Cluster Editing Parameterized above Modification-disjoint P_3-packings

SHAOHUA LI and MARCIN PILIPCZUK, Institute of Informatics, University of Warsaw, Poland
MANUEL SORGE, University of Warsaw, Poland and TU Wien, Austria

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. Here packed P_3s are those belonging to the packing $\mathcal{H}$. 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.

CCS Concepts: • **Theory of computation → Parameterized complexity and exact algorithms**; **Unsupervised learning and clustering**; **Problems, reductions and completeness**; • **Mathematics of computing** → *Graph theory;*

Additional Key Words and Phrases: Graph algorithms, fixed-parameter tractability, parameterized complexity

ACM Reference format:
Shaohua Li, Marcin Pilipczuk, and Manuel Sorge. 2023. Cluster Editing Parameterized above Modification-disjoint P_3-packings. *ACM Trans. Algor.* 20, 1, Article 3 (December 2023), 43 pages.
https://doi.org/10.1145/3626526

An extended abstract of this work appeared at STACS 2021 [37].
This research is part of 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. MS also gratefully acknowledges funding by the Alexander von Humboldt Foundation.


Authors' addresses: S. Li, CISPA Helmholtz Center for Information Security, Kaiserstr. 21, 66386 St. Ingbert, Saarland, Germany; e-mail: shaohua.li@mimuw.edu.pl; M. Pilipczuk, Institute of Informatics, University of Warsaw, Banacha 2, 02-097 Warszawa, Poland; e-mail: malcin@mimuw.edu.pl; M. Sorge, Technische Universität Wien, Institute of Logic and Computation, Favoritenstraße 9-11, E192-01, 1040 Wien, Austria; e-mail: manuel.sorge@ac.tuwien.ac.at.

1549-6325/2023/12-ART3
https://doi.org/10.1145/3626526

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, 3–6, 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 [7, 9–13, 19, 21, 24, 28, 30, 31, 35, 40, 42]. 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, 24, 35, 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 [7, 9, 11, 27, 28] continually improved the base in the exponential part of the running time, culminating in the current fastest fixed-parameter algorithm with running time $O(1.62^k + n + m)$ [7], where n is the number of vertices of the input graph and m its number of edges. Similarly, a series of papers [14, 16, 20, 22, 28, 29, 42] 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 [8, 10], the parameter k is not very small in several real-world data sets [9, 10, 44]. For instance, Böcker et al. [9, 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 [18, 26, 36, 38, 39], 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

[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.

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 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 nor 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 [23, 35] 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 Modification-Disjoint 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 Modification-Disjoint P_3 Packing parameterized by the number ℓ of excess edits is in XP. It can be solved in $n^{2\ell+O(1)}$ 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 2-SAT formula which we then solve in polynomial time.

Organization. After brief preliminaries in Section 2, we give some intuition about CEaMP in Section 3. Then we proceed to the reduction used to show Theorem 1 in Section 4.1 (containing the construction) and Section 4.2 (containing the correctness proof). Section 5 then contains the proof of Theorem 2.

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.

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 none of the edges of a path P is packed, we say that the path P is *non-packed*. 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. Let $x \in \mathbb{F}_p$. Where it is not ambiguous, $-x$ and x^{-1} will denote the additive and multiplicative inverse, respectively, of x in $\mathbb{F}_p$.

When we say that we relabel the vertices of a graph, we use $v \leftarrow u$ to denote that we relabel the vertex v by the new label u.

3 INTUITION

Before giving the hardness proof, it is instructive to determine some easy and difficult cases when solving CEAMP with $\ell = 0$. This will give us an intuition about the underlying combinatorial problem that we need to solve.

Let $(G, \mathcal{H}, 0)$ be an instance of CEAMP. It is helpful to consider the subgraph G_{fix} of G that contains only those edges of G that are not contained in any P_3 in $\mathcal{H}$, that is, the non-packed edges. Suppose that $(G, \mathcal{H}, 0)$ has a solution S and let G_{sol} be the associated cluster graph. Observe that each connected component of G_{fix} is part of a single cluster in G_{sol}. Let us hence call the connected components of G_{fix} *proto-clusters*. Our task in finding G_{sol} is thus indeed to find a vertex partition $\mathcal{P}$ that is coarser than the vertex partition given by the proto-clusters and that satisfies certain further conditions. The additional conditions herein are given by the P_3s in G and also by the non-edges of G which are not contained in any P_3 in $\mathcal{H}$, that is, by the non-packed non-edges. A non-packed non-edge between two proto-clusters implies that these proto-clusters cannot be together in a cluster in G_{sol}. Hence, we are searching for a vertex partition $\mathcal{P}$ as above subject to the constraints that certain proto-cluster pairs end up in different parts.

The constraints on $\mathcal{P}$ given by P_3s in G can be distinguished based on the intersection of the P_3s with the proto-clusters. We only want to highlight two situations that are most relevant for the hardness construction. The first situation is when a P_3, name it P, intersects with three proto-clusters D_1, D_2, and D_3, each in exactly one vertex and with center vertex in D_2. The corresponding constraint on $\mathcal{P}$ is that either D_1 and D_2 are merged or D_2 and D_3 are merged into one cluster. We can satisfy such constraints easily, in the absence of further constraints, by merging all proto-clusters into one large cluster. However, together with the constraints from non-packed non-edges a difficult picture emerges. Consider Figure 1. Proto-clusters B and D cannot be merged into one cluster because of a non-packed non-edge. However, there is a path in G from B to D via vertices of C. Hence, either B and C are in different clusters in G_{sol} or C and D are. If B and C are in different clusters, then since we have only budget one for the P_3 involving A, B, and C, it follows that A and B are merged into one cluster in G_{sol}. It is not hard to imagine that such behavior can be very non-local and in fact two different generalizations of this behavior form the basis for the variable and clause gadget in our hardness reduction.

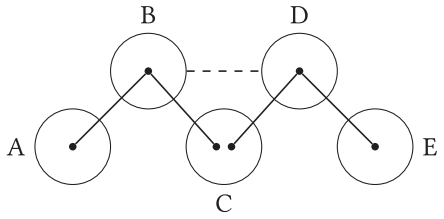

Fig. 1. Five proto-clusters A through E and two P_3s in the underlying graph and in the P_3-packing that connects A to C via B and C to E via D, respectively. The dashed edge between B and D means that there is a non-packed non-edge between B and D.

The second case is when there is a P_3 in G and also in the packing $\mathcal{H}$ that has an edge contained in one proto-cluster A and the remaining vertex in a different proto-cluster B. Call this P_3 P. Intuitively, regardless of whether A and B are merged into one cluster in G_{sol}, P can be edited without excess cost over $\mathcal{H}$ to accommodate this choice. In our hardness reduction, a main difficulty will be to pad subconstructions with P_3s in the packing $\mathcal{H}$, so that we are able to find a solution with zero excess edits. For this we will heavily use P_3s of the form that we just described.

4 NP-HARDNESS FOR TIGHT MODIFICATION-DISJOINT PACKINGS

In this section, we prove Theorem 1 by showing a reduction from the NP-hard problem of deciding satisfiability of 3-CNF formulas. Given a 3-CNF formula Φ, we construct a graph $G = (V, E)$ with a modification-disjoint $\mathcal{H}$ 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 vertex pair of each P_3 in $\mathcal{H}$. In other words, the CEAMP instance $(G, \mathcal{H}, 0)$ is a YES-instance. We assume that every clause of Φ has exactly 3 literals of pair-wise different variables as we can preprocess the formula to achieve this in polynomial time otherwise. Similarly, we can assume that every variable of Φ appears at least twice. In the following, we let m denote the number of clauses in Φ, denote the clauses of Φ by $\Gamma_0, \ldots, \Gamma_{m-1}$, let n be the number of variables, and denote the variables of Φ by $x_0, \ldots, x_{n-1}$. Furthermore, we let m_i denote the number of clauses that contain the variable x_i, $i = 0, \ldots, n-1$.

4.1 Construction

The outline of our construction is as follows. In Section 4.1.1 and 4.1.2 we explain the basic construction of the variable and clause gadgets. In these two sections we first show how to construct a subgraph of the final construction that enables us to show the soundness, that is, if the CEAMP instance is a yes-instance, then Φ is satisfiable. The main difficulty is then to extend this construction so that the completeness also holds. This we do in Section 4.1.3 and 4.1.4. Section 4.2.1 and 4.2.2 then contain the correctness proof.

