to admit fast parameterized algorithms by the solution size [6,8,17,27].

As an initial point, we establish that weighted Cluster Deletion is paraNP-hard when parameterized by the twin cover or the neighborhood diversity. This is achieved through an interesting reduction from a terminal cut problem with a small number of terminals.

Theorem 1.1. weighted Cluster Deletion *is NP-hard on graphs with twin cover number at most three and graphs with neighborhood diversity at most two.*

Based on this negative result, we also consider the more restrictive vertex cover number as a structural parameter. The vertex cover number is unrelated to the solution size. However, with our next algorithm, vertex cover can be considered as one of the few parameters for which the weighted variant admits a positive result. Our technique relies on carefully applying a dynamic programming approach that handles vertices that lie outside the vertex cover. We note that Cluster Deletion is expressible in monadic second order logic (MSO_2) as explicitly given in [22]. Thus weighted Cluster Deletion is FPT when parameterized by treewidth [9]. However, we are not aware if such an approach leads to a single-exponential running time, as we deduce for the vertex cover.

Theorem 1.2. weighted Cluster Deletion *can be solved in* $2^{O(\mathsf{vc})} \cdot O(n^2)$ *time, where* vc *is the vertex cover number of the input graph.*

To illustrate the wider applicability of the algorithm given in Theorem 1.2, we turn our attention to the unweighted variant of Cluster Deletion. Twin cover introduced by Ganian [15] generalizes vertex cover in the sense that vertices outside the cover set form an independent set or a true twin class. Our approach for Cluster Deletion relies on carefully contracting true twin classes that lie outside the cover set. It turns out that this process results in an edge-weighted graph of bounded vertex cover. Then we apply the algorithm given in Theorem 1.2 for the weighted Cluster Deletion problem.

Theorem 1.3. Cluster Deletion *can be solved in* $2^{O(\mathsf{tc})} \cdot O(n^2)$ *time, where* tc *is the twin cover number of the input graph.*

It should be noted that the FPT membership given in Theorem 1.3, can be obtained with a different approach, though with a worse running time. In particular, one could use the *cluster vertex deletion* number (also known as *distance to cluster*) which stands for the number of deleted vertices that is required to obtain a cluster graph. Doucha and Kratochvíl [12] showed that for any graph

G, the cluster vertex deletion number of G is at most the twin cover number of G. Combined with the fact that Cluster Deletion is FPT parameterized by the cluster vertex deletion number [21], we get an algorithm with running time $2^{O(\mathsf{tc}\log\mathsf{tc})} \cdot n^{O(1)}$. Thus Theorem 1.3 reveals the first FPT algorithm with single-exponential running time. Moreover, we believe that our algorithm is interesting on its own because it exploits further connections between the two variations of the problem.

Regarding the neighborhood diversity which was introduced by Lampis [24], we use a completely different approach. This notion is based on true twin and false twin classes of vertices. We use integer linear programming (ILP) as a subroutine in our FPT algorithm. In particular, we translate part of our problem as an instance of choosing sufficient maximum cliques in an auxiliary graph of bounded size. As the size is bounded, we show that the formulation to an ILP problem with bounded number of variables is feasible.

Theorem 1.4. Cluster Deletion *can be solved in* $2^{2^{O(\mathsf{nd})}} \cdot n^{O(1)}$ *time, where* nd *is the neighborhood diversity of the input graph.*

2 Preliminaries

All graphs considered here are simple and undirected. For $S \subseteq V$, $N(S) = \bigcup_{v\in S} N(v) \setminus S$ and $N[S] = N(S) \cup S$. For $X \subseteq V(G)$, the subgraph of G *induced* by X, $G[X]$, has vertex set X, and for each vertex pair u, v from X, uv is an edge of $G[X]$ if and only if $u \neq v$ and uv is an edge of G. For $R \subseteq E(G)$, $G \setminus R$ denotes the graph $(V(G), E(G) \setminus R)$, that is a subgraph of G and for $S \subseteq V(G)$, $G - S$ denotes the graph $G[V(G) - S]$, that is an induced subgraph of G. For two disjoint sets of vertices A and B, we write $E(A, B)$ to denote the edges that have one endpoint in A and one endpoint in B. A *matching* in G is a set of edges having no common endpoint. A *cluster graph* is a graph in which every connected component is a clique. *Contracting* a set of vertices S is the operation of substituting the vertices of S by a new vertex w with $N(w) = N(S)$. We next formalize Cluster Deletion, as a decision problem.

