Cluster Editing Parameterized Above Modification-Disjoint P_3-Packings

Shaohua Li
Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Poland

Marcin Pilipczuk
Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Poland

Manuel Sorge
Faculty of Mathematics, Informatics and Mechanics, University of Warsaw, Poland
Institute of Logic and Computation, TU Wien, Austria

Abstract

Given a graph $G = (V, E)$ and an integer k, the Cluster Editing problem asks whether we can transform G into a union of vertex-disjoint cliques by at most k modifications (edge deletions or insertions). In this paper, we study the following variant of Cluster Editing. We are given a graph $G = (V, E)$, a packing $\mathcal{H}$ of modification-disjoint induced P_3s (no pair of P_3s in $\mathcal{H}$ share an edge or non-edge) and an integer ℓ. The task is to decide whether G can be transformed into a union of vertex-disjoint cliques by at most $\ell + |\mathcal{H}|$ modifications (edge deletions or insertions). We show that this problem is NP-hard even when $\ell = 0$ (in which case the problem asks to turn G into a disjoint union of cliques by performing exactly one edge deletion or insertion per element of $\mathcal{H}$) and when each vertex is in at most 23 P_3s of the packing. This answers negatively a question of van Bevern, Froese, and Komusiewicz (CSR 2016, ToCS 2018), repeated by C. Komusiewicz at Shonan meeting no. 144 in March 2019. We then initiate the study to find the largest integer c such that the problem remains tractable when restricting to packings such that each vertex is in at most c packed P_3s. Van Bevern et al. showed that the case $c = 1$ is fixed-parameter tractable with respect to ℓ and we show that the case $c = 2$ is solvable in $|V|^{2\ell+O(1)}$ time.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases Graph algorithms, fixed-parameter tractability, parameterized complexity

Digital Object Identifier 10.4230/LIPIcs.STACS.2021.49

Related Version *Full Version*: https://arxiv.org/abs/1910.08517 [37]

Funding This research is part a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement 714704.

1 Introduction

Correlation Clustering is a well-known problem motivated by research in computational biology [6] and machine learning [5]. In this problem we aim to partition data points into groups or clusters according to their pairwise similarity and this has been intensively studied in the literature, see [1, 3, 4, 5, 6, 15], for example.

In this paper, we study Correlation Clustering from a graph-based point of view, resulting in the following problem formulation. A graph H is called a *cluster graph* if H is a union of vertex-disjoint cliques; we also call these cliques *clusters*. Given a graph $G = (V, E)$, in the optimization version of Cluster Editing we ask for a minimum-size *cluster-editing set* S, that is, a set $S \subseteq \binom{V}{2}$ of vertex pairs such that $G \triangle S := (V, E \triangle S)$ is a cluster graph. Here $E \triangle S$ is the symmetric difference of E and S, that is, $E \triangle S = (E \setminus S) \cup (S \setminus E)$. We also sometimes refer to vertex pairs as *edits*. Cluster Editing is NP-hard [43]. Constant-ratio

approximation algorithms have been found for the optimization variant [1, 5, 15] but it is also APX-hard [15]. We focus here on exact algorithms and the decision version of Cluster Editing.

Given a natural number k and a graph $G = (V, E)$, the decision version of Cluster Editing asks whether there exists a cluster-editing set S such that $|S| \leq k$. Exact parameterized algorithms for Cluster Editing and some of its variants have been extensively studied [28, 7, 42, 19, 12, 31, 10, 11, 21, 30, 8, 35, 24, 40, 13]. Cluster Editing is but one of a large group of edge modification problems that have been studied, see Crespelle et al. [17] for a recent survey. Perhaps it is one of the most important such problems because of the practical motivation. Barring few exceptions [19, 35, 24, 44], Cluster Editing has mainly been studied with respect to the solution-size parameter k. It is not hard to observe that Cluster Editing is fixed-parameter tractable with respect to k and a series of papers [28, 27, 7, 11, 8] continually improved the base in the exponential part of the running time, culminating in the current fastest fixed-parameter algorithm with running time $O(1.62^k + n + m)$ [8], where n is the number of vertices of the input graph and m its number of edges. Similarly, a series of papers [28, 20, 22, 29, 42, 14, 16] gave more and more effective problem kernels[1] until a problem kernel with $2k$ vertices was achieved [14, 16].

As mentioned, the interest in Cluster Editing is not merely theoretical. Indeed, parameterized techniques are implemented in standard clustering tools [41, 45]. Although practitioners report that in particular the parameterized data-reduction techniques are effective [10, 9], the parameter k is not very small in several real-world data sets [7, 10, 44]. For instance, Böcker et al. [7, Table 2] considered 26 graphs derived from biological data with 91 to 100 vertices on which the average number of needed edits is 315, despite noting that the Cluster Editing model outperformed other clustering models.

A technique to deal with such large parameters is *parameterization above lower bounds*. Herein, the parameter is of the form $\ell = k - h$ where h is a lower bound on the solution size, usually computable in polynomial time, and ℓ is the *excess* of the solution size above the lower bound. Research into parameterizations above lower bounds has been active and fruitful for several parameterized problems, not only on the theory-side (see [39, 18, 26, 38, 36], for example) but also in practice, as algorithms based on that approach yielded quite efficient implementations for Vertex Cover [2] and among the most efficient ones for Feedback Vertex Set [32, 34]. For Cluster Editing we are aware of only one research work considering parameterizations above lower bounds: Van Bevern, Froese, and Komusiewicz [44] studied edge-modification problems parameterized above the lower bound from packings of forbidden induced subgraphs and showed that Cluster Editing parameterized by the excess above the size of a given packing of *vertex-disjoint* P_3s is fixed-parameter tractable. Observe that a graph is a cluster graph if and only if it does not contain any P_3, a path on three vertices, as an induced subgraph. Consequently, one needs to perform at least one edge deletion or insertion per element of the packing.

As the P_3s in the above packing are vertex-disjoint, the value by which the packing can decrease the parameter is limited. In the previous example with 315 edits, subtracting the resulting lower bound would reduce the parameter by at most 33. In their conclusion, van Bevern et al. [44] asked whether Cluster Editing is fixed-parameter tractable when parameterized above a stronger lower bound, the size of a modification-disjoint packing

[1] A problem kernel is a formalization of provably effective and efficient data reduction. It is a polynomial-time self-reduction that produces instances of size bounded by some function of the parameter.

of P_3s. Here, a packing $\mathcal{H}$ of induced P_3s in G is *modification-disjoint* if every two P_3s in $\mathcal{H}$ do not share edges or non-edges, that is, they share at most one vertex. The formal problem definition is as follows.

> Cluster Editing above modification-disjoint P_3 packing (CEaMP)
> **Input:** A graph $G = (V, E)$, a modification-disjoint packing $\mathcal{H}$ of induced P_3s of G, and a non-negative integer ℓ.
> **Question:** Is there a cluster editing set, i.e. a set of vertex pairs $S \subseteq \binom{V}{2}$ so that $G \triangle S$ is a union of disjoint cliques, with $|S| - |\mathcal{H}| \leq \ell$?

We also say that a set S as above is a *solution*.

Our results. At Shonan Meeting no. 144 [33] Christian Komusiewicz re-iterated the question of van Bevern et al. [44] and it was also asked in Vincent Froese's dissertation [25]. In this paper, we answer this question negatively by showing the following.

▶ **Theorem 1.** *Cluster Editing above modification-disjoint P_3 packing is NP-hard even for $\ell = 0$ and when each vertex in the input graph is incident with at most 23 P_3s of $\mathcal{H}$.*

In other words, given a graph G and a packing $\mathcal{H}$ of modification-disjoint P_3s in G, it is NP-hard to decide if one can delete or insert exactly one edge per element of $\mathcal{H}$ to obtain a cluster graph. Proving Theorem 1 was surprisingly nontrivial. A straightforward approach would be to amend the known reductions [35, 23] that show NP-hardness for constant maximum vertex degree by specifying a suitable packing of P_3s. However, an argument based on the linear-programming relaxation of packing modification-disjoint P_3s shows that the graphs produced by these reductions do not admit tight P_3 packing bounds. We did not find a way around this issue and thus developed a novel reduction based on new gadgets.