Both the variable gadget and the clause gadget rely on some ideas outlined in Section 3. Our basic building blocks will be proto-clusters. A proto-cluster is a subgraph that is connected through edges that are not contained in any P_3 in the constructed packing $\mathcal{H}$. The proto-clusters then have to be joined into larger clusters in a way that represents a satisfying assignment to Φ. The variable gadget basically consists of an even-length cycle of proto-clusters, connected by P_3s so that either odd or even pairs of proto-clusters on the cycle have to be merged. These two options represent a truth assignment. The construction of the variable gadget is more involved than a simple cycle of proto-clusters, however, because of the connection to the clause gadgets: We need to ensure that all vertex pairs between certain proto-clusters of a variable and clause gadget are covered by P_3s in $\mathcal{H}$, so to be able to merge these clusters in the completeness proof. The way in which we cover

these vertex pairs imposes some constraints on the construction of the variable gadgets, making the gadgets more complicated.

4.1.1 Variable Gadget. 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. It would be tempting to make each proto-cluster a single vertex. However, due to the connections to the clause gadget later on, we need proto-clusters containing five vertices each.

Recall that m_i denotes 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 $K^i_0, \ldots, K^i_{4m_i-1}$. In each K^i_j, $j = 0, 1, \ldots, 4m_i - 1$, the vertices are $v^i_{j,0}, \ldots, v^i_{j,4}$. For each $j = 0, 2, \ldots, 4m_i - 2$, we create P_3s connecting K^i_j, K^i_{j+1} and K^i_{j+2} (where we identify K^i_0 as $K^i_{4m_i}$) as we explain below, adding all edges between each two consecutive cliques.

Throughout the construction, the cliques we have just introduced will remain proto-clusters, that is, they contain a spanning tree of edges that are not covered by P_3s in the packing $\mathcal{H}$. We now add pairwise modification-disjoint P_3s so as to cover all edges between the cliques K^i_j 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 by the arithmetic progression.

Formally, for each $j = 0, 2, \ldots, 4m_i - 2$ and 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 we add 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 modification-disjoint packing of the variable gadget to be

$$\mathcal{H}_{\text{var}} = \{P_3 \text{ given by } v^i_{j,p}v^i_{j+1,q}v^i_{j+2,r} \mid i = 0, \ldots, n-1; j = 0, 2, \ldots, 4m_i - 2; p, q, r \in \mathbb{F}_5; \text{and } q - p = r - q\}.$$

This finishes the first stage of the construction. Notice that the cliques K^i_j form a cyclic structure. Intuitively, every second pair of cliques needs to be merged into one cluster by any solution due to the P_3s we have introduced, and we will see that the two resulting solutions are in fact the only ones. The truth values of the variable are then 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 $j = 0, 2, \ldots, 4m_i - 2$, then this represents the

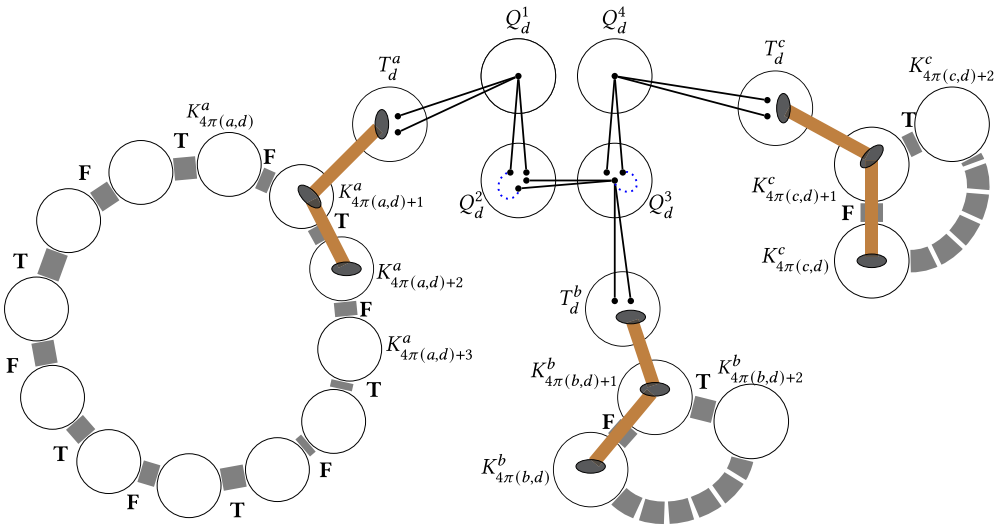

Fig. 2. Skeleton of a clause gadget $\Gamma_d = (x_a \vee \neg x_b \vee \neg x_c)$. The white circles represent cliques. The blue dotted lines inside Q_d^2 and Q_d^3 indicate that Q_d^1, Q_d^2, Q_d^3, and Q_d^4 are in one connected component. A pair of incident brown thick lines indicates a set of four transferring P_3s used to connect a clause gadget to a variable gadget. The cycles made from cliques and gray thick lines represent variable gadgets, where a dashed gray line indicates an omitted part of the cycle. The cycle for variable x_a is shown completely, where we assume that $m_a = 3$, that is, variable x_a is in three clauses. Labels **T** and **F** on thick gray edges indicate the pairs of cliques that shall be merged into one cluster if the variable is to be set to true or false, respectively.

situation that we assign false to the variable x_i. If K^i_{j+1} and K^i_{j+2} are merged for $j = 0, 2, \ldots, 4m_i - 2$, then this represents variable x_i being true. We will make minor modifications to the variable gadgets and $\mathcal{H}_{\text{var}}$ in the following section, so as to transmit the choice of truth value to the clause gadgets.

4.1.2 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 is a subgraph of the final construction that allows us to prove the soundness. The final construction is given in the succeeding sections. We give a picture of the skeleton in Figure 2. The basic idea is a generalization of the idea explained in Section 3: A clause Γ_d is represented by four proto-clusters (cliques), Q_d^i, $i = 1, \ldots, 4$, as in Figure 2. The proto-clusters are connected by a path P of length 5 containing vertices of Q_d^1, Q_d^2, Q_d^3, and Q_d^4 in that order. However, between Q_d^1 and Q_d^4 there is a non-packed non-edge, meaning that every solution has to cut the path P by deleting all edges between Q_d^1 and Q_d^2, or between Q_d^2 and Q_d^3, or between Q_d^3 and Q_d^4. We use this three-way choice to force the solution to select a variable that satisfies the clause Γ_d.

Main gadget. Formally, for each variable x_i, $i = 0, 1, \ldots, n-1$, we fix an arbitrary ordering of the clauses that contain x_i. If a clause Γ_j contains a variable 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 $\mathcal{H}_{\text{tra}} = \emptyset$. For each clause Γ_d ($d = 0, \ldots, m-1$) proceed as follows. We first introduce four cliques Q_d^1, Q_d^2, Q_d^3 and Q_d^4. Let Γ_d contain the variables x_a, x_b and x_c. We introduce the cliques T_d^a, T_d^b and T_d^c, called *transferring cliques*. All of the cliques introduced are pairwise vertex disjoint and can be of different sizes. We will give the exact sizes in Section 4.1.4.

Next, we introduce the following P_3s:

- Introduce two P_3s, P_d^1 and P_d^2, that both connect T_d^a and Q_d^2 via Q_d^1, such that P_d^1 and P_d^2 share the same vertex in Q_d^1.
- Introduce two P_3s, P_d^3 and P_d^4, that both connect T_d^b and Q_d^2 via Q_d^3, such that P_d^3 and P_d^4 share the same vertex in Q_d^3.
- Introduce two P_3s, P_d^5 and P_d^6, that both connect T_d^c and Q_d^3 via Q_d^4, such that P_d^5 and P_d^6 share the same vertex in Q_d^4.

All the P_3s P_d^i are pairwise vertex-disjoint except for the pairs sharing the center (as explicitly mentioned in the description). We add each P_d^i for $i = 1, \ldots, 6$ to $\mathcal{H}_{\text{tra}}$. We call the P_3s of $\mathcal{H}_{\text{tra}}$ *transferring* P_3*s*.

Connection to the variable gadgets. Next we connect the transferring cliques T_d^a, T_d^b, and T_d^c to the variable gadgets of x_a, x_b, and x_c, respectively. To avoid additional notation, we only explain the procedure for T_d^a and x_a, the other pairs are connected analogously. We connect T_d^a to the variable gadget of x_a by a set of four modification-disjoint P_3s as shown in Figure 3 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_d^a which is different from the endpoints of the P_3s connecting T_d^a to Q_d^1; 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 $K^a_{4\pi(a,d)+2}$ 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 $\mathcal{H}_{\text{var}}$ that were introduced while constructing the variable gadgets, we will remove such P_3s in the variable gadget from $\mathcal{H}_{\text{var}}$, remove their corresponding edges from the graph, and add some new P_3s to $\mathcal{H}_{\text{var}}$ 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 $\mathcal{H}_{\text{var}}$ are pairwise modification-disjoint.