Cluster Deletion
Input: A graph $G = (V, E)$ and a non-negative integer k.
Task: Decide whether there is $E' \subseteq E(G)$ such that $G \setminus E'$ is a cluster graph and $\lvert E'\rvert \leq k$.

In the optimization setting, the task of Cluster Deletion is to turn the input graph G into a cluster graph by deleting the minimum number of edges. We describe a solution of Cluster Deletion in two equivalent ways: either a solution is given as a set of edges $E' \subseteq E(G)$ such that $G \setminus E'$ is a cluster graph, or it is given as a vertex partition $S = \{C_1, \ldots, C_t\}$ of $V(G)$ such that each $G[C_i]$ is a clique. The equivalence follows because the connected components of the cluster graph $G \setminus E'$ correspond to the induced subgraphs $G[C_i]$, so that the set $E' = \bigcup E(C_i, C_j)$ where $C_i, C_j \in S$ forms the required set of deleted edges.

Let $S = \{C_1, \ldots, C_t\}$ be a solution of Cluster Deletion such that each $G[C_i]$ is a clique. In such terms, the problem can be viewed as a vertex partition

problem into $C_1, \ldots, C_t$. Each C_i is called *cluster*. Edgeless clusters, i.e., clusters containing exactly one vertex, are called *trivial clusters*. An *optimal solution* S for Cluster Deletion is a clique partition of G such that the number of edges $E(G) \setminus E(S)$ is minimum, where $E(S)$ stands for the set of edges $\bigcup E(C_i)$.

For the edge-weighted variation, every edge of the input graph admits a cost of deletion (represented by a weight function w) and the task is to perform the minimum sum of weights deletions. Given a subset E' of edges, we let $\mathsf{w}(E') = \sum_{e \in E'} \mathsf{w}(e)$, for the ease of notation. Hereafter we assume that the edge weights are positive integers. The reason of not considering negative weights comes from the fact that any graph can be completed into a clique with arbitrary edge-weights, so that weighted Cluster Deletion becomes trivially difficult even on graphs with sufficiently large cliques.

weighted Cluster Deletion
Input: A graph $G = (V, E)$, a weight function $\mathsf{w} : E(G) \to \mathbb{Z}^+$, and a non-negative integer k.
Task: Decide whether there is $E' \subseteq E(G)$ such that $G \setminus E'$ is a cluster graph and $\mathsf{w}(E') \leq k$.

Notice that if all edge weights are equal to one then the two variants of the problem coincide. Moreover, positive results propagate from weighted Cluster Deletion towards Cluster Deletion, whereas negative results propagate in the reverse order. However, a notable difference among the two problems is the aspect of computing a minimum solution. Indeed, the formal description of the edge-weighted problem is not sufficient to find a minimum weight solution. Despite this fact, we point out that our positive results concerning both problems are able to compute an optimal solution with an additional polynomial factor on the stated running times.

Next we provide some useful properties concerning twin vertices. Two adjacent vertices u and v are called *true twins* if $N[u] = N[v]$, whereas two non-adjacent vertices x and y are called *false twins* if $N(x) = N(y)$. A *true twin class* of G is a maximal set of vertices that are pairwise true twins. Note that the set of true twin classes of G constitutes a partition of $V(G)$. We denote by $\mathcal{T}(G) = \{T_1, \ldots, T_r\}$ the true twin classes of G, so that each set of vertices T_i forms a true twin class in G. Observe that the partition $\mathcal{T}(G) = \{T_1, \ldots, T_r\}$ of $V(G)$ into classes of true twins can be constructed in linear time.