The verdict spelt by Theorem 1 is unfortunately quite damning. It indicates that even just reaching the lower bound given by a modification-disjoint P_3 packing already captures the algorithmic hardness of the problem. However, there may be a way out of this conundrum: Call a modification-disjoint P_3 packing $1/c$-*integral* if each vertex is in at most c packed P_3s (and say *integral* in place of *1-integral* and *half-integral* in place of $1/2$-*integral*). As the case $c = 1$ is just the case of vertex-disjoint packings, van Bevern et al. [44] showed that Cluster Editing parameterized by the excess over integral P_3 packings is fixed-parameter tractable. Thus it becomes an intriguing question to find the largest $c < 23$ such that CEaMP remains tractable with respect to the excess over $1/c$-integral packings. We provide progress towards answering this question here. The problem Cluster Editing above half-integral P_3 packing (CEaHMP) is defined in the same way as CEaMP except that the input packing $\mathcal{H}$ is half-integral. It turns out that the complexity of the problem indeed drops when making the packing half-integral:

▶ **Theorem 2.** *Cluster Editing above half-integral P_3 packing parameterized by the number ℓ of excess edits is in XP. It can be solved in $O(n^{2\ell+O(1)})$ time, where n is the number of vertices in the input graph.*

A straightforward idea to prove Theorem 2 would be to adapt the fixed-parameter algorithm for vertex-disjoint packings given by van Bevern et al. [44]. Their main idea is to show that if a packed P_3 P of the input graph G admits a solution that is optimal for P and that respects certain conditions on the neighborhood of $V(P)$ in G then this solution can be used in an optimal cluster-editing set for G. Afterwards, each packed P_3 P either needs an excess edit

in $V(P)$ or an edit incident with $V(P)$ in G. Since the P_3s in the packing are vertex-disjoint an edit incident with $V(P)$ will be in excess over the packing lower bound as well. It then follows that the overall number of edits is bounded by a function of the excess edits.

Unfortunately, the above idea fails for modification-disjoint packings for two reasons. First, the property that packed P_3s have an edit incident with them is not helpful anymore, because these edits may be part of other packed P_3s and hence not be in excess. Second, if we would like to preserve that these edits are excess, we need to check the special neighborhood properties of van Bevern et al. [44] for arbitrarily large connected components of packed P_3s efficiently. We did not see a way around these issues and instead designed an algorithm from scratch: A straightforward guessing of the excess edits reduces the problem to the case where we need to check for zero excess edits. This case is then solved by an extensive set of reduction rules that exploit the structure given by the half-integral packing. Essentially, we successively reduce the maximum size of clusters in the final cluster graph. This then allows us to reduce the problem to Cluster Deletion. Together with the properties of the packing, this problem allows a formulation as a 2SAT formula which we then solve in polynomial time.

Organization. After brief preliminaries in Section 2, we give an outline of the reduction used to show Theorem 1 in Section 3. Section 4 then contains an outline of the proof of Theorem 2. Due to space constraints, parts of the constructions and algorithms are deferred to a full version of the paper [37].

2 Preliminaries

In this paper, we denote an undirected graph by $G = (V, E)$, where $V = V(G)$ is the set of vertices, $E = E(G)$ is the set of edges, and $\binom{V}{2} \setminus E$ is the set of *non-edges*. An undirected edge between two vertices u and v will be denoted by uv where we put $uv = vu$. An undirected non-edge between two vertices x and y will be denoted by xy, where we put $xy = yx$, and we will explicitly mention that xy is a non-edge in case of confusion with the notation of an edge. If uv is an edge in the graph, we say u and v are *adjacent*. We denote a bipartite graph by $B = (U, W, E)$, where U, W are the two parts of the vertex set of B and E is the set of edges of B. We say that a bipartite graph is *complete* if for every pair of vertices $u \in U$ and $w \in W$, $uw \in E$. For a non-empty subset of vertices $X \subseteq V$, we denote the subgraph induced by X by $G[X]$. A *clique* Q in a graph G is a subgraph of G in which any two distinct vertices are adjacent. A *cluster graph* is a graph in which every connected component is a clique. A connected component in a cluster graph is called a *cluster*.

Let G' be a cluster graph and let S be a cluster editing set S such that $G \triangle S = G'$. We say that two cliques Q_1 and Q_2 of G are *merged* (in G') if they belong to the same cluster in G'. We say that Q_1 and Q_2 are *separated* (in G') if they belong to two different clusters in G'. When mentioning the edges or non-edges between the vertices of the clique Q_1 and the vertices of the clique Q_2, we refer to the edges or non-edges between the clique Q_1 and the clique Q_2 for short. Let $\ell, r \in \mathbb{N}$. We denote a path with ℓ vertices by P_ℓ and a cycle with r vertices by C_r.

Let x, y, z be vertices in a graph G. We say that xyz is an induced P_3 of G if $xy, yz \in E(G)$ and $xz \notin E(G)$. Vertex y is called the *center* of xyz. We say that vertices x, y, z *belong to* xyz or x, y, z are *incident with* xyz. We also say that xyz is *incident with* the vertices x, y and z. In this paper, all P_3s we mention are induced P_3s; we sometimes skip the qualifier "induced" for convenience.

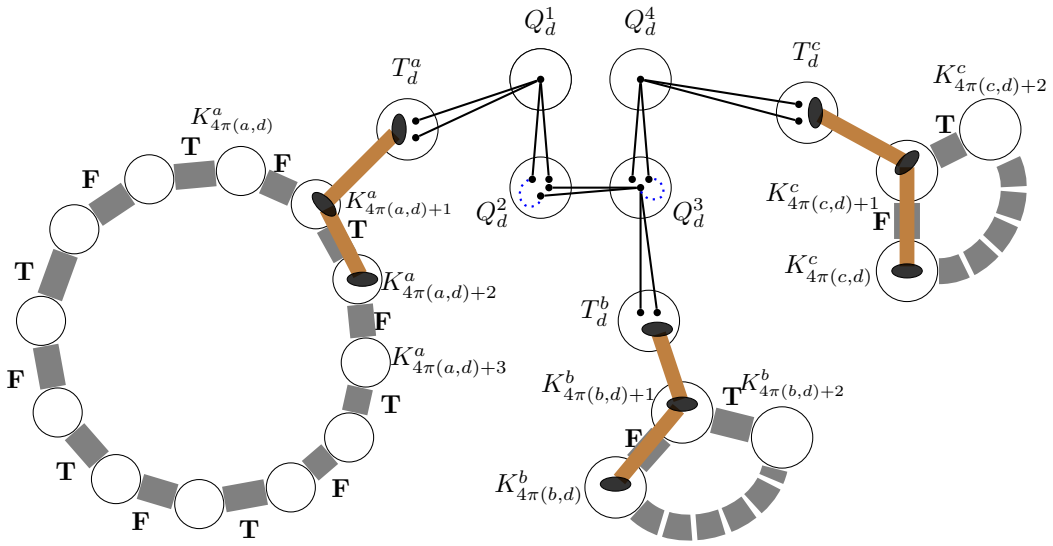

Figure 1 Basic structure of graph constructed from a 3-CNF formula. Depicted is a clause gadget for a clause $x_a \vee \neg x_b \vee \neg x_c$ and the variable gadgets for x_a, x_b, and x_c.

Given an instance $(G, \mathcal{H}, \ell)$ of CEAMP, if xyz is a P_3 in G and $xyz \in \mathcal{H}$, we say that xyz is *packed*, and we say that the edges xy, yz are *covered* by xyz and the non-edge xz is covered by xyz. If an edge xy is covered by some P_3 of $\mathcal{H}$, we say that xy is a *packed* edge. Otherwise we say that xy is a *non-packed* edge. If a non-edge uv is covered by some P_3 of $\mathcal{H}$, we say that uv is a *packed* non-edge. Otherwise we say that uv is a *non-packed* non-edge. If xyz is a P_3 in G and Q_1, Q_2, and Q_3 are pair-wise non-intersecting vertex sets of G, we say that xyz *connects* Q_1 *and* Q_3 *via* Q_2 if the center y of xyz belongs to Q_2 and x, z belong to Q_1 and Q_3, respectively.