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

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

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:

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

As shown in Figure 3, we remove P_3s given by $v_8v_1v_3$, $v_7v_1v_4$, $v_6v_2v_3$, $v_5v_2v_4$ from $\mathcal{H}_{\text{var}}$ 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 $\mathcal{H}_{\text{var}}$. Finally, we connect T_d^a 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 $\mathcal{H}_{\text{tra}}$. 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.

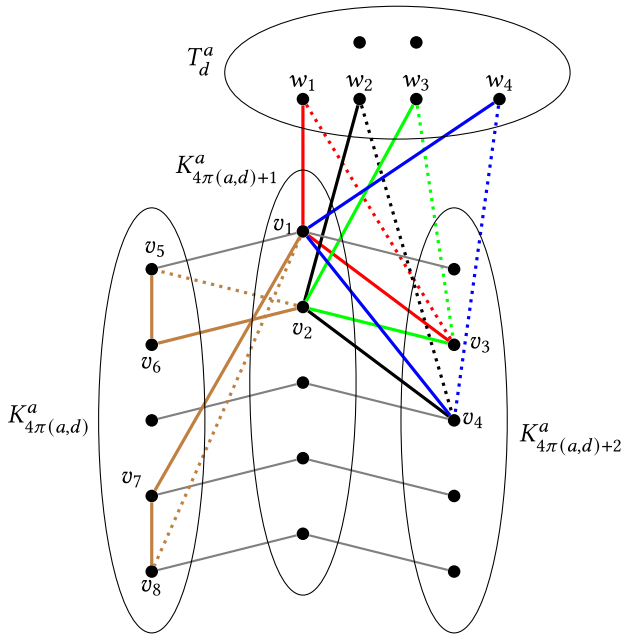

Fig. 3. Connection of a clause gadget with a variable gadget for a variable x_a which appears positively in the clause. White ellipses represent cliques. The vertices in the cliques in the variable gadget are ordered from top to bottom according to the elements of $\mathbb{F}_5$ which they represent. For example, the topmost 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 bottom-most 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. (Some gray lines are super-seeded by edges of other colors.) The P_3s represented by the gray lines have the associated arithmetic progression "+0", that is, $q - p = r - q = 0$ in the definition of the P_3s. The P_3s for the remaining arithmetic progressions are omitted for clarity. 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 from both G and $\mathcal{H}$. 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 two brown P_3s.

The intuitive idea behind the connection to the variable gadget and how it is used in the soundness proof is 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 2. Since we can modify at most one vertex pair for each of 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.

The P_3s added so far are indeed sufficient to conduct a soundness proof of the above reduction: They ensure that there exists a satisfying assignment to the input formula provided that there

exists an appropriate cluster editing set. However, the completeness is much more difficult: We need to add some more "padding" P_3s to the packing (and edges to the graph between the cliques that can be potentially merged) to ensure that a satisfying assignment can always be translated into a cluster-editing set. The goal of the next two sections is to develop a methodology of padding such cliques with P_3s in the packing. The padding will rely on the special structure of P_3s that we have established above in the clause gadget and connection between clause and variable gadget.

4.1.3 Merging Model of the Clause Gadget. In the sections above, we have defined all proto-clusters of the final constructed graph: As we will see in the correctness proof, each clique will be a proto-cluster in the end. Thus, all solutions will construct a cluster graph whose clusters represent a coarser partition than the partition given by the proto-clusters, or cliques. 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. To do this, we pad the proto-clusters with P_3s (in the graph and packing $\mathcal{H}$). To simplify this task we now divide the set of proto-clusters into five levels $L_0, \ldots, L_4$. Then, we will go through the levels in increasing order and add padding P_3s from proto-clusters of the current level to proto-clusters of all lower levels if necessary.

There are two issues that we need to deal with when introducing the padding P_3s. For the padding, we will use a number-theoretic tool that we introduce in Section 4.1.4 which has the limitation that, when padding a proto-cluster D with P_3s to some sequence $D_1, \ldots, D_s$ of proto-clusters of lower level, we need to increase the number of vertices in D to be roughly $2 \cdot \sum_{i=1}^{s} |D_i|$. Hence, first, we need to make sure that the number of levels is constant since the number of size increases of proto-clusters compounds exponentially with the number of levels. Second, we aim for the property that each vertex is only in a constant number of P_3s in $\mathcal{H}$ and thus, we need to ensure that the number s of lower-level proto-clusters and their size is constant.

To achieve the above goals, we introduce an auxiliary graph H, the *merging model*, which will further guide the padding process. The merging model has as vertices the cliques that were introduced before and an edge between two cliques if we want it to be possible that they are merged by a solution. Formally,

$$\begin{aligned} V(H) := {} & \{K_j^i \mid i = 0, 1, \ldots, n-1 \text{ and } j = 0, 1, \ldots, 4m_i - 1\} \\ & \cup \{Q_d^1, Q_d^2, Q_d^3, Q_d^4 \mid d = 0, 1, \ldots, m-1\} \\ & \cup \{T_s^a \mid \text{variable } x_a \text{ occurs in clause } \Gamma_s\}, \end{aligned}$$

and the edge set, $E(H)$, is defined as follows. See also Figure 4. First, it shall be possible to merge the cliques in the variable gadget in a cyclic fashion,[2] that is, we add

$$\{\{K_j^i, K_{j+1}^i\} \mid i = 0, 1, \ldots, n-1 \text{ and } j = 0, 1, \ldots, 4m_i - 1\}$$

to $E(H)$. Second, it shall be possible to merge transferring cliques of a clause gadget to any of the relevant cliques of the associated variable gadget, that is, we add to $E(H)$ the set

$$\{\{T_d^i, K_{4\pi(i,d)}^i\}, \{T_d^i, K_{4\pi(i,d)+1}^i\}, \{T_d^i, K_{4\pi(i,d)+2}^i\} \mid \text{variable } x_i \text{ occurs in clause } \Gamma_d\}.$$

Third, it shall be possible to merge subsets of $\{Q_d^1, Q_d^2, Q_d^3, Q_d^4\}$, and hence we add to $E(H)$ the set

$$\{\{Q_d^1, Q_d^2\}, \{Q_d^1, Q_d^3\}, \{Q_d^2, Q_d^3\}, \{Q_d^2, Q_d^4\}, \{Q_d^3, Q_d^4\} \mid d = 0, 1, \ldots, m-1\}.$$

[2]Indeed, we have already ensured that this is possible. The edges introduced in the first step purely serve to reinforce the intuition of the merging model.

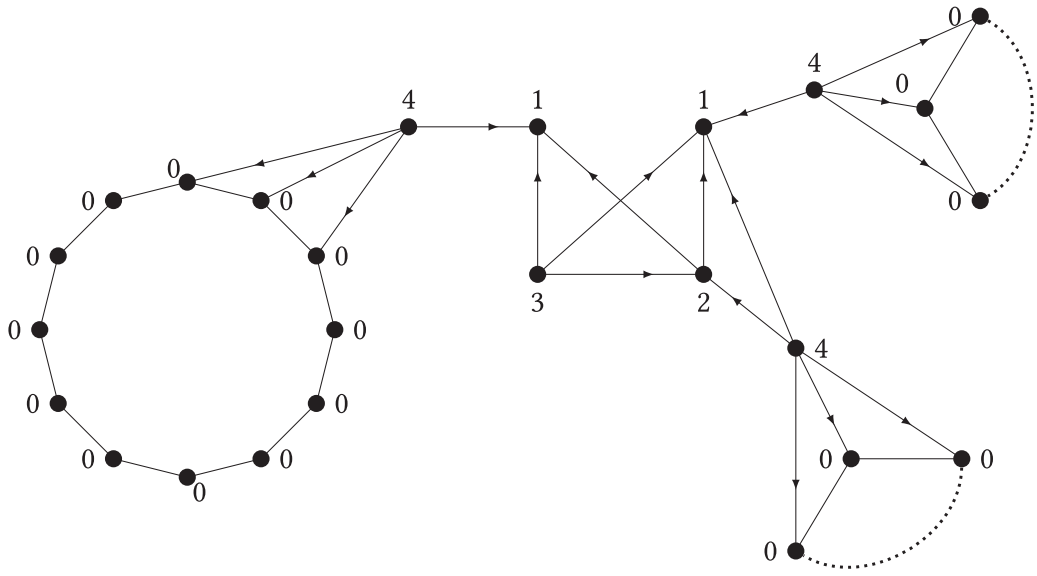

Fig. 4. Merging model of a clause $\Gamma_d = (x_a \vee \neg x_b \vee \neg x_c)$. The number $i \in \{0, 1, 2, 3, 4\}$ beside a vertex v denotes that $v \in L_i$. The placement of vertices corresponds to the placement of the cliques in Figure 2. For example, the two vertices of level 1 on the top correspond to Q_d^1 and Q_d^4. We assume that $m_a = 3$.