Lemma 2.1 ([5]). *Let x and y be true twin vertices in G. Then, in any optimal solution for* Cluster Deletion *x and y belong to the same cluster.*

It is not difficult to extend Lemma 2.1 for a set of true twin vertices.

Observation 2.1. *The vertices of a true twin class of G belong to the same cluster in any optimal solution for* Cluster Deletion.

We should point out that the above characterization does not hold for the edge-weighted variant of the problem, even if we relax the restriction to certain (rather than *any*) optimal solutions. However the following result holds.

Lemma 2.2. *Let X be a true twin class of G such that all edges incident to the vertices of X have the same positive weight. Then the vertices of X belong to the same cluster in any optimal solution for* WEIGHTED CLUSTER DELETION.

Graph parameters A *vertex cover* of G is a set of vertices that includes at least one endpoint of every edge of the graph. The *vertex cover number*, denoted by $\mathsf{vc}(G)$, is the size of a minimum cardinality vertex cover in G. Notice that a set of vertices X is a vertex cover if and only if $V(G) \setminus X$ is an independent set. Unfortunately, vertex cover is a rather restrictive graph parameter and, for that reason the following generalizations have been proposed. The twin cover of a graph has been introduced by Ganian [15] as follows.

Definition 2.1. *A set of vertices $X \subseteq V(G)$ is a* twin cover *of G if for every edge $uv \in E(G)$ either one of the following holds: (i) $u \in X$ or $v \in X$, (ii) u and v are true twin vertices. Then G has twin cover number* tc *if* tc *is the minimum possible size of a twin cover of G.*

It is known that if a minimum twin cover in G has size at most k, then it is possible to find a twin cover of size k in time $O(|E| + k|V| + 1.2738^k)$ [15].

Another generalization of vertex cover is the neighborhood diversity which has been defined by Lampis [24]. Two vertices x, y of G have the *same type* if $N(x) \setminus \{y\} = N(y) \setminus \{x\}$. The relation of having the same type is an equivalence. In particular, two vertices x and y have the same type if and only if x and y are either true twin or false twin vertices.

Definition 2.2. *A graph $G = (V, E)$ has neighborhood diversity at most* nd*, if there exists a partition of $V(G)$ into at most* nd *sets, such that all vertices in each set have the same type.*

Observe that the vertices of a given type not only have the same (closed) neighborhood in G, but also form either a clique or an independent set in G. Moreover, it is useful to consider a *type graph* H of a graph G, in which every node V_i of H is a type class of G and two such nodes V_i, V_j are adjacent in H if and only if $uv \in E(G)$ for $u \in V_i$ and $v \in V_j$. There exists an algorithm which runs in polynomial time and given a graph $G = (V, E)$ finds a minimum partition of $V(G)$ into neighborhood types [24].

Notice here that there are graphs that have bounded twin cover and unbounded neighborhood diversity, and vice versa (see for e.g., [15]). Moreover, twin cover and neighborhood diversity are incomparable with treewidth (tw) but more restrictive than cliquewidth (cw) [15,24].

We now relate the above mentioned parameters with the number k of deleted edges for CLUSTER DELETION. We consider connected graphs, because any solution for CLUSTER DELETION or WEIGHTED CLUSTER DELETION of a disconnected graph G is obtained by the union of partial solutions on each connected component of G. Regarding vc there are simple examples for which $k = O(n)$ and $\mathsf{vc} = O(1)$, whereas other examples exist to show the opposite situation: a star graph is typical example for the former case and a graph consisting of

two vertex-disjoint cliques with an additional edge is an example for the latter case. Thus k and vc are unrelated. Notice that this comes in contrast to similar relations with respect to the *cluster vertex deletion* number (also known as *distance to cluster*) which stands for the number of deleted vertices that is required to obtain a cluster graph. It is not difficult to see that the cluster vertex deletion number is at most $2k$. Let us now show that nd and tc are both linearly upper-bounded in k.