We sometimes need finite fields of prime order. Let p be some prime. By $\mathbb{F}_p$ we denote the finite field with the p elements $0, \ldots, p-1$ with addition and multiplication modulo p.

3 NP-hardness for tight modification-disjoint packings

Overview. In this section, we outline the proof of Theorem 1 that shows a reduction from the NP-hard problem of deciding satisfiability of 3-CNF formulas. Given a 3-CNF formula Φ with variables $x_0, \ldots, x_{n-1}$ and clauses $\Gamma_0, \ldots, \Gamma_{m-1}$ we construct a graph $G = (V, E)$ with a modification-disjoint packing $\mathcal{H}$ of induced P_3s such that Φ has a satisfying assignment if and only if G has a cluster editing set S which consists of exactly one edit in each P_3 in $\mathcal{H}$.

Let us start with Figure 1 which depicts the basic structure of the graph G. The fundamental building blocks of G and $\mathcal{H}$ are what we call proto-clusters, indicated by white circles. A *proto-cluster* is an induced subgraph H of G whose vertex set is maximal with respect to the property that H contains a spanning tree that consists entirely of non-packed edges. Note that the set of proto-clusters partitions the vertex set of G. As we cannot edit non-packed edges, the clusters in each solution that we may obtain induce a partition that is coarser than the partition given by the proto-clusters.

Our first concern is to interconnect the proto-clusters in such a way that a grouping into solution clusters implies a satisfying assignment of Φ – the construction is *sound*. To this end, a straightforward idea of modeling the truth-value of a variable comes to mind: Use

an even-length cycle of proto-clusters and add P_3s to the packing $\mathcal{H}$ such that either the odd pairs or the even pairs of proto-clusters on the cycle need to be merged into clusters. The variable gadgets are represented by the three gray cycles in Figure 1. A clause gadget is slightly less obvious, because we need a three-way choice and straightforward constructions yield only two-way choices. A solution is shown in Figure 1: There are four proto-clusters, Q_d^1 through Q_d^4 such that there is a non-packed nonedge between Q_d^1 and Q_d^4 and a path in G from Q_d^1 over Q_d^2 and Q_d^3 to Q_d^4. Because of the non-packed nonedge, the proto-clusters Q_d^1 and Q_d^4 are in different solution clusters. Hence, we need to separate a pair of Q_d^i and Q_d^{i+1} for some $i \in [3]$. This models the choice of the variable that shall satisfy the clause. This choice is then transferred to the variable gadgets by suitable packed P_3s and further proto-clusters. A nontrivial issue in this transfer of choices is how to connect variable gadgets to the rest of the construction. On the one hand, we need to pack P_3s that are partly in the variable gadgets and partly outside so as to transfer the choice and on the other hand, we need a packing of P_3s inside the variable gadgets in order to allow both the merging of odd pairs and even pairs of proto-clusters in the gadget.

The most involved part of the construction is indeed how to ensure the *completeness*, that is, the property that a satisfying assignment for Φ gives a cluster-editing set with zero excess edits for G. This issue makes the construction that we obtain somewhat special: We need to pack P_3s into the above "skeleton" construction so as to allow for the merging and cutting of pairs of proto-cluster according to the satisfying assignment. We accomplish this by a careful implementation of the above gadgets such that the edges and non-edges that are covered by the packed P_3s of the skeleton construction have a special structure. We then use an algebraic construction that allows us to prove that the needed covering of the remaining edges and non-edges by modification-disjoint P_3s exists.

We now proceed to describing the variable and clause gadgets more formally and then show how we have resolved the above two issues.

Variable gadgets. As mentioned, a variable will be represented by a cycle of proto-clusters such that any solution needs to merge either each odd or each even pair of consecutive proto-clusters. These two options represent the truth value assigned to the variable. In order to enable both associated solutions with zero edits above the packing lower bound, we build an associated packing of P_3s such that all vertex pairs between consecutive proto-clusters are covered by a P_3 in the packing. Since we later on need to connect the variable gadgets to the clause gadgets, each proto-cluster will contain five vertices, giving us enough attachment points for later.

Let m_i denote the number of clauses that contain the variable x_i, $i = 0, 1, \ldots, n-1$. For each variable x_i, $i = 0, 1, \ldots, n-1$, we create $4m_i$ vertex-disjoint cliques with 5 vertices each, namely $K_0^i, \ldots, K_{4m_i-1}^i$. In each K_j^i, $j = 0, 1, \ldots, 4m_i - 1$, the vertices are $v_{j,0}^i, \ldots, v_{j,4}^i$. For each $j = 0, 2, \ldots, 4m_i - 2$, we create P_3s connecting K_j^i, K_{j+1}^i, and K_{j+2}^i as follows (here we identify K_0^i as $K_{4m_i}^i$).

We add pairwise modification-disjoint P_3s to cover all edges between the cliques K_j^i we have just introduced. Recall that $\mathbb{F}_5$ is the finite field of the integers modulo 5. We take three consecutive cliques and add P_3s with one vertex in each of the three cliques. To do this without overlapping two P_3s, we think about the cliques' vertices as elements of $\mathbb{F}_5$ and add a P_3 for each possible arithmetic progression. That is, in each added P_3 the difference of the first two elements of the P_3 is equal to the difference of the second two elements. In this way, each vertex pair is contained in a single P_3 since the third element is uniquely defined.

Formally, for every triple of elements $p, q, r \in \mathbb{F}_5$ satisfying the equality $q - p = r - q$ over $\mathbb{F}_5$, we add to the graph the edges $v^i_{j,p}v^i_{j+1,q}$ and $v^i_{j+1,q}v^i_{j+2,r}$ and to the packing $\mathcal{H}$ the P_3 given by $v^i_{j,p}v^i_{j+1,q}v^i_{j+2,r}$. Note that in this manner the clique K^i_{j+1} becomes fully adjacent to K^i_j and to K^i_{j+2} while K^i_{j+1} stays anti-adjacent to all other cliques $K^i_{j'}$.

Observe that the P_3s given by $v^i_{j,p}v^i_{j+1,q}v^i_{j+2,r}$ for $j = 0, 2, \ldots, 4m_i - 2$ such that $q - p = r - q$ are pairwise modification-disjoint: For each $j = 0, 2, \ldots, 4m_i - 2$, an arbitrary edge just introduced between K^i_j and K^i_{j+1} has the form $\{v^i_{j,p}, v^i_{j+1,q}\}$ for some $p, q \in \mathbb{F}_5$. It belongs to the unique P_3 given by $v^i_{j,p}v^i_{j+1,q}v^i_{j+2,r}$, where $r = 2q - p$. Similarly, an arbitrary edge $\{v^i_{j+1,q}, v^i_{j+2,r}\}$ for $q, r \in \mathbb{F}_5$ belongs to the unique P_3 given by $v^i_{j,2q-r}v^i_{j+1,q}v^i_{j+2,r}$ and an arbitrary non-edge $\{v^i_{j,p}, v^i_{j+2,r}\}$ for $p, r \in \mathbb{F}_5$ belongs to the unique P_3 given by $v^i_{j,p}v^i_{j+1,(p+r)\cdot 2^{-1}}v^i_{j+2,r}$, where 2^{-1} is the multiplicative inverse of 2 over $\mathbb{F}_5$, that is, $2^{-1} = 3$.

After this construction, we set the P_3 packing of the variable gadgets to

$$\mathcal{H}_{\mathsf{var}} = \{v^i_{j,p}v^i_{j+1,q}v^i_{j+2,r} \mid i = 0, \ldots, n-1;\ j = 0, 2, \ldots, 4m_i - 2;\ p, q, r \in \mathbb{F}_5;\ \text{and } q - p = r - q\}.$$

This finishes the first stage of the construction. The truth values of the variable are represented as follows. For every variable x_i, $i = 0, \ldots, n-1$, if K^i_j and K^i_{j+1} are merged for $j = 0, \ldots, 4m_i - 2$, then this represents assigning false to the variable x_i. If K^i_{j+1} and K^i_{j+2} are merged for $j = 0, \ldots, 4m_i - 2$, then this represents variable x_i being true. We will make minor modifications to the variable gadgets and $\mathcal{H}_{\mathsf{var}}$ below so as to transmit the choice of truth value to the clause gadgets.