Finally, it shall be possible to merge the transferring cliques to subsets of $\{Q_d^1, Q_d^2, Q_d^3, Q_d^4\}$. Hence, we add to $E(H)$ the set

$$\{\{T_d^i, Q_d^k\} \mid \text{if variable } x_i \text{ occurs in } \Gamma_d \text{ and } T_d^i \text{ is adjacent in } G \text{ to } Q_d^k \text{ with } k \in \{1, 4\}\}$$
$$\cup \{\{T_d^i, Q_d^3\}, \{T_d^i, Q_d^4\} \mid \text{if variable } x_i \text{ occurs in } \Gamma_d \text{ and } T_d^i \text{ is adjacent in } G \text{ to } Q_d^3\}.$$

Note that this construction is slightly asymmetric (see Figure 4).

Now we define the levels L_0 to L_4 such that orienting the edges in H from higher to lower level gives an acyclic orientation when ignoring the edges in level L_0.

- L_0 contains all cliques in variable gadgets.
- L_1 contains Q_d^1 and Q_d^4 for each $d = 0, \ldots, m-1$.
- L_2 contains Q_d^3 for each $d = 0, \ldots, m-1$.
- L_3 contains Q_d^2 for each $d = 0, \ldots, m-1$.
- L_4 contains all transferring cliques.

We now orient all edges in H from higher-level vertices to lower-level vertices. Edges in level L_0 remain undirected. Observe that, apart from edges in L_0, all edges in H are between vertices of different levels and, indeed, ignoring edges in L_0, there are no cycles in G when orienting the edges from higher to lower level. In the following section, we will look at each clique R in levels L_1 and higher, and add P_3s to the packing $\mathcal{H}$ so as to cover all vertex pairs containing a vertex of R and an out-neighbor of R in H.

4.1.4 Implementation of the Clause Gadget. In this section, we first introduce a number-theoretical construction (Lemma 1) that serves as a basic building block for "padding" P_3s in the packing. Then we use this construction to perform the actual padding of P_3s.

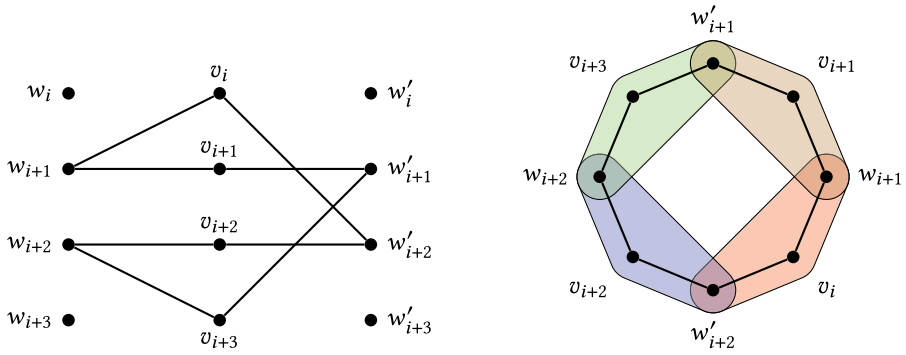

Fig. 5. Left: The labels of a C_8 in $(V \cup W, F)$. Right: The triangles in τ_F^2 covering a C_8.

The abstract process of padding P_3s works as follows. It takes as input a clique R in H (represented by W in the below Lemma 1), and a set of cliques that are out-neighbors of R in H (represented by V). Furthermore, it receives a set of vertex pairs between R and its out-neighbors that have previously been covered (represented by F). The goal is then to find a packing of P_3s that cover all vertex pairs *except* the previously covered pairs. The previously covered vertex pairs have some special structure that we carefully selected so as to make covering of all remaining vertex pairs possible in a general way: The construction so far was carried out in such a way that the connected components induced by previously covered vertex pairs are P_3s or C_8s.

In Lemma 1 we will indeed pack triangles instead of P_3s because this is more convenient in the proof. We will replace the triangles by P_3s afterwards: Recall the intuition from Section 3 that P_3s in the packing $\mathcal{H}$ which have exactly one endpoint in one clique T and their remaining two vertices in another clique R can accommodate both merging R and T or separating R and T without excess edits. Hence, we will replace the triangles by such P_3s. Recall that we aim for each clique to be a proto-cluster in the final construction, that is, each clique contains a spanning tree of edges which are not contained in P_3s in $\mathcal{H}$. Since putting the above kind of P_3s into the packing $\mathcal{H}$ allows in principle to delete edges within R, we need to ensure that R remains a proto-cluster. We achieve this via the connectedness property in Lemma 1.

Number-theoretic padding tool.

Lemma 1. *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 singleton, 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 the graph $(W, \binom{W}{2} \setminus E(\tau))$ is connected. Moreover, each vertex $v \in V \cup W$ is in at most p triangles of τ, it is in at most $p-1$ triangles if v is in a connected component of $(V \cup W, F)$ that is a P_3, and in at most $p-2$ triangles if v is in connected component of $(V \cup W, F)$ that is a C_8.*

Proof. First, we divide W into two parts W_1 and W_2 of equal sizes such that if two vertices $w, w' \in W$ are connected to the same vertex $v \in V$ by edges in F, then w and w' are in different parts. Note that this is easy for a connected component of $(V \cup W, F)$ if it is a P_3. For a connected component of $(V \cup W, F)$ which is a C_8, this is also doable as shown in Figure 5, where $w_i, w_{i+1}, w_{i+2}, w_{i+3}$ belong to W_1, $w'_i, w'_{i+1}, w'_{i+2}, w'_{i+3}$ belong to W_2, and $v_i, v_{i+1}, v_{i+2}, v_{i+3}$ belong to V.

We now label the vertices by elements from the finite field $\mathbb{F}_p$ of size p (recall that $\mathbb{F}_p$ consists of the elements $\{0, 1, \ldots, p-1\}$ with addition and multiplication modulo p). To each vertex $v \in V$, each vertex $w \in W_1$, and each vertex $w' \in W_2$, we will assign a unique label v_i, w_j, and w'_k, respectively, with $i, j, k \in \mathbb{F}_p$. In other words, we construct three bijections that map $\mathbb{F}_p$ to V, W_1, and W_2, respectively.

First, we label the vertices from the connected components of $(V \cup W, F)$ (and some singleton vertices) by going through the connected components one-by-one. For each yet-unlabeled connected component of $(V \cup W, F)$ that is a P_3 given by wvw' such that $v \in V, w \in W_1, w' \in W_2$, we label vertex w as w_j, vertex v as v_j and vertex w' as w'_j for the smallest j from $\mathbb{F}_p$ which is not yet used in the labeling of vertices of V. For each yet-unlabeled connected component C in $(V \cup W, F)$ that is a C_8 we proceed as follows. By the way we have divided vertices from W into W_1 and W_2, we can assign, to each such connected component C, four vertices which have degree zero in $(V \cup W, F)$: two in W_1 and two in W_2; see also Figure 5. We thus label the vertices in C and the four degree-zero vertices assigned to C as in Figure 5, for the smallest integer i from $\mathbb{F}_p$ such that $i, i+1, i+2$ and $i+3$ are not used in the labeling of vertices of V.

Second, we label the remaining unlabeled vertices that are not in the connected components of $(V \cup W, F)$. For an unlabeled vertex $w \in W_1$, label it as w_k for an arbitrary integer k from $\mathbb{F}_p$ which is not used in the labeling of vertices in W_1. Similarly, for an unlabeled vertex $v \in V$, we label it as v_h for an arbitrary integer h from $\mathbb{F}_p$ which is not used in the labeling of vertices in V and for an unlabeled vertex $w' \in W_2$, we label it as w'_s for an arbitrary integer s from $\mathbb{F}_p$ which is not used in the labeling of vertices in W_2. After the labeling, the vertices in V, W_1 and W_2 are $v_1, \ldots, v_{p-1}$, $w_1, \ldots, w_{p-1}$ and $w'_1, \ldots, w'_{p-1}$, respectively.