Proposition 2.1. *Let G be a connected graph and let H be a cluster subgraph of G with $k = |E(G) \setminus E(H)|$. Then, $\mathsf{nd}(G) \leq 3k + 1$ and $\mathsf{tc}(G) \leq 2k$.*

3 Algorithmic results for WEIGHTED CLUSTER DELETION

In this section, we present our results on WEIGHTED CLUSTER DELETION when the parameter is the twin cover (tc) or neighborhood diversity (nd) or vertex cover (vc) of the given graph. We begin with the hardness result for the more general parameters tc and nd and then provide an efficient algorithm for the restricted parameter vc.

We obtain our result from the k-MULTIWAY CUT problem: given a graph $G = (V, E)$, a set $T = \{t_1, \ldots, t_k\} \subseteq V(G)$ of k terminals, and a non-negative integer ℓ, the task is to find a set of edges $F' \subseteq E(G)$ such that $|F'| \leq \ell$ and each terminal belongs to a separate connected component in $G \setminus F'$. It is known that k-MULTIWAY CUT problem remains NP-hard even if the number of terminals is three (i.e., $k = 3$) [11].

Theorem 3.1. WEIGHTED CLUSTER DELETION *is NP-hard on graphs with twin cover number at most three and graphs with neighborhood diversity at most two.*

Proof. We give a polynomial-time reduction to WEIGHTED CLUSTER DELETION on graphs with twin cover 3 or neighborhood diversity 2 from the NP-hard problem 3-MULTIWAY CUT. Let $(G = (V, E), T = \{t_1, t_2, t_3\}, \ell)$ be an instance of 3-MULTIWAY CUT where $|V(G)| = n$ and $|E(G)| = m$. We assume that G is connected and $G[T]$ is edgeless, because we can restrict to the connected components of G and any edge between terminals belongs to a solution. Starting from G, we construct a graph H by adding all necessary edges so that (i) $V(G) \setminus T$ is a clique and (ii) every vertex t of T is adjacent to every vertex of $V(G) \setminus T$. Observe that H has the same vertex set with G and contains all edges of the complete graph except the edges of the triangle among the three terminals. We assign the following edge-weight function for the edges of H: if $e \in E(G)$ then $\mathsf{w}(e) = n^2$; otherwise, $\mathsf{w}(e) = 1$.

Notice that H can be constructed in polynomial time and contains n vertices and $\frac{n(n-1)}{2} - 3$ edges. Since the vertices of $V(G) \setminus \{t_1, t_2, t_3\}$ form a clique and the neighborhood of t_1, t_2 and t_3 is exactly this clique, the vertices of the clique are true twins. Thus, by the definition of twin cover, we get that $\mathsf{tc}(H) = 3$, with a twin cover set $\{t_1, t_2, t_3\}$. Moreover, the vertices of H can be partitioned into a clique $V(G)$ and an independent set T such that every vertex of the independent

set T is adjacent to every vertex of the clique $V(G)$. Hence H not only has a bounded twin cover, but it also admits a neighborhood diversity exactly 2, since one type class is the clique $V(G) \setminus T$ and the other type class is the independent set T. Let $W = \ell \cdot n^2 + p$, where $p = \frac{n(n-1)}{2} - m - 3$. We prove that 3-Multiway Cut has a solution $F' \subseteq E(G)$ with $|F'| \leq \ell$ if and only if weighted Cluster Deletion has a solution $E' \subseteq E(H)$ with $\mathsf{w}(E') \leq W$. □

3.1 Vertex cover

Here we provide an FPT algorithm for the weighted Cluster Deletion problem parameterized by the vertex cover. As a consequence, notice that Cluster Deletion can be solved within the same running time.

Theorem 3.2. weighted Cluster Deletion *can be solved in* $2^{O(\mathsf{vc})} \cdot O(n^2)$ *time, where* vc *is the vertex cover number of the input graph.*

Proof. Let (G, k) be an instance of weighted Cluster Deletion and let X be a vertex cover of $G = (V, E)$ of size vc. The vertices of $V \setminus X$ form an independent set I. Let $S = \{C_1, \ldots, C_r\}$ be a solution for weighted Cluster Deletion. Observe that any cluster C_i contains at most one vertex from I. That is, $|C_i \cap I| \leq 1$, for every $1 \leq i \leq r$. Since $|X| = \mathsf{vc}$, there are at most vc vertices of I that belong to non-trivial clusters. We design an FPT algorithm that computes a solution by applying a dynamic programming scheme. For a set of vertices Y, we assign its total edge-weight $\mathsf{w}(Y)$ as $-\infty$ whenever $G[Y]$ is not a cluster graph and $\mathsf{w}(Y)$ is the sum of the edge-weights in $G[Y]$, otherwise.