Skeleton of the clause gadget. In order to introduce the construction of the clause gadget, we first give a description of the skeleton of the clause gadget. The skeleton, depicted in Figure 1, is a subgraph of the final construction that allows us to prove the soundness. The construction is finalized later.

For each variable x_i, $i = 0, 1, \ldots, n-1$, of Φ we fix an arbitrary ordering of the clauses that contain x_i. If a clause Γ_j contains x_i, let $\pi(i, j) \in \{0, \ldots, m_i - 1\}$ denote the position of the clause Γ_j in this ordering. Let initially $\mathcal{H}_{\mathsf{tra}} = \emptyset$. For each clause Γ_d ($d = 0, \ldots, m-1$) proceed as follows. We first introduce four cliques Q^1_d, Q^2_d, Q^3_d and Q^4_d. Let Γ_d contain the variables x_a, x_b and x_c. We introduce the cliques T^a_d, T^b_d and T^c_d, called *transferring cliques*. All of the cliques introduced are pairwise vertex disjoint and can be of different sizes. The concrete size will be determined later. Next, we introduce the following P_3s into G and $\mathcal{H}_{\mathsf{tra}}$ (see the center of Figure 1):

- P^1_d and P^2_d that both connect T^a_d and Q^2_d via Q^1_d and that share a vertex in Q^1_d.
- P^3_d and P^4_d that both connect T^b_d and Q^2_d via Q^3_d and that share a vertex in Q^3_d.
- P^5_d and P^6_d that both connect T^c_d and Q^3_d via Q^4_d and that share a vertex in Q^4_d.

All the P_3s above are pairwise vertex-disjoint except for the shared vertices explicitly mentioned in the definition. We call the P_3s of $\mathcal{H}_{\mathsf{tra}}$ *transferring* P_3*s*.

Connection to the variable gadgets. Next we connect the transferring cliques T^a_d, T^b_d, and T^c_d to the variable gadgets of x_a, x_b, and x_c, respectively. To avoid additional notation, we only explain the procedure for T^a_d and x_a, the other pairs are connected analogously. We connect T^a_d to the variable gadget of x_a by a set of four modification-disjoint P_3s as shown in Figure 2 and explained formally below. The centers of these P_3s are in $K^a_{4\pi(a,d)+1}$. For each of these four P_3s, exactly one endpoint is an arbitrary distinct vertex in T^a_d which is different from the endpoints of the P_3s connecting T^a_d to Q^1_d; we denote these endpoints as w_1, w_2, w_3, w_4. The other endpoint is in $K^a_{4\pi(a,d)+2}$ if x_a appears positively in Γ_d and the other endpoint is in $K^a_{4\pi(a,d)}$ otherwise. The precise centers and endpoints in the cliques

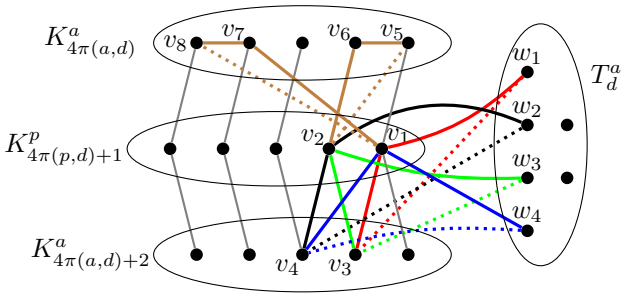

Figure 2 Connection of a clause gadget with a variable gadget for a variable x_a which appears positively in the clause. White ellipses represent cliques. The vertices in the cliques in the variable gadget are ordered from right to left according to the elements of $\mathbb{F}_5$ which they represent. For example, the rightmost vertex in $K^a_{4\pi(a,d)}$ is $v^a_{4\pi(a,d),0}$ (corresponding to $0 \in \mathbb{F}_5$) and the leftmost is $v^a_{4\pi(a,d),4}$ (corresponding to $4 \in \mathbb{F}_5$). The gray lines adjacent to cliques in the variable gadget represent some of the P_3s that were introduced into the variable gadgets in the beginning. In colors red, black, green, and blue we show the P_3s that connect the transferring clique T^a_d with the variable gadget of variable x_a. Herein, dotted lines are non-edges and solid lines are edges. Note that these connecting P_3s supplant some of the edges of previously present P_3s in the variable gadget – the previously present P_3s are then removed. For example the green P_3 replaces the edge v_2v_3 of the P_3 given by $v_6v_2v_3$ that was previously present. To maintain that each vertex pair between consecutive cliques in the variable gadget is covered by some P_3 in the packing, we add the brown P_3s.

$K^a_{4\pi(a,d)+2}$ or $K^a_{4\pi(a,d)}$ are specified below. Since these newly introduced P_3s use edges that belong to some P_3s in $\mathcal{H}_{\mathsf{var}}$ that were introduced while constructing the variable gadgets, we will remove such P_3s in the variable gadget from $\mathcal{H}_{\mathsf{var}}$, remove their corresponding edges from the graph, and add some new P_3s to $\mathcal{H}_{\mathsf{var}}$ as described below. As a result, the clique $K^a_{4\pi(a,d)+1}$ may no longer be fully adjacent to $K^a_{4\pi(a,d)}$ or $K^a_{4\pi(a,d)+2}$. We will however maintain the invariant that each vertex pair between $K^a_{4\pi(a,d)+1}$ and $K^a_{4\pi(a,d)}$ or $K^a_{4\pi(a,d)+2}$ is covered by a P_3 in the packing and that all the P_3s of $\mathcal{H}_{\mathsf{var}}$ are pairwise modification-disjoint.

Formally, if x_a appears positively in Γ_d, we denote:

$$v_1 = v^a_{4\pi(a,d)+1,0} \quad v_2 = v^a_{4\pi(a,d)+1,1} \quad v_3 = v^a_{4\pi(a,d)+2,1} \quad v_4 = v^a_{4\pi(a,d)+2,2}$$
$$v_5 = v^a_{4\pi(a,d),0} \quad v_6 = v^a_{4\pi(a,d),1} \quad v_7 = v^a_{4\pi(a,d),3} \quad v_8 = v^a_{4\pi(a,d),4}.$$

If x_a appears negatively in Γ_d, we swap the roles of $K^a_{4\pi(a,d)}$ and $K^a_{4\pi(a,d)+2}$, that is:

$$v_1 = v^a_{4\pi(a,d)+1,0} \quad v_2 = v^a_{4\pi(a,d)+1,1} \quad v_3 = v^a_{4\pi(a,d),1} \quad v_4 = v^a_{4\pi(a,d),2}$$
$$v_5 = v^a_{4\pi(a,d)+2,0} \quad v_6 = v^a_{4\pi(a,d)+2,1} \quad v_7 = v^a_{4\pi(a,d)+2,3} \quad v_8 = v^a_{4\pi(a,d)+2,4}.$$

As shown in Figure 2, we remove the P_3s given by $v_8v_1v_3$, $v_7v_1v_4$, $v_6v_2v_3$, and $v_5v_2v_4$ from $\mathcal{H}_{\mathsf{var}}$ and we remove their corresponding edges from the graph. Then we add the P_3s given by $v_5v_6v_2$ and $v_1v_7v_8$ to the graph and to $\mathcal{H}_{\mathsf{var}}$. Finally, we connect T^a_d via $K^a_{4\pi(a,d)+1}$ by adding the P_3s given by $w_1v_1v_3$, $w_2v_2v_4$, $w_3v_2v_3$, and $w_4v_1v_4$ to the graph and to $\mathcal{H}_{\mathsf{tra}}$. Note that, indeed, each vertex pair between $K^a_{4\pi(a,d)+1}$ and $K^a_{4\pi(a,d)}$ and between $K^a_{4\pi(a,d)+1}$ and $K^a_{4\pi(a,d)+2}$ remains covered by a P_3 in the packing after replacing all P_3s. This finishes the construction of the skeleton of the clause gadgets.