We now proceed to constructing the packing τ. First, let

$$\tau_{\mathsf{all}} = \left\{ uvw \;\middle|\; uvw \text{ is a triangle in } \left(V \cup W, E \cup \binom{W}{2}\right) \text{ such that } u \in V, v \in W_1, w \in W_2 \right\}, \text{ and}$$

$$\tau_{\mathsf{cover}} = \{v_i w_j w'_k \in \tau_{\mathsf{all}} \mid i, j, k \in \mathbb{F}_p \text{ and } j - i = k - j \text{ over } \mathbb{F}_p\}.$$

In the following, for any triangle packing τ, by $E(\tau)$ we will denote the union of the edge sets of the triangles in τ.

We claim that the triangles in τ_{cover} are edge-disjoint and cover all edges of E. Consider an arbitrary edge $v_i w_j \in E$ between V and W_1 for $i, j \in \mathbb{F}_p$. According to the definition of τ_{cover}, each triangle $v_i w_j w'_x \in \tau_{\mathsf{cover}}$ that covers edge $v_i w_j$ satisfies $x = 2j - i$ (over $\mathbb{F}_p$). Since $\mathbb{F}_p$ is a field, there is thus exactly one such triangle. Similarly, each edge $v_h w'_k \in E$ between V and W_1 for some $h, k \in \mathbb{F}_p$ is covered by the unique triangle $v_h w_{(h+k)\cdot 2^{-1}} w'_k \in \tau_{\mathsf{cover}}$. Finally, each edge $w_s w'_t$ between W_1 and W_2 is covered by the unique triangle $v_{2s-t} w_s w'_t \in \tau_{\mathsf{cover}}$. Thus the claim holds.

Let

$$\tau_F^1 = \{v_h w_h w'_h \in \tau_{\mathsf{all}} \mid \text{vertices } w_h, v_h, w'_h \text{ induce a } P_3 \text{ in } (V \cup W, F)\}, \text{ and}$$

$$\tau_F^2 = \{v_h w_{h+1} w'_{h+2},\ v_{h+1} w_{h+1} w'_{h+1},\ v_{h+2} w_{h+2} w'_{h+2},\ v_{h+3} w_{h+2} w'_{h+1} \in \tau_{\mathsf{all}} \mid \text{vertices } v_h, w'_{h+2}, v_{h+2}, w_{h+2}, v_{h+3}, w'_{h+1}, v_{h+1}, w_{h+1} \text{ induce a } C_8 \text{ in } (V \cup W, F)\}.$$

Observe that $\tau_F^1, \tau_F^2 \subseteq \tau_{\mathsf{cover}}$. For example, if we put $v_{h+3} w_{h+2} w'_{h+1} = v_i w_j w'_k$, then it follows that $j - i = p - 1 = k - j$ over $\mathbb{F}_p$, that is, $v_{h+3} w_{h+2} w'_{h+1}$ satisfies the conditions in the definition of τ_{cover}. Moreover, $\tau_F^1 \cup \tau_F^2$ covers all edges of F. Furthermore, each edge in the edge set $E(\tau_F^1 \cup \tau_F^2)$ of $\tau_F^1 \cup \tau_F^2$ is either in F or between W_1 and W_2. (See also Figure 5.) Thus, $E \setminus F$ has an empty intersection with $E(\tau_F^1 \cup \tau_F^2)$. Let $\tau = \tau_{\mathsf{cover}} \setminus (\tau_F^1 \cup \tau_F^2)$. It follows that τ covers all edges of $E \setminus F$. It remains only to show that τ satisfies the connectedness condition. Since τ_{cover} does not cover any edge of $\binom{W_1}{2}$ or $\binom{W_2}{2}$, it follows that $(W_1, \binom{W_1}{2} \setminus E(\tau))$ and $(W_2, \binom{W_2}{2} \setminus E(\tau))$ are cliques. Now observe that

$\tau_F^1 \cup \tau_F^2$ contains at most $|V| = p$ edges of $\binom{W}{2}$, while $W_1 \times W_2$ is of size $p^2 > p$. Thus in the graph $(W, \binom{W}{2} \setminus E(\tau))$ there is at least one edge $\{w_1, w_2\}$ such that $w_1 \in W_1$ and $w_2 \in W_2$. As a result, $(W, \binom{W}{2} \setminus E(\tau))$ is connected. Finally, observe that each vertex $v \in V \cup W$ is in at most p triangles in τ_{cover}. If v is in a P_3 of $(V \cup W, F)$, then at least one of these triangles is removed from τ_{cover} to obtain τ. If v is in a C_8 of $(V \cup W, F)$, then at least two of the triangles in τ_{cover} that contain v are removed to obtain τ. This concludes the proof. □

The following corollary is slightly easier to apply than Lemma 1.

Corollary 1. *Let p be a prime and let $B = (V, W, E)$ be a complete bipartite graph with $|V| \leq p$, $|W| = 2p$. Let $F \subseteq E$ be a nonempty set of edges such that every 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 $(W, \binom{W}{2} \setminus E(\tau))$ is connected. Each vertex $v \in V \cup W$ is in at most p triangles of τ, at most $p-1$ if v is in a connected component of $(V \cup W, F)$ that is a P_3, and at most $p-2$ if v is in connected component of $(V \cup W, F)$ that is a C_8.*

Proof. Add extra $p - |V|$ dummy vertices to V, obtaining a complete bipartite graph $B' = (V', W, E)$, apply Lemma 1 to B', p, and F, obtaining a packing τ', and return a sub-packing $\tau \subseteq \tau'$ containing only triangles with vertices in B. Since every triangle in τ' contains exactly one vertex of V', τ satisfies all the required properties. □

Concluding the construction. Equipped with Lemma 1 and Corollary 1, we can finish the construction of the clause gadgets and indeed the whole instance $(G, \mathcal{H}, 0)$ of CEAMP. We now specify the exact size of each clique introduced above and add padding P_3s to G and $\mathcal{H}$ so as to cover all vertex pairs between cliques that are adjacent in the merging model H. Put initially the set $\mathcal{H}_{\text{pad}}$ of *padding P_3s* to be $\mathcal{H}_{\text{pad}} = \emptyset$. We start with levels 0 and 1. We do not change the sizes of any clique on level 0. That is, as shown in the variable gadget, there are five vertices in every clique of level 0. Besides, we set the size of every clique of level 1 to be one. Note that no cliques of levels 0 and 1 are adjacent in the merging model H, that is, no two of them need to be merged in the solution. Hence, it is not necessary to add padding P_3s within these levels.

Now we turn each level i, $i \geq 2$, in order of increasing i. For each clique Q of level i, we apply Corollary 1 in the following scenario. Let V be the union of all cliques of levels $j < i$ that are out-neighbors of Q in the merging model H. Let p be the smallest prime with $p \geq |V|$ and $2p \geq |Q|$. Introduce $2p - |Q|$ new vertices, put them into Q, and make Q a clique. Put $W = Q$ and let $E = \{\{u, v\} \mid u \in V, v \in W\}$.

We claim that Corollary 1 is applicable to p, graph $B = (V, W, E)$, and F. To see this, we need to show that each connected component in $(V \cup W, F)$ is either a P_3 with center in V or a C_8. Indeed, if Q is not a transferring clique, that is, $Q = Q_d^j$ for some $d \in \{0, 1, \ldots, m-1\}$ and $j \in \{1, 2, 3, 4\}$, then each connected component in $(V \cup W, F)$ consists of two edges of two different transferring P_3s with the same center in V, as claimed (see also Figure 2). If Q is a transferring clique, then each connected component of $(V \cup W, F)$ consists either of two edges of two different transferring P_3s with the same center in some $Q_d^j \subseteq V$ for some $j \in \{1, 3, 4\}$, or of some vertex pairs of transferring P_3s between Q and the cliques of a variable gadget. In the first case, the claim clearly holds. In the second case, observe that the edges and non-edges between V and W in the transferring P_3s are each incident with one of w_1, w_2, w_3, w_4 and one of v_1, v_2, v_3, v_4 as defined when connecting variable and clause gadgets. These edges and non-edges indeed induce a C_8 given by $v_1 w_1 v_3 w_3 v_2 w_2 v_4 w_4 v_1$ (see also Figure 3). Thus, Corollary 1 is applicable.