We number the vertices of I in an arbitrary order $I = \{1, \ldots, |I|\}$. For technical reasons, we extend I by adding a vertex u in G that is non-adjacent to any vertex, so that $I' = I \cup \{u\}$. Let G be the resulting graph and let $I' = \{0, 1, \ldots, |I|\}$, assuming that u is numbered with zero. For the dynamic programming, we construct a table T as follows: given a subset X' of X and an integer $j \in \{0, 1, \ldots, |I|\}$, $T[X', j]$ denotes the total edge-weights of a maximum edge-weighted cluster subgraph of $G[X' \cup \{0, \ldots, j\}]$. Clearly $T[X, |I|]$ is the desired value for our problem. As a base case, observe that $T[\emptyset, j] = 0$ for any j. For the recurrence, we have the following equation:

$$T[X', j] = \max_{\emptyset \neq Y' \subseteq X'} \begin{cases} T[X' \setminus Y', 0] + w(Y'), & \text{if } j = 0 \\ T[X' \setminus Y', j-1] + w(Y' \cup \{j\}), & \text{otherwise.} \end{cases} \quad (1)$$

□

4 An application on twin cover

In this section, we illustrate how Theorem 3.2 can be applied in the more relaxed variation with no edge weights but in a more powerful setting. We consider twin cover number as a parameter for Cluster Deletion and we show that Cluster Deletion can be solved in FPT time under this parameterization. For

doing so, we apply a natural process related to the true twin vertices resulting in an edge-weighted graph of bounded vertex cover.

Even though we deal with the unweighted variant, we consider edge-weighted graphs in a natural way. If there is no weight function defined on the edges of a graph G, we assign a weight to each edge of G equal to one and assume that G is an edge-weighted graph. A true twin class T of an edge-weighted graph G, is called *1-class* if all edges incident to the vertices of T have weight one.

Definition 4.1 (T-contraction). *Given a 1-class T of G, we define the T-*contraction *of G as the edge-weighted graph H obtained from G by contracting T into a single vertex v_T s.t. all edges incident to v_T in H have weight $|T|$.*

Observe that after a T-contraction, H has $|V(G)| - |T| + 1$ vertices, whereas the total value of the new edge weights in H is at most $|E(G)|$.

Lemma 4.1. *Let T be a 1-class of a graph G and let H be the T-contraction of G. There is a solution $E_1 \subseteq E(G)$ for* weighted Cluster Deletion *on G with $\mathsf{w}(E_1) \leq k$ if and only if there is a solution $E_2 \subseteq E(H)$ for* weighted Cluster Deletion *on H with $\mathsf{w}(E_2) \leq k$.*

Theorem 4.1. Cluster Deletion *can be solved in $2^{O(\mathsf{tc})} \cdot O(n^2)$ time, where tc is the twin cover number of the input graph.*

Proof. Let (G, k) be an instance of Cluster Deletion and let X be a twin cover of $G = (V, E)$ of size $\mathsf{tc} = |X|$. By the definition of twin cover, the vertices of $V(G) \setminus X$ induce a disjoint union of cliques in G. In particular, observe that every connected component of $G - X$ is a clique and forms a 1-class in G. Based on this fact, it is not difficult to prove the following. Let $Y_1, \ldots, Y_p$ be the connected components of $G - X$ such that $|Y_i| \geq 2$, for every $1 \leq i \leq p$. Any Y_i-contraction results in an edge-weighted graph H_i in which the connected components of $H_i - X$ with at least one edge are $\{Y_1, \ldots, Y_p\} \setminus Y_i$, and every $Y' \in \{Y_1, \ldots, Y_p\} \setminus Y_i$ is a 1-class.