Intuitively, the skeleton ensures the soundness as follows. Recall from above that we need to delete at least one of three sets of edges in the solution, namely the edges between Q^1_d and Q^2_d, the edges between Q^2_d and Q^3_d, or the edges between Q^3_d and Q^4_d. Assume that the

edges between Q^1_d and Q^2_d are deleted and the variable x_a appears positively in the clause Γ_d as in Figure 1. Because of the constraints imposed by the P_3s P^1_d and P^2_d, cliques T^a_d and Q^1_d have to be merged in the final cluster graph. Since $K^a_{4\pi(a,d)+1}$ cannot be merged with Q^1_d (there are no edges between Q^1_d and $K^a_{4\pi(a,d)+1}$, and no P_3s connecting Q^1_d and $K^a_{4\pi(a,d)+1}$), we have to separate T^a_d from $K^a_{4\pi(a,d)+1}$. Then, the P_3s connecting T^a_d with $K^a_{4\pi(a,d)+2}$ force $K^a_{4\pi(a,d)+1}$ and $K^a_{4\pi(a,d)+2}$ to merge. This means x_a is true and it satisfies the clause Γ_d.

Merging model and P_3 padding. Above we have defined all proto-clusters of the final constructed graph. What remains is to ensure that the proto-clusters indeed can be merged as required to construct a solution from a satisfying assignment to Φ in the completeness proof. Intuitively, in this construction we have pairs of proto-clusters A and B which we would like to be able to either merge or separate without incurring excess edits. To achieve this, we add P_3s that have both an edge and a nonedge between A and B. If we are able to cover all vertex pairs between A and B with such P_3s, then merging or separating A and B will indeed not incur excess edits. The pairs of proto clusters that we want to be able to merge are captured in the *merging model*, a graph H that contains as vertices the cliques in the gadgets that we have introduced and the following edges:

- $\{\{K^i_j, K^i_{j+1}\} \mid i = 0, 1, \ldots, n-1 \text{ and } j = 0, 1, \ldots, 4m_i - 1\}$; these pairs are needed to be able to set the variable gadgets according to their truth values.
- $\{\{T^i_d, K^i_{4\pi(i,d)}\}, \{T^i_d, K^i_{4\pi(i,d)+1}\}, \{T^i_d, K^i_{4\pi(i,d)+2}\} \mid \text{variable } x_i \text{ occurs in clause } \Gamma_d\}$; these edges are needed to merge a transferring clique to its corresponding variable gadget if the variable does not satisfy the clause associated with the transferring clique.
- $\{\{Q^1_d, Q^2_d\}, \{Q^1_d, Q^3_d\}, \{Q^2_d, Q^3_d\}, \{Q^2_d, Q^4_d\}, \{Q^3_d, Q^4_d\} \mid d = 0, 1, \ldots, m-1\}$; in order to be able to merge proto-clusters in a clause gadget if they do not correspond to a variable that was chosen to satisfy the clause.
- $\{\{T^i_d, Q^k_d\} \mid \text{if variable } x_i \text{ occurs in } \Gamma_d \text{ and } T^i_d \text{ is adjacent in } G \text{ to } Q^k_d \text{ with } k \in \{1, 4\}\}$; in order to be able to merge a transferring clique to a proto-cluster in a clause gadget if the corresponding variable was chosen to satisfy the clause.
- $\{\{T^i_d, Q^3_d\}, \{T^i_d, Q^4_d\} \mid \text{if variable } x_i \text{ occurs in } \Gamma_d \text{ and } T^i_d \text{ is adjacent in } G \text{ to } Q^3_d\}$; ditto (the construction is asymmetric).

The aim is now to define a vertex partition of the merging model into levels and to pad P_3s between levels. The levels are as follows: L_0 contains all cliques in variable gadgets; L_1 contains Q^1_d and Q^4_d for each $d = 0, \ldots, m-1$; L_2 contains Q^3_d for each $d = 0, \ldots, m-1$; L_3 contains Q^2_d for each $d = 0, \ldots, m-1$; and L_4 contains all transferring cliques. Observe that apart from edges in L_0, all edges of H are between different levels. Moreover, orienting the edges in H from higher to lower level gives an acyclic orientation when ignoring the edges in level L_0 and each vertex in H on some level L_i is adjacent to only a constant number of vertices on a lower level L_j, $j < i$. Hence, we now go through the cliques in $V(H)$ in increasing order of levels and, for each clique Q, we pad P_3s between Q and its constant number of neighbors on lower levels. The padding is done so as to cover all vertex pairs between Q and the lower neighbors that are not covered by the skeleton yet. To show that such a covering exists, we need to analyze the structure of the vertex pairs that are already covered. We can show that these pairs form either P_3s (e.g. between T^a_d and Q^1_d in Figure 1) or cycles of length eight (e.g. between T^a_d and the two cliques $K^p_{4\pi(p,d)+1}$ and $K^p_{4\pi(p,d)+2}$ in Figure 2). Using this property, we can show that the desired padding of P_3s exists by proving the following result.[2] Note that the statement is about triangle packings; the triangles correspond to the vertex pairs covered by P_3s.

[2] To obtain the desired bound on the number of P_3s containing a fixed vertex we need a slightly more general result. See the details in the full version.

▶ **Lemma 3.** *Let p be a prime number with $p \geq 2$. Let $B = (V, W, E)$ be a complete bipartite graph such that $|V| = p$ and $|W| = 2p$. Let $F \subseteq E$ be a set of edges such that each connected component of $(V \cup W, F)$ is a either a P_3 with a center in V or a C_8. Then there exists an edge-disjoint triangle packing τ in $(V \cup W, E \setminus F \cup \binom{W}{2})$ which covers $E \setminus F$ such that every triangle in τ contains exactly one vertex of V, the graph $(W, \binom{W}{2} \setminus E(\bigcup \tau))$ is connected, and each vertex is in at most p triangles of τ.*

We apply Lemma 3 as follows: W is the clique for which we want to pad P_3s and V is the union of the cliques that are neighbors of W in H on lower layers. The set F contains the vertex pairs already covered by the skeleton. The packing τ corresponds to our desired P_3 packing. Finally, the connectedness property on $(W, \binom{W}{2} \setminus E(\bigcup \tau))$ ensures that W remains a proto-cluster. Using the padding we can ensure the soundness, concluding the proof of Theorem 1.

The proof of Lemma 3 works roughly as follows. We partition W into two parts W_1, W_2, each of size p. The triangles in the packing contain one vertex of each of W_1, W_2, and V; they are defined by interpreting W_1, W_2, and V each as the field F_p and taking three vertices $i \in W_1, j \in W_2, k \in V$ such that $j - i = k - j$. This defines a covering of all vertex pairs between V and W. The vertex pairs in F are avoided by covering them with specific triangles, which are then removed from the final packing.

4 XP-algorithm for half-integral packings

In this section, we study CEAMP in the special setting where every vertex is incident with at most two P_3s of the packing $\mathcal{H}$. We define this problem as Cluster Editing above half-integral P_3 packing (CEaHMP). We prove the following.

▶ **Theorem 2** (Restated). *Cluster Editing above half-integral P_3 packing parameterized by the number ℓ of excess edits is in XP. It can be solved in $O(n^{2\ell+O(1)})$ time, where n is the number of vertices in the input graph.*

The main tool in proving Theorem 2 is a polynomial-time algorithm for the case where $\ell = 0$:

▶ **Theorem 4.** *Cluster Editing above half-integral P_3 packing can be solved in polynomial time when $\ell = 0$, that is, when no excess edits are allowed.*

The proof of Theorem 4 will be given in the final part of this section. Using this we can prove Theorem 2 as follows: Essentially, the XP algorithm for Cluster Editing above half-integral P_3 packing guesses (by trying all possibilities) the number, ℓ_a, of excess edits that are not contained in any P_3 in $\mathcal{H}$ and guesses the concrete edits to be made. Then it guesses the P_3s in $\mathcal{H}$ that harbor the remaining excess edits and it guesses how these P_3s are resolved. Then it checks whether the remaining instance has a cluster-editing set without excess edits over the remaining P_3 packing $\mathcal{H}'$ using the algorithm from Theorem 4. The full proof for Theorem 2 is contained in the full version [37].