Corollary 1 gives us an edge-disjoint triangle packing τ in $(V \cup W, (E \setminus F) \cup \binom{W}{2})$ which covers all edges of $E \setminus F$ such that $(W, \binom{W}{2} \setminus E(\tau))$ is connected. Note that every triangle $vw_1w_2 \in \tau$ has one vertex $v \in V$ and two vertices $w_1, w_2 \in W$. For every triangle $vw_1w_2 \in \tau$, we add a P_3 to G by using exactly two edges of the triangle in G; more precisely, we put $\{v, w_1\}, \{w_1, w_2\} \in E(G), vw_2 \notin E(G)$, and then add the P_3 of G given by vw_1w_2 into $\mathcal{H}_{\mathsf{pad}}$. Finally, let $\mathcal{H} = \mathcal{H}_{\mathsf{var}} \cup \mathcal{H}_{\mathsf{tra}} \cup \mathcal{H}_{\mathsf{pad}}$. Note that $\mathcal{H}$ is a modification-disjoint packing of P_3s: This is by construction for $\mathcal{H}_{\mathsf{var}} \cup \mathcal{H}_{\mathsf{tra}}$ and, by Corollary 1, no P_3 in $\mathcal{H}_{\mathsf{pad}}$ shares a vertex pair with any P_3 in $\mathcal{H}_{\mathsf{var}} \cup \mathcal{H}_{\mathsf{tra}}$. This concludes the construction of the CEAMP instance $(G, \mathcal{H}, 0)$.

To see that the construction takes polynomial time and to see that indeed each vertex is in some constant number of P_3s in $\mathcal{H}$, let us now derive the precise sizes of each clique in the construction. Recall that the cliques on level 0 are exactly those in the variable gadgets, and these have exactly 5 vertices each. The cliques on level 1 are Q^1_d and Q^4_d for $d \in \{0, 1, \ldots, m-1\}$, and they have 1 vertex each. On level 2 we have the cliques Q^3_d, $d \in \{0, 1, \ldots, m-1\}$, and since the only out-neighbor in H of Q^3_d is Q^4_d, our procedure sets $p = 2$ and thus Q^3_d has 4 vertices. On level 3 there are the cliques Q^2_d, $d \in \{0, 1, \ldots, m-1\}$, and we set $p = 7$ as $|Q^1_d \cup Q^3_d \cup Q^4_d| = 6$. Thus clique Q^2_d has 14 vertices. For the clique T^a_d, we set $p = 17$ as $|Q^1_d \cup K^a_{4\pi(a,d)} \cup K^a_{4\pi(a,d)+1} \cup K^a_{4\pi(a,d)+2}| = 16$. So the clique T^a_d has $2 \cdot 17 = 34$ vertices. Similarly, T^c_d has 34 vertices as well. For the clique T^b_d, we set $p = 23$, as $|Q^3_d \cup Q^4_d \cup K^b_{4\pi(b,d)} \cup K^b_{4\pi(b,d)+1} \cup K^b_{4\pi(b,d)+2}| = 20$. Thus T^b_d is a clique of size $2 \cdot 23 = 46$. By the bounds on the number of triangles in the packing, each vertex is in at most 23 P_3s of $\mathcal{H}$. It also follows that the construction takes overall polynomial time.

4.2 Correctness

We now prove the correctness of the reduction given in Section 4.1.

4.2.1 Completeness. Now we show how to translate a satisfying assignment of Φ into a cluster editing set of size $|\mathcal{H}|$ for the constructed instance.

LEMMA 2. *If the input formula Φ is satisfiable, then the constructed instance $(G, \mathcal{H}, \ell = 0)$ is a YES-instance.*

PROOF. Assume that there is a satisfying assignment α for the formula Φ. Recall that n is the number of variables of Φ and m is the number of clauses of Φ. Instead of building the solution directly, we build a partition $\mathcal{P}$ of $V(G)$ into clusters. Then, we argue that the number of edges between clusters and the number of non-edges inside clusters is at most $|\mathcal{H}|$. Thus, the partition $\mathcal{P}$ will induce a solution with the required number of edge edits.

Recall that H denotes the merging model of our hardness construction. The basic building blocks of our vertex partition $\mathcal{P}$ are the cliques in G that correspond to the vertices of $V(H)$. We will never separate such a clique during building $\mathcal{P}$, that is, $\mathcal{P}$ corresponds to a partition of $V(H)$. For simplicity, we will slightly abuse notation and indeed also treat $\mathcal{P}$ as a partition of $V(H)$. We build $\mathcal{P}$ by taking initially $\mathcal{P} = V(H)$ and then successively *merging* parts of $\mathcal{P}$, which means to take the parts out of $\mathcal{P}$ and replace them by their union. Each vertex of H is a clique of G, so has no non-edges in G. Thus, below it suffices to consider edges and non-edges between pairs of cliques corresponding to vertices in $V(H)$ to determine the number of edits in the solution corresponding to $\mathcal{P}$.

We start with the variable gadgets. Consider each variable x_i, $i = 0, 1, \ldots, n-1$. Call a pair of cliques K^i_j, K^i_{j+1} in x_i's variable gadget *even* if j is even and *odd* otherwise (indices are taken modulo $4m_i$). If $\alpha(x_i) = \text{true}$, then merge each odd pair. If $\alpha(x_i) = \text{false}$, then merge each even pair. We will not merge any further pair of cliques contained in variable gadgets.

Now consider each clause Γ_d, $d = 0, \ldots, m-1$, in some arbitrary order. Let x_a, x_b, and x_c be the variables in Γ_d. We use the same notation as when defining the clause gadgets. See Figure 2 for the skeleton of the clause gadget of Γ_d, up to variables appearing positively instead of negatively or vice versa. We choose an arbitrary variable that satisfies Γ_d. The basic idea is to separate (that is, to not merge) the transferring clique from the the cliques in the satisfying variable's gadget by deleting some edges of the transferring P_3s. This will induce at most one edit for each transferring P_3 since the remaining edge in a transferring P_3 will be part of a cluster in $\mathcal{P}$. Then we cut from the clause gadget all transferring cliques belonging to variables that have not been chosen. Since we do not spend edits inside of transferring P_3s in this way, this allows us to merge the transferring cliques to the variable gadgets regardless of whether the variable was set to true or false.

Formally, we perform the following merges in $\mathcal{P}$.

If we have chosen x_a from the variables satisfying the clause Γ_d:
- Merge T_d^a with Q_d^1.
- Merge the cliques Q_d^2, Q_d^3 and Q_d^4.

If we have chosen x_b:
- Merge the cliques Q_d^1, Q_d^2.
- Merge the cliques T_d^b, Q_d^3, and Q_d^4.

If we have chosen x_c:
- Merge T_d^c with Q_d^4.
- Merge the cliques Q_d^1, Q_d^2 and Q_d^3.

Finally, let $\beta \in \{a, b, c\}$ be the index of the chosen variable that satisfies Γ_d. For both $\gamma \in \{a, b, c\} \setminus \{\beta\}$ do the following. If $\alpha(x_\gamma) = \text{true}$, then merge T_d^γ with the part of $\mathcal{P}$ consisting of $K^\gamma_{4\pi(\gamma,d)+1}$ and $K^\gamma_{4\pi(\gamma,d)+2}$. If $\alpha(x_\gamma) = \text{false}$, then merge T_d^γ with the part of $\mathcal{P}$ consisting of $K^\gamma_{4\pi(\gamma,d)+1}$ and $K^\gamma_{4\pi(\gamma,d)}$. This concludes the definition of the vertex partition $\mathcal{P}$. Let us denote the corresponding cluster editing set by S. That is, S contains all edges in G between parts of $\mathcal{P}$ and all non-edges within parts of $\mathcal{P}$.

We claim that (c1) each edit in S is contained in a P_3 of $\mathcal{H}$ and (c2) every P_3 of $\mathcal{H}$ is edited at most once by S. Note that the claim implies that S is a solution to $(G, \mathcal{H}, 0)$. We first prove part (c1) of the claim. Note that each edit in S is between two cliques in $V(H)$. There are three types of edits in $\mathcal{H}$: within a variable gadget, between a clause and a variable gadget, and within a clause gadget.

Consider first the edits contained in the variable gadget of an arbitrary variable x_i. Observe that each such edit is contained in an odd or an even pair of x's gadget. Such an edit is contained in a P_3 in $\mathcal{H}$, because, by construction of the variable gadgets, all edges and non-edges between the cliques of an odd or an even pair are covered by P_3s in $\mathcal{H}$.