We apply a Y_i-contraction in an arbitrary order with respect to $Y_1, \ldots, Y_p$. The resulting graph H has vertex cover at most $|X|$: every connected component of $H - X$ has size one implying that the vertices of $H - X$ form an independent set. Therefore we can apply the weighted Cluster Deletion algorithm on the edge-weighted graph H given in Theorem 3.2. □

5 Cluster Deletion and neighborhood diversity

In this section, we provide an FPT algorithm for Cluster Deletion when parameterized by neighborhood diversity. We will use integer linear programming as a subroutine of our main result. In particular, we translate part of our problem to an instance of the p-Variable Integer Linear Programming Feasibility problem: given an $m \times p$ matrix A over $\mathbb{Z}$ and a vector $b \in \mathbb{Z}^m$, decide whether there is a vector $x \in \mathbb{Z}^p$ such that $Ax \leq b$. Lenstra [19] showed

that the above problem is FPT parameterized by p, while Frank and Tardos [13] showed that this algorithm can be made to run also in polynomial space. We will make use of these results, that we formally state next[3].

Theorem 5.1 ([13,19]). p-Variable Integer Linear Programming Feasibility *can be solved using* $O(p^{2.5p+o(p)} \cdot L)$ *arithmetic operations and space polynomial in* L*, where* L *is the number of bits in the input.*

Before giving the details of our FPT algorithm for Cluster Deletion when parameterized by neighborhood diversity, we describe the basic idea how to compute a solution for Cluster Deletion by using the type graph. Let G be a graph and let H be its type graph of size nd. Moreover, let $\{V_1, \ldots, V_{\mathsf{nd}}\}$ be the type classes of G, or equivalently, the nodes of H. In the beginning, we find all possible cliques (not necessarily maximal) of H by taking all the subsets of nodes of H that form cliques. For every clique H_i of H, it is possible to find a maximum clique in G that is induced by the vertices of the nodes which belong to H_i. Now, our task is to choose some cliques H_i of H and for each H_i to choose a specific number of maximum cliques contained in G. Those maximum cliques will be considered as clusters for the problem.

Theorem 5.2. Cluster Deletion *can be solved in* $2^{2^{O(\mathsf{nd})}} \cdot n^{O(1)}$ *time, where* nd *is the neighborhood diversity of the input graph.*

Proof. Let (G, k) be an instance of Cluster Deletion and let H be the type graph of G of size nd. By the definition of neighborhood diversity, the vertices of G can be partitioned in nd type classes. Let $\{V_1, \ldots, V_{\mathsf{nd}}\}$ be the type classes of G. For ease of notation, we let $G_i = G[V_i]$. In the forthcoming arguments, we assume that $S = \{S_1, \ldots, S_r\}$ is the set of clusters of an optimal solution for Cluster Deletion on (G, k).

Since the vertices of a type class that forms a clique are true twin vertices, they belong to exactly one cluster of the solution S by Lemma 2.2. On the other hand, the vertices of a type class of G that forms an independent set belong to different clusters by the definition of cluster. Hence, for any cluster $S_j \in S$ that contains a vertex of V_i, we have the properties: (i) $V_i \subseteq S_j$, if G_i is a clique and (ii) $|V_i \cap S_j| = 1$, if G_i is an independent set. Based on this, we define the following quantities:

$$\|V_i\|_F = \begin{cases} |V_i| \text{ if } G_i = \text{clique} \\ 1 \quad \text{if } G_i = \text{independent set} \end{cases} \qquad \|V_i\|_T = \begin{cases} 1 \quad \text{if } G_i = \text{clique} \\ |V_i| \text{ if } G_i = \text{independent set} \end{cases}$$

Now we focus on the type graph H of G. We consider all subgraphs of H that form a clique in H. Since every type class of G is a node in H, there are at most 2^{nd} cliques in H. Let $\mathcal{H} = \{H_1, \ldots, H_q\}$ be the set of cliques in H, where $q = |\mathcal{H}| \leq 2^{\mathsf{nd}}$. Given a clique $H_j \in \mathcal{H}$, we denote by $G[H_j]$ the graph induced

[3] In general the applicability of Theorem 5.1 has revealed close connections between FPT and ILP (see for e.g. [10]).

by the vertices of G that belong to the nodes of H_j. Formally, the set of vertices of $G[H_j]$ is exactly $\{u \in V_i \mid V_i \in V(H_j)\}$.