We outline the polynomial-time algorithm for CEaHMP when $\ell = 0$. It is based on a series of reduction rules that perform successive modifications to the input graph and P_3 packing to get an equivalent new instance. Whenever we state a reduction rule in the following, we assume that all the reduction rules before it have been applied exhaustively. We will omit the correctness proofs for most reduction rules and lemmas here – they are contained in the full version [37]. We first aim to decrease the maximum size of clusters in a solution cluster graph, then reduce to Cluster Deletion and then to 2SAT.

We again use the notion of proto-clusters as in the previous sections. We say a proto-cluster C is *isolated* from a proto-cluster D if there are no edges between C and D. We classify the P_3s of $\mathcal{H}$ into four types. For an induced P_3 $xyz \in \mathcal{H}$, if x, y belong to one proto-cluster and z belongs to another proto-cluster, or symmetrically y, z belong to one proto-cluster and x belongs to another proto-cluster, then xyz is a *type-α* P_3; if x, z belong to one proto-cluster and y belongs to another proto-cluster, then xyz is a *type-β* P_3; if x, y, z belong to three distinct proto-clusters, then xyz is a *type-γ* P_3; if x, y, z belong to one proto-cluster then xyz is a *type-δ* P_3.

The first five reduction rules are simple and their correctness follows almost immediately:

▶ **Reduction Rule 1.** *For any proto-cluster C, if there are two vertices $u, v \in V(C)$ such that uv is a non-packed non-edge, i.e., uv is not covered by any P_3 of $\mathcal{H}$, then return NO.*

▶ **Reduction Rule 2.** *If there is a type-β or type-δ P_3 $xyz \in \mathcal{H}$, insert the edge xz into G and remove xyz from $\mathcal{H}$.*

▶ **Reduction Rule 3.** *For any two proto-clusters A and B, if there is a non-packed non-edge uv such that $u \in V(A)$ and $v \in V(B)$, and there is a packed edge xy such that $x \in V(A)$ and $y \in V(B)$, then delete xy and remove the corresponding packed P_3 from $\mathcal{H}$.*

▶ **Reduction Rule 4.** *If there is a connected component $\mathcal{C}$ in the graph of size at most 6, then do brute force on $\mathcal{C}$ to check if there is a cluster-editing set F for $\mathcal{C}$ such that $|F|$ is equal to the number of packed P_3s incident with a vertex of C. If there is such a cluster-editing set F, then perform the operations of F to $\mathcal{C}$ and remove the corresponding packed P_3s from $\mathcal{H}$. Otherwise, if there is no such cluster-editing set F, return NO.*

▶ **Reduction Rule 5.** *If there is a proto-cluster C which is an isolated clique, then remove C from the graph.*

Already, the above simple rules effectively remove proto-clusters of size at least four:

▶ **Lemma 5.** *After applying Reduction Rules 1 - 5 exhaustively, if the algorithm did not return NO, then there is no proto-cluster of size at least 4.*

Proof. It is not hard to check that there are no isolated proto-clusters of size at least 4 after the previous reduction rules. Assume that there is a proto-cluster A of size at least 5 and a vertex $v \in V(G) \setminus V(A)$ such that v is adjacent to a vertex of A. There are at least five vertex pairs between v and $V(A)$ which are covered by packed P_3s because Reduction Rule 3 is not applicable. But v is incident with at most four packed edges or packed non-edges because of half-integrality, a contradiction. Next, assume that there is a proto-cluster B of size exactly 4 and a vertex $u \in V(G) \setminus V(B)$ such that u is adjacent to a vertex of B. There are four vertex pairs between u and $V(B)$ and, moreover, these are covered by two type-α P_3 since there is no type-β P_3 after Reduction Rule 2. We claim that $V(B) \cup \{u\}$ induces a connected component $\mathcal{C}$ in the graph. Suppose for contradiction that there is another vertex x adjacent to one vertex of $V(B)$. Then either Reduction Rule 3 can be applied if the vertex pairs between x and $V(B)$ are not all packed or otherwise B is not a proto-cluster: It would have to contain four packed edges and hence could not contain a spanning tree of non-packed edges. Thus Reduction Rule 4 applies to $\mathcal{C}$, again a contradiction. ◀

Next, only proto clusters of size at most three remain, but those of size exactly three have a very restricted structure. The following rule takes care of them.

▶ **Reduction Rule 6.** *After applying Reduction Rules 1 - 5 exhaustively, if there is a proto-cluster C of size 3, a proto-cluster B of size 1 and a proto-cluster A of size 1 such that C is not isolated from B, and a type-γ P_3 connects A and C via B, then delete the packed edge between A and B, insert an edge to the packed non-edge between C and B, and remove the corresponding P_3s from $\mathcal{H}$.*

▶ **Lemma 6.** *After applying Reduction Rules 1 - 6 exhaustively, there are no isolated cliques in the instance and every proto-cluster of the instance is of size at most 2. Moreover, every packed P_3 is a type-γ P_3.*

After applying Reduction Rules 1 - 6 exhaustively, suppose that the resulting instance $(G, \mathcal{H}, \ell = 0)$ of CEAHMP has a solution S. We now focus on the sizes of the clusters in $G \triangle S$. We can see that the maximum size of a cluster in $G \triangle S$ is six by some simple observation. The next lemma shows that clusters of size exactly six can be removed by Reduction Rule 4.

▶ **Lemma 7.** *Let $(G, \mathcal{H}, \ell = 0)$ be an instance of CEAHMP such that the size of every proto-cluster in G is at most 2 and let S be a solution to $(G, \mathcal{H}, \ell = 0)$. Suppose that A is a clique of size 6 in $G \triangle S$. Then the vertices of $V(A)$ belong to three proto-clusters C_1, C_2, and C_3 of size two in G. In addition, every vertex pair between C_1 and C_2, between C_1 and C_3, between C_2 and C_3 is covered by some P_3 of $\mathcal{H}$. Furthermore, $V(C_1) \cup V(C_2) \cup V(C_3)$ forms a connected component $\mathcal{C}$ in G.*

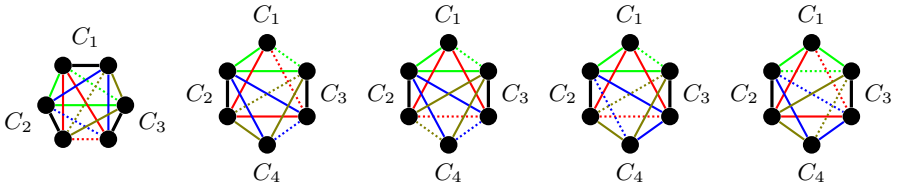

Figure 3 The first picture is an example of forming a clique of size 6 in $G \triangle S$ as in Lemma 7, the other pictures are potential examples of forming a clique of size 5 in $G \triangle S$ as in Lemma 8.

Clusters of size four and five also have restricted structures as shown by following lemmas.

▶ **Lemma 8.** *After applying Reduction Rules 1 – 3 exhaustively, let $(G, \mathcal{H}, \ell = 0)$ be an instance of CEAHMP such that the size of every proto-cluster in G is at most 2 and S is a solution to $(G, \mathcal{H}, \ell = 0)$. Suppose that A is a clique of size 5 in $G \triangle S$. Then the vertices of $V(A)$ belong to three proto-clusters C_1, C_2 and C_3 (or C_2, C_3 and C_4) in G such that $|C_1| = |C_4| = 1$ and $|C_2| = |C_3| = 2$. Every vertex pair between C_i and C_j $(i, j \in \{1, 2, 3, 4\}, i \neq j)$ is covered by a packed P_3 except that the vertex pair between C_1 and C_4 is a non-packed non-edge. In addition, $V(C_1) \cup V(C_2) \cup V(C_3) \cup V(C_4)$ forms a connected component $\mathcal{C}$ in G.*

▶ **Lemma 9.** *After applying Reduction Rules 1 – 6 exhaustively, let $(G, \mathcal{H}, \ell = 0)$ be an instance of CEAHMP such that the size of every proto-cluster in G is at most 2 and S is a solution to $(G, \mathcal{H}, \ell = 0)$. Suppose that A is a clique of size 4 in $G \triangle S$ and $V(A) = \{x, y, z_1, z_2\}$. Then three vertices of $V(A)$, say x, y, z_2 belong to one packed P_3 in G, and one vertex of x, y, z_2, say z_2, with z_1 forms a proto-cluster C_1 of size two in G while x and y form a proto-cluster C_2 of size one and a proto-cluster C_3 of size one in G respectively. Moreover, there are two vertices u and v such that x, u, z_1 belong to a packed P_3 in G, y, v, z_1 belong to another packed P_3 in G. u and v form a proto-cluster C_4 of size one and a proto-cluster C_5 of size one in G respectively.*