For the edits in S which are not contained in variable gadgets, observe that between each pair of cliques in a single level L_s, $s > 0$, there are no edges in G. Whenever we merge two or more parts during the construction of $\mathcal{P}$, we either merge a clique on level L_4 to two cliques on level L_0 or we merge cliques on pairwise different positive levels. Hence, each edit $e \in S$ which is not in a variable gadget is between two cliques on different levels. Moreover, observe that the cliques containing the endpoints of e are adjacent in $V(H)$. Thus, by the way we have defined $\mathcal{H}_{\text{pad}}$ via Corollary 1, there is a P_3 in $\mathcal{H}_{\text{pad}}$ containing e. We have thus shown that claim (c1) holds.

For part (c2) of the claim, we first observe the following. Each P_3 in $\mathcal{H}$ that intersects only two cliques in $V(H)$ contains at most one edit of S. Let P be such a P_3 and let D_1, D_2 be the two cliques in $V(H)$ that intersect P. Note that $\mathcal{H}_{\text{tra}}$ does not contain P_3s that intersect only two cliques in $V(H)$ and thus either $P \in \mathcal{H}_{\text{var}}$ or $P \in \mathcal{H}_{\text{pad}}$. In both cases, there is exactly one edge and one

non-edge of P between D_1 and D_2: This is clear if $P \in \mathcal{H}_{\text{pad}}$. If $P \in \mathcal{H}_{\text{var}}$ then P was introduced when connecting a clause gadget to a variable gadget. In the notation used there, either $P = v_5v_6v_2$ or $P = v_1v_7v_8$, both of which have the required form. Thus, as D_1 and D_2 are either merged or not in $\mathcal{P}$, there is at most one edit in P.

To prove (c2) it remains to consider P_3s in $\mathcal{H}$ that intersect three cliques in $V(H)$. Let P be such a P_3. Note that $P \notin \mathcal{H}_{\text{pad}}$. If $P \in \mathcal{H}_{\text{var}}$, then it connects K^i_j to K^i_{j+2} via K^i_{j+1} for some even j and some variable index $i \in \{0, 1, \ldots, n-1\}$. Since we merge either all odd or all even pairs in x_i's variable gadget to obtain $\mathcal{P}$, indeed exactly one edge of P is edited, as claimed. If $P \in \mathcal{H}_{\text{tra}}$, then we distinguish two cases.

First, P does not contain a vertex of some variable-gadget clique. Then, P connects some clique Q^s_d to some transferring clique T^δ_d via $Q^{s'}_d$. According to the construction of $\mathcal{P}$, either T^δ_d and $Q^{s'}_d$ are in different parts of $\mathcal{P}$ and $Q^{s'}_d$ and Q^s_d are merged, or T^δ_d and $Q^{s'}_d$ are merged and Q^s_d and $Q^{s'}_d$ are in different parts of $\mathcal{P}$. In both cases, there is at most one edit of S in P.

Second, P contains a vertex of some variable-gadget clique. Then, by construction of G and $\mathcal{H}$, path P indeed contains two vertices of two variable-gadget cliques, say K^i_j and K^i_{j+1} and one vertex of a transferring clique, say T^i_d. Assume that variable x_i appears positively in clause Γ_d, the other case is analogous. Then the center of P is K^i_j and moreover j is odd. If x_i was not chosen among the variables satisfying clause Γ_d when constructing $\mathcal{P}$, then T^i_d and K^i_j is in the same part Q of $\mathcal{P}$. Furthermore K^i_{j+1} is either in a part different from Q or also in Q. In both cases, there is at most one edit from S in P. If x_i was chosen among the the variables satisfying clause Γ_d when constructing $\mathcal{P}$, then T^i_d is in a part in $\mathcal{P}$ which is different from the one(s) containing K^i_j and K^i_{j+1}. However, since x_i satisfies Γ_d, we have $\alpha(x_i) = \text{true}$ and thus K^i_j and K^i_{j+1} are merged (recall that j is odd).

Thus, indeed, the claim holds, that is, each edit in S is contained in a P_3 in $\mathcal{H}$ and every P_3 of $\mathcal{H}$ is edited at most once by S. □

4.2.2 Soundness. Before we show how to translate a cluster editing set of size $|\mathcal{H}|$ for the constructed instance into a satisfying assignment of Φ, we make some structural observations.

Recall the definition of a proto-cluster, a connected component of the subgraph of G whose edge set contains precisely those edges of G which are not contained in any P_3 in $\mathcal{H}$.

Lemma 3. *$V(H)$ is precisely the set of proto-clusters of G with respect to $\mathcal{H}$.*

Proof. By construction, all edges in G between two cliques in $V(H)$ are in a P_3 in $\mathcal{H}$. Thus each proto-cluster is contained in some clique in $V(H)$. We claim that each clique $C \in V(H)$ contains a spanning tree of edges which are not contained in a P_3 in $\mathcal{H}$. If $C \in L_1$, then this is clear; such a C contains only a single vertex and a trivial spanning tree. If $C \in L_0$, then there are only two P_3s in $\mathcal{H}$ that contain edges of C: The one given by $v_5v_6v_2$ and the one given by $v_1v_7v_8$ as defined in Section 4.1.2 when connecting variable and clause gadgets. Since $|C| = 5$, indeed C contains the required spanning tree. If $C \in L_i$ for $i \geq 2$, then by the connectedness property of Corollary 1, C has the required spanning tree. □

Recall that each solution S to $(G, \mathcal{H}, 0)$ cannot remove any edge from G which is not contained in a P_3 in $\mathcal{H}$. Thus, since $V(H)$ is a vertex partition of G, each solution S generates a cluster graph $G \triangle S$ whose clusters induce a coarser vertex partition than $V(H)$. This leads to the following.

Observation 1. *For each solution S to $(G, \mathcal{H}, 0)$, each cluster in $G \triangle S$ is a disjoint union of cliques in $V(H)$.*

Using the above structural observations, we are now ready to prove the soundness of the construction.

Lemma 4. *If the constructed instance $(G, \mathcal{H}, \ell = 0)$ is a YES-instance, then the formula Φ is satisfiable.*

Proof. Suppose that there exists a set of vertex pairs $S \subseteq \binom{V}{2}$ so that $G\Delta S$ is a union of vertex-disjoint cliques and $|S| - |\mathcal{H}| = 0$. In other words, there exists a solution that transforms G into a cluster graph G' by editing exactly one edge or non-edge of every P_3 of $\mathcal{H}$. We will construct a satisfying assignment $\alpha\colon \{x_0, x_1, \ldots, x_{n-1}\} \to \{\text{true}, \text{false}\}$ for the formula Φ.

By Observation 1, the set of clusters in G' induces a partition of the cliques in $V(H)$. Recall that we say that two cliques in $V(H)$ are *merged* if they are in the same cluster in G' and *separated* otherwise.

To define α, we need the following observation on the solution. Consider variable x_i and the cliques K^i_j, $j = 0, 1, \ldots, 4m_i - 1$, in x_i's variable gadget. Call a pair K^i_j, K^i_{j+1} *even* if j is even (where $j+1$ is taken modulo $4m_i$) and call this pair *odd* otherwise. We claim that either (i) each even pair is merged and each odd pair is separated, or (ii) each odd pair is merged and each even pair is separated (and not both). Note that, for each even j, pair K^i_j, K^i_{j+1} is merged or pair K^i_{j+1}, K^i_{j+2} is merged, because there is a P_3 in G containing vertices in these cliques with center in K^i_{j+1}. To show the claim, it is thus enough to show that not both an odd pair and an even pair is merged.

For the sake of contradiction, suppose that an odd pair is merged and an even pair is merged. Then, there exists an index $j \in \{0, 1, \ldots, 4m_i - 1\}$ and a cluster C in G' such that $K^i_j, K^i_{j+1}, K^i_{j+2} \subseteq C$, where here and below the indices are taken modulo $4m_i$. Observe that there are no edges between K^i_j and K^i_{j+2} in G. If j is odd, then all of these non-edges are non-packed. All of these non-edges are thus in S. This is a contradiction to the fact that S contains at most $|\mathcal{H}|$ vertex pairs. Thus, j is even.

We now show that for each $k \in \mathbb{N} \cup \{0\}$, pair $K^i_{j+1+2k}, K^i_{j+2+2k}$ is merged by induction on k. Clearly, for $k = 0$, this holds by supposition. If $k > 0$ then, by the construction of $\mathcal{H}_{\text{var}}$, there are non-packed non-edges between K^i_{j+2k-1} and K^i_{j+2k+1}. Combining this with the fact that $K^i_{j+1+2(k-1)} = K^i_{j+2k-1}$ and $K^i_{j+2+2(k-1)} = K^i_{j+2k}$ are merged by inductive assumption, it follows that K^i_{j+2k} and K^i_{j+2k+1} are separated. Since there is a P_3 in G connecting K^i_{j+2k}, K^i_{j+2k+1}, and K^i_{j+2k+2} with center in K^i_{j+2k+1} and S contains at most one edit in this P_3, it follows that K^i_{j+2k+1}, K^i_{j+2k+2} are merged, as required.