For every clique $H_j \in \mathcal{H}$, we define the following values: n_j is the number of vertices of a maximum clique in $G[H_j]$, m_j is the number of edges of a maximum clique in $G[H_j]$, $t_j = \min \|V_i\|_T$, for any $V_i \in V(H_j)$.

Claim 5.1 *For any $H_j \in \mathcal{H}$, a maximum clique S_j in $G[H_j]$ fulfils properties (i) and (ii) and contains the following number of vertices: $n_j = \sum_{V_i \in V(H_j)} \|V_i\|_F$.*

Thus, by Claim 5.1 we can compute in polynomial time both numbers n_j and m_j for every clique H_j. Similarly, t_j can be computed in time linear in the size of H_j. Now consider a maximum clique of $G[H_j]$ for a clique H_j. Observe that $G[H_j]$ may contain more than one maximum cliques that are vertex-disjoint. We capture this property by a non-negative integer y_j assigned to $H_j \in \mathcal{H}$, which describes some number of vertex-disjoint maximum cliques that belong to $G[H_j]$.

Given the set of cliques $\mathcal{H}$, for every node V_i of the type graph H we define $M(i)$ as the cliques of $\mathcal{H}$ that contain V_i. Formally, we have $M(i) = \{H_j \in \mathcal{H} \mid V_i \in V(H_j)\}$. Based on the number of cliques $q = |\mathcal{H}|$, we define a sequence $X = (x_1, \ldots, x_q)$ of non-negative integers and we say that X is *valid vector* of $\mathcal{H}$ if the following conditions hold: $0 \leq x_j \leq t_j$ and $\sum_{H_j \in M(i)} x_j = \|V_i\|_T$, for every $V_i \in V(H)$.

Furthermore, for a valid vector X of $\mathcal{H}$, we let $|E(X)|$ be the total number of edges if we choose x_j number of vertex-disjoint maximum cliques that belong to $G[H_j]$ for all $x_j \in X$. That is, $|E(X)| = \sum_{x_j \in X} x_j \cdot m_j$. With the next claim, our task is translated into finding a valid vector with an appropriate cost.

Claim 5.2 *There is a solution $E' \subseteq E(G)$ for (G, k) if and only if there is a valid vector X of $\mathcal{H}$ with $|E(G)| - |E(X)| \leq k$.*

Regarding the running time, the set $\mathcal{H}$ can be computed in $2^{\mathsf{nd}} \cdot n^{O(1)}$ time by taking all possible subset of nodes of H and checking if each subset induces a clique in H. Additionally, for every V_i the set $M(i)$ can be found in $|\mathcal{H}| \cdot n^{O(1)}$ time by traversing all cliques of $\mathcal{H}$. Moreover, the ILP has at most 2^{nd} number of variables, where the value of any variable is bounded by n. Thus by applying Theorem 5.1 the ILP can be solved in $2^{2^{O(\mathsf{nd})}} \cdot n^{O(1)}$ time. Then we can compute a solution for Cluster Deletion from a valid vector in the same running time, as described in Claim 5.1. Therefore the overall running time is bounded by the time needed to solve the ILP system as it contains 2^{nd} number of variables. □

References

1. Bansal, N., Blum, A., Chawla, S.: Correlation clustering. Mach. Learn. **56**, 89–113 (2004)
2. Ben-Dor, A., Shamir, R., Yakhini, Z.: Clustering gene expression patterns. J. Comput. Biol. **6**, 281–297 (1999)
3. Böcker, S., Briesemeister, S., Bui, Q.B.A., Truß, A.: Going weighted: Parameterized algorithms for cluster editing. Theor. Comput. Sci. **410**, 5467–5480 (2009)