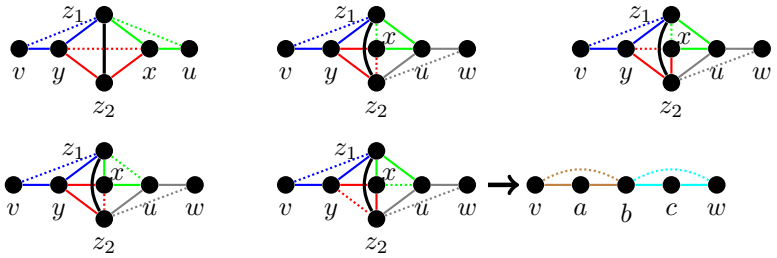

Figure 4 Subgraphs that may form clusters of size 4 and how to reduce them (bottom right).

Based on the previous lemmas, we design some reduction rules that handle all potential cases in which there are cliques of size at least 4 in $G \triangle S$. Due to space constraints they appear only in the full version [37]. We show some examples of these cases in Figures 3 and 4.

▶ **Lemma 10.** *After applying all reduction rules exhaustively, let $(G, \mathcal{H}, \ell = 0)$ be an instance of CEAHMP which has a solution S. Then there is no clique of size at least 4 in $G \triangle S$.*

Next, we introduce a new problem called CLUSTER DELETION ABOVE MODIFICATION-DISJOINT P_3 PACKING (CDAMP), defined as follows: Given a graph $G = (V, E)$, a modification-disjoint packing $\mathcal{H}$ of induced P_3s of G, and a non-negative integer ℓ, decide whether there is a *cluster-deletion set*, that is, a set of edges $S \subseteq E$ so that $G' = (V, E \setminus S)$ is a union of disjoint cliques, with $|S| - |\mathcal{H}| \leq \ell$.

▶ **Lemma 11.** *Let $(G, \mathcal{H}, \ell = 0)$ be an instance of CEAHMP. After applying all reduction rules exhaustively, we get an instance $(G', \mathcal{H}', \ell = 0)$ of CEAHMP. Then $(G, \mathcal{H}, \ell = 0)$ is a YES-instance of CEAHMP if and only if $(G', \mathcal{H}', \ell = 0)$ is a YES-instance of CDAMP.*

Let $(G', \mathcal{H}', \ell = 0)$ be the resulting instance of CDAMP. Let $E_c \subseteq E(G')$ be the set of edges covered by some P_3 of $\mathcal{H}'$ and let $\lambda = 2|\mathcal{H}'|$. We fix an arbitrary ordering of the edges of E_c and label these edges by $e_0, e_1, ..., e_{\lambda-1}$ according to this ordering. We construct an instance of 2-SAT with λ variables $x_0, x_1, ..., x_{\lambda-1}$ as follows. First, initialize the 2-SAT formula $\Phi = \mathsf{true}$. For each induced P_3 $xyz \in \mathcal{H}'$, let $e_i = xy, e_j = yz$ and update $\Phi \leftarrow \Phi \wedge (x_i \vee x_j) \wedge (\neg x_i \vee \neg x_j)$. For each induced P_3 uvw in G' such that uv and vw belong to two distinct P_3s of $\mathcal{H}'$ respectively, if $uv = e_p$ and $vw = e_q$, then update $\Phi \leftarrow \Phi \wedge (x_p \vee x_q)$. This completes the construction of the 2-SAT instance.

▶ **Lemma 12.** *Let $(G', \mathcal{H}', \ell = 0)$ be an instance of CDAMPand construct a 2-SAT formula Φ as described above. Then $(G, \mathcal{H}, \ell = 0)$ is a YES-instance if and only if Φ is satisfiable.*

As a result we can reduce the problem to 2-SAT and solve it in polynomial time.

5 Conclusion

Unfortunately the lower bound that we have obtained is a major roadblock in designing fixed-parameter algorithms for CLUSTER EDITING parameterized above modification-disjoint P_3s. On the positive side, CLUSTER EDITING ABOVE HALF-INTEGRAL P_3 PACKING (CEAHMP) admits an XP-algorithm with respect to the number of excess edits. We have left open whether CEAHMP is fixed-parameter tractable. Towards this, on the one hand the half-integral P_3 packings provide quite strong structure that can be exploited to design several branching rules. On the other hand, when attacking this question from several angles we discovered large grid-like structures that seemed difficult to overcome in fixed-parameter time, and a corresponding W[1]-hardness result would also not be surprising.

A different future research direction is to deconstruct our hardness reduction by examining which substructures it contains that are seldom in practical data. Forbidding such substructures may destroy the already somewhat fragile hardness construction, perhaps paving the way for fixed-parameter algorithms.

Finally, it would be interesting to see how modification-disjoint P_3 packings look in practice. If it is true that only few vertices are in a large number of packed P_3s and most of them are in a small constant number, then a strategy that settles the clustering around the vertices with large number of P_3s and then applies reduction rules from Section 4 could be efficient.

References

1 Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5):23:1–23:27, 2008. doi:10.1145/1411509.1411513.

2 Takuya Akiba and Yoichi Iwata. Branch-and-reduce exponential/FPT algorithms in practice: A case study of vertex cover. *Theoretical Computer Science*, 609:211–225, 2016. doi:10.1016/j.tcs.2015.09.023.

3 Noga Alon, Konstantin Makarychev, Yury Makarychev, and Assaf Naor. Quadratic forms on graphs. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC 2005)*, pages 486–493. ACM, 2005. doi:10.1145/1060590.1060664.

4 Sanjeev Arora, Eli Berger, Elad Hazan, Guy Kindler, and Muli Safra. On non-approximability for quadratic programs. *Electronic Colloquium on Computational Complexity (ECCC)*, 058, 2005. URL: http://eccc.hpi-web.de/eccc-reports/2005/TR05-058/index.html.

5 Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine Learning*, 56(1-3):89–113, 2004. doi:10.1023/B:MACH.0000033116.57574.95.

6 Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3/4):281–297, 1999. doi:10.1089/106652799318274.

7 S. Böcker, S. Briesemeister, Q.B.A. Bui, and A. Truss. Going weighted: Parameterized algorithms for cluster editing. *Theoretical Computer Science*, 410(52):5467–5480, 2009. doi:10.1016/j.tcs.2009.05.006.

8 Sebastian Böcker. A golden ratio parameterized algorithm for cluster editing. *Journal of Discrete Algorithms*, 16:79–89, 2012. doi:10.1016/j.jda.2012.04.005.

9 Sebastian Böcker and Jan Baumbach. Cluster editing. In Paola Bonizzoni, Vasco Brattka, and Benedikt Löwe, editors, *Proceedings of the 9th Conference on Computability in Europe (CiE 2013)*, volume 7921 of *Lecture Notes in Computer Science*, pages 33–44. Springer, 2013. doi:10.1007/978-3-642-39053-1_5.

10 Sebastian Böcker, Sebastian Briesemeister, and Gunnar W. Klau. Exact algorithms for cluster editing: Evaluation and experiments. *Algorithmica*, 60(2):316–334, 2011. doi:10.1007/s00453-009-9339-7.

11 Sebastian Böcker and Peter Damaschke. Even faster parameterized cluster deletion and cluster editing. *Information Processing Letters*, 111(14):717–721, 2011. doi:10.1016/j.ipl.2011.05.003.

12 Hans L. Bodlaender, Michael R. Fellows, Pinar Heggernes, Federico Mancini, Charis Papadopoulos, and Frances A. Rosamond. Clustering with partial information. *Theoretical Computer Science*, 411(7-9):1202–1211, 2010. doi:10.1016/j.tcs.2009.12.016.

13 Nicolas Bousquet, Jean Daligault, and Stéphan Thomassé. Multicut is FPT. *SIAM Journal on Computing*, 47(1):166–207, 2018. doi:10.1137/140961808.

14 Yixin Cao and Jianer Chen. Cluster editing: Kernelization based on edge cuts. *Algorithmica*, 64(1):152–169, 2012. doi:10.1007/s00453-011-9595-1.

15 Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3):360–383, 2005. doi:10.1016/j.jcss.2004.10.012.

16 Jianer Chen and Jie Meng. A $2k$ kernel for the cluster editing problem. *Journal of Computer and System Sciences*, 78(1):211–220, 2012. doi:10.1016/j.jcss.2011.04.001.

17 Christophe Crespelle, Pål Grønås Drange, Fedor V. Fomin, and Petr A. Golovach. A survey of parameterized algorithms and the complexity of edge modification. *arXiv:2001.06867 [cs]*, 2020. arXiv:2001.06867.

18 Marek Cygan, Marcin Pilipczuk, Michal Pilipczuk, and Jakub Onufry Wojtaszczyk. On multiway cut parameterized above lower bounds. *ACM Transactions on Computation Theory*, 5(1):3:1–3:11, 2013. doi:10.1145/2462896.2462899.

19 Peter Damaschke. Fixed-parameter enumerability of cluster editing and related problems. *Theory of Computing Systems*, 46(2):261–283, 2010. doi:10.1007/s00224-008-9130-1.

20 Michael R. Fellows. The lost continent of polynomial time: Preprocessing and kernelization. In Hans L. Bodlaender and Michael A. Langston, editors, *Proceedings of the Second International Workshop on Parameterized and Exact Computation (IWPEC 2006)*, volume 4169 of *Lecture Notes in Computer Science*, pages 276–277. Springer, 2006. doi:10.1007/11847250_25.

21 Michael R. Fellows, Jiong Guo, Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. Graph-based data clustering with overlaps. *Discrete Optimization*, 8(1):2–17, 2011. doi:10.1016/j.disopt.2010.09.006.

22 Michael R. Fellows, Michael A. Langston, Frances A. Rosamond, and Peter Shaw. Efficient parameterized preprocessing for cluster editing. In Erzsébet Csuhaj-Varjú and Zoltán Ésik, editors, *Proceedings of the 16th International Symposium on Fundamentals of Computation Theory (FCT 2007)*, volume 4639 of *Lecture Notes in Computer Science*, pages 312–321. Springer, 2007. doi:10.1007/978-3-540-74240-1_27.

23 Fedor V. Fomin, Stefan Kratsch, Marcin Pilipczuk, Michał Pilipczuk, and Yngve Villanger. Subexponential fixed-parameter tractability of cluster editing. *arXiv:1112.4419 [cs]*, 2013. arXiv:1112.4419.

24 Fedor V. Fomin, Stefan Kratsch, Marcin Pilipczuk, Michał Pilipczuk, and Yngve Villanger. Tight bounds for parameterized complexity of cluster editing with a small number of clusters. *Journal of Computer and System Sciences*, 80(7):1430–1447, 2014. doi:10.1016/j.jcss.2014.04.015.

25 Vincent Froese. *Fine-Grained Complexity Analysis of Some Combinatorial Data Science Problems*. PhD thesis, Technische Universität Berlin, 2018. doi:10.14279/depositonce-7123.

26 Shivam Garg and Geevarghese Philip. Raising the bar for vertex cover: Fixed-parameter tractability above A higher guarantee. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2016)*, pages 1152–1166. SIAM, 2016. doi:10.1137/1.9781611974331.ch80.

27 Jens Gramm, Jiong Guo, Falk Hüffner, and Rolf Niedermeier. Automated generation of search tree algorithms for hard graphmodification problems. *Algorithmica*, 39(4):321–347, 2004. doi:10.1007/s00453-004-1090-5.

28 Jens Gramm, Jiong Guo, Falk Hüffner, and Rolf Niedermeier. Graph-modeled data clustering: Exact algorithms for clique generation. *Theory of Computing Systems*, 38(4):373–392, 2005. doi:10.1007/s00224-004-1178-y.

29 Jiong Guo. A more effective linear kernelization for cluster editing. *Theoretical Computer Science*, 410(8):718–726, 2009. doi:10.1016/j.tcs.2008.10.021.

30 Jiong Guo, Iyad A. Kanj, Christian Komusiewicz, and Johannes Uhlmann. Editing graphs into disjoint unions of dense clusters. *Algorithmica*, 61(4):949–970, 2011. doi:10.1007/s00453-011-9487-4.

31 Jiong Guo, Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. A more relaxed model for graph-based data clustering: s-plex cluster editing. *SIAM Journal on Discrete Mathematics*, 24(4):1662–1683, 2010. doi:10.1137/090767285.

32 Yoichi Iwata. Linear-time kernelization for feedback vertex set. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *Proceedings of the 44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*, volume 80 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 68:1–68:14. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.68.

33 Bart M.P. Jansen, Christian Schulz, and Hisao Tamaki. NII shonan meeting report no. 144 parameterized graph algorithms and data reduction, 2019. URL: https://shonan.nii.ac.jp/docs/No.144.pdf.

34 Krzysztof Kiljan and Marcin Pilipczuk. Experimental evaluation of parameterized algorithms for feedback vertex set. In Gianlorenzo D'Angelo, editor, *Proceedings of the 17th International Symposium on Experimental Algorithms (SEA 2018)*, volume 103 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:12, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.SEA.2018.12.

35 Christian Komusiewicz and Johannes Uhlmann. Cluster editing with locally bounded modifications. *Discrete Applied Mathematics*, 160(15):2259–2270, 2012. doi:10.1016/j.dam.2012.05.019.

36 Stefan Kratsch. A randomized polynomial kernelization for vertex cover with a smaller parameter. *SIAM Journal on Discrete Mathematics*, 32(3):1806–1839, 2018. doi:10.1137/16M1104585.

37 Shaohua Li, Marcin Pilipczuk, and Manuel Sorge. Cluster editing parameterized above modification-disjoint P_3-packings. *arXiv:1910.08517 [cs]*, 2019. arXiv:1910.08517.

38 Daniel Lokshtanov, N. S. Narayanaswamy, Venkatesh Raman, M. S. Ramanujan, and Saket Saurabh. Faster parameterized algorithms using linear programming. *ACM Transactions on Algorithms*, 11(2):15:1–15:31, 2014. doi:10.1145/2566616.

39 Meena Mahajan and Venkatesh Raman. Parameterizing above guaranteed values: Maxsat and maxcut. *Journal of Algorithms*, 31(2):335–354, 1999. doi:10.1006/jagm.1998.0996.

40 Dániel Marx and Igor Razgon. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM Journal on Computing*, 43(2):355–388, 2014. doi:10.1137/110855247.

41 John H. Morris, Leonard Apeltsin, Aaron M. Newman, Jan Baumbach, Tobias Wittkop, Gang Su, Gary D. Bader, and Thomas E. Ferrin. clusterMaker: a multi-algorithm clustering plugin for Cytoscape. *BMC Bioinformatics*, 12(1):436, 2011. doi:10.1186/1471-2105-12-436.

42 Fábio Protti, Maise Dantas da Silva, and Jayme Luiz Szwarcfiter. Applying modular decomposition to parameterized cluster editing problems. *Theory of Computing Systems*, 44(1):91–104, 2009. doi:10.1007/s00224-007-9032-7.

43 Ron Shamir, Roded Sharan, and Dekel Tsur. Cluster graph modification problems. *Discrete Applied Mathematics*, 144(1-2):173–182, 2004. doi:10.1016/j.dam.2004.01.007.

44 René van Bevern, Vincent Froese, and Christian Komusiewicz. Parameterizing edge modification problems above lower bounds. *Theory of Computing Systems*, 62(3):739–770, 2018. doi:10.1007/s00224-016-9746-5.

45 Tobias Wittkop, Dorothea Emig, Sita Lange, Sven Rahmann, Mario Albrecht, John H. Morris, Sebastian Böcker, Jens Stoye, and Jan Baumbach. Partitioning biological data with transitivity clustering. *Nature Methods*, 7(6):419–420, 2010. doi:10.1038/nmeth0610-419.