4. Bonnet, É., Sikora, F.: The graph motif problem parameterized by the structure of the input graph. Discret. Appl. Math. **231**, 78–94 (2017)
5. Bonomo, F., Durán, G., Valencia-Pabon, M.: Complexity of the cluster deletion problem on subclasses of chordal graphs. Theor. Comp. Science **600**, 59–69 (2015)
6. Cai, L.: Fixed-parameter tractability of graph modification problems for hereditary properties. Inf. Process. Lett. **58**, 171–176 (1996)
7. Cao, Y., Chen, J.: Cluster editing: Kernelization based on edge cuts. Algorithmica **64**(1), 152–169 (2012)
8. Cao, Y., Ke, Y.: Improved Kernels for Edge Modification Problems. In: Proceedings of IPEC 2021. pp. 13:1–13:14 (2021)
9. Courcelle, B.: The monadic second-order logic of graphs i: Recognizable sets of finite graphs. Information and Computation **85**, 12–75 (1990)
10. Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer (2015)
11. Dahlhaus, E., Johnson, D.S., Papadimitriou, C.H., Seymour, P.D., Yannakakis, M.: The complexity of multiterminal cuts. SIAM J. on Computing **23**, 864–894 (1994)
12. Doucha, M., Kratochvíl, J.: Cluster vertex deletion: A parameterization between vertex cover and clique-width. In: Proceedings of MFCS 2012. vol. 7464, pp. 348–359 (2012)
13. Frank, A., Tardos, É.: An application of simultaneous diophantine approximation in combinatorial optimization. Comb. **7**, 49–65 (1987)
14. Ganian, R.: Twin-cover: Beyond vertex cover in parameterized algorithmics. In: Parameterized and Exact Computation. pp. 259–271 (2012)
15. Ganian, R.: Improving Vertex Cover as a Graph Parameter. Discrete Mathematics & Theoretical Computer Science **Vol. 17 no.2** (2015)
16. Gao, Y., Hare, D.R., Nastos, J.: The cluster deletion problem for cographs. Discrete Mathematics **313**, 2763–2771 (2013)
17. Gramm, J., Guo, J., Hüffner, F., Niedermeier, R.: Graph-modeled data clustering: Fixed-parameter algorithms for clique generation. In: Proceedings of CIAC 2003. vol. 2653, pp. 108–119 (2003)
18. Grüttemeier, N., Komusiewicz, C.: On the relation of strong triadic closure and cluster deletion. Algorithmica **82**, 853–880 (2020)
19. Jr., H.W.L.: Integer programming with a fixed number of variables. Math. Oper. Res. **8**, 538–548 (1983)
20. Komusiewicz, C., Uhlmann, J.: Cluster editing with locally bounded modifications. Discrete Applied Mathematics **160**, 2259–2270 (2012)
21. Komusiewicz, C., Uhlmann, J.: Alternative parameterizations for cluster editing. In: Proceedings of SOFSEM 2011. vol. 6543, pp. 344–355 (2011)
22. Konstantinidis, A.L., Papadopoulos, C.: Maximizing the strong triadic closure in split graphs and proper interval graphs. Discret. Appl. Math. **285**, 79–95 (2020)
23. Konstantinidis, A.L., Papadopoulos, C.: Cluster deletion on interval graphs and split related graphs. Algorithmica **83**(7), 2018–2046 (2021)
24. Lampis, M.: Algorithmic meta-theorems for restrictions of treewidth. Algorithmica **64**(1), 19–37 (2012)
25. Li, P., Puleo, G.J., Milenkovic, O.: Motif and hypergraph correlation clustering. IEEE Trans. Inf. Theory **66**, 3065–3078 (2020)
26. Shamir, R., Sharan, R., Tsur, D.: Cluster graph modification problems. Discrete Applied Mathematics **144**, 173–182 (2004)
27. Tsur, D.: Cluster deletion revisited. Inf. Process. Lett. **173**, 106171 (2022)