It now follows in particular that K^i_{j-1} and K^i_j are merged (recall that indices are taken modulo $4m_i$). Since by assumption also K^i_j and K^i_{j+1} are merged, we have that K^i_j, $K^i_{j'+1}$, and $K^i_{j'+2}$ are contained in the same cluster in G' for some odd j'. As already argued, this leads to a contradiction. Thus the claim holds.

We define the assignment α as follows. For each variable x_i, $i = 0, 1, \ldots, n-1$, if in G' all even pairs K^i_{2j}, K^i_{2j+1}, $j = 0, 1 \ldots, m_i - 1$, are merged, then $\alpha(x_i) = \text{false}$. Otherwise $\alpha(x_i) = \text{true}$.

We now show that α satisfies Φ. Consider an arbitrary clause Γ_d of Φ containing the three variables x_a, x_b, and x_c. We use the same notation as when defining the clause gadget and its connection to the variable gadget. Since there are non-packed non-edges between cliques Q^1_d and Q^4_d, cliques Q^1_d and Q^4_d must end up in different clusters in G'. In other words, Q^1_d and Q^4_d are separated. Observe that there is a path in G consisting of vertices in Q^1_d, Q^2_d, Q^3_d, and Q^4_d in this sequence. Since each of these four cliques is a proto-cluster (Lemma 3), in order to separate Q^1_d and Q^4_d, one of the following three cases must happen in the solution S: (i) The edges between Q^1_d and Q^2_d are deleted. In other words, Q^1_d and Q^2_d are separated. (ii) Q^2_d and Q^3_d are separated. (iii) Q^3_d and Q^4_d are separated. We now show that case (i), (ii), and (iii) imply that variable x_a, x_b, and x_c, respectively, is set by α so as to satisfy Γ_d. We only give the proof showing that case (i) implies that x_a is set accordingly. The other cases are analogous.

Assume that case (i) holds. Then, by the constraints imposed by the two transferring P_3s P_d^1 and P_d^2, cliques T_d^a and Q_d^1 are merged. Since there are non-packed non-edges between $K^a_{4\pi(a,d)+1}$ and Q_d^1, it follows that $K^a_{4\pi(a,d)+1}$ and Q_d^1 are separated. Consider the case that x_a appears positively in Γ_d. Then, when connecting the variable gadget of x_a to the clause gadget of Γ_d we have introduced into G a P_3 connecting T_d^a, $K^a_{4\pi(a,d)+1}$, and $K^a_{4\pi(a,d)+2}$ with center in $K^a_{4\pi(a,d)+1}$ (for example, the P_3 given by $w_1v_1v_3$). Since T_d^a and $K^a_{4\pi(a,d)+1}$ are separated, thus $K^a_{4\pi(a,d)+1}$ and $K^a_{4\pi(a,d)+2}$ are merged. There is thus at least one odd pair in x_a's variable gadget that is merged and thus $\alpha(x_a) = \text{true}$. The case where x_a appears negatively in Γ_d is similar: We have introduced into G a P_3 connecting T_d^a, $K^a_{4\pi(a,d)+1}$, and $K^a_{4\pi(a,d)}$ with center in $K^a_{4\pi(a,d)+1}$ (for example, the P_3 given by $w_1v_1v_3$). It follows that $K^a_{4\pi(a,d)+1}$, and $K^a_{4\pi(a,d)}$ are merged, showing that at least one even pair is merged in x_a's variable gadget. Thus, $\alpha(x_a) = \text{false}$.

Thus each clause Γ_d is satisfied, finishing the proof. □

5 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 $\mathcal{H}$. More precisely, we consider the following variant of CEaMP.

Cluster Editing above Half-Integral Modification-Disjoint P_3 Packing (CEaHMP)

Input: A graph $G = (V, E)$, a modification-disjoint packing $\mathcal{H}$ of induced P_3s of G such that every vertex $v \in V(G)$ is incident with at most two P_3s of $\mathcal{H}$, 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 give a polynomial-time algorithm to solve CEaHMP when ℓ is a fixed constant, in contrast with the NP-hardness of the general version of CEaMP when $\ell = 0$.

Theorem 2 (Restated). *Cluster Editing above Half-Integral Modification-Disjoint P_3 Packing parameterized by the number ℓ of excess edits is in XP. It can be solved in $n^{2\ell+O(1)}$ 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 3. *Cluster Editing above Half-Integral Modification-Disjoint P_3 Packing can be solved in polynomial time when $\ell = 0$, that is, when no excess edits are allowed.*

The proof of Theorem 3 will be given in Section 5.1. With this tool in hand, we can show Theorem 2.

Proof of Theorem 2. Let $(G, \mathcal{H}, \ell)$ be an instance of CEaHMP. The algorithm is given in Algorithm 1. Essentially, it 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 (Lines 1–4). Then it guesses the P_3s in $\mathcal{H}$ that harbor the remaining excess edits and it guesses how these P_3s are resolved (Lines 5–9). 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 3.

For the running time, observe that there are at most $n^{2\ell_a}$ choices for S_a. Since each vertex is in at most two P_3s in $\mathcal{H}$ and each P_3 covers exactly three vertices, we have $3|\mathcal{H}| \leq 2n$ and thus there are in total at most n P_3s in $\mathcal{H}$. Thus, there are $O(n^{\ell_b})$ choices for $\mathcal{H}_b$. Since there are four possibilities to select a set of at least two vertex-pairs in the vertex set of a P_3, there are $O(4^{\ell_b})$ possibilities for S_b in Line 6. Hence, overall the running time is $O(4^{\ell_b} n^{2\ell_a+\ell_b+O(1)}) \leq n^{2\ell+O(1)}$.

ALGORITHM 1: Solve CEaHMP.

Input: An instance $(G, \mathcal{H}, \ell)$ of CEaHMP.
Output: Whether $(G, \mathcal{H}, \ell)$ is a YES-instance.

1 **foreach** $\ell_a = 0, 1, \ldots, \ell$ **do**
2 **foreach** $\ell_b = 0, 1, \ldots, \ell - \ell_a$ **do**
3 **foreach** *set* S_a *of* ℓ_a *vertex pairs* $\{u, v\} \in \binom{V(G)}{2}$ *such that* $\forall P \in \mathcal{H} : |\{u, v\} \cap V(P)| \leq 1$ **do**
4 $G_a \leftarrow G \triangle S_A$
5 **foreach** *set* $\mathcal{H}_b$ *of* ℓ_b *distinct* P_3*s in* $\mathcal{H}$ **do**
6 **foreach** *set* S_b *containing for each* $P \in \mathcal{H}_b$ *at least two vertex pairs in* $V(P)$ **do**
7 **if** $|S_a| + |S_b| \leq |\mathcal{H}_b| + \ell$ **then**
8 $G_b \leftarrow G_a \triangle S_B$
9 $\mathcal{H}' \leftarrow \mathcal{H} \setminus \mathcal{H}_b$
10 **if** G_b *has a cluster-editing set with* $|\mathcal{H}'|$ *edits* **then** `/* Using Theorem 3 */`
11 accept and halt
12 reject

It remains to prove the correctness. If the algorithm accepts, then there is a cluster-editing set S_0 for G_b with $|\mathcal{H}'|$ edits. Since S_0 is contained in the vertex sets of the P_3s in $\mathcal{H}'$, set S_0 is disjoint from S_a and S_b. Thus, $G \triangle S^\star$ is a cluster graph where $S^\star = S_a \cup S_b \cup S_0$. Moreover, $|S^\star| \leq |\mathcal{H}'| + |\mathcal{H}_b| + \ell = |\mathcal{H}| + \ell$, and thus, $(G, \mathcal{H}, \ell)$ is a YES-instance.

Conversely, if $(G, \mathcal{H}, \ell)$ is a YES-instance, then there is a cluster-editing set $S^\star$ of G with $|S^\star| \leq |\mathcal{H}| + \ell$. Let $S_a^\star$ be the subset of $S^\star$ that contains precisely those edits in $S^\star$ that are not contained in P_3s of $\mathcal{H}$. In one of the iterations of Algorithm 1, $\ell_a = |S_a^\star|$ and $S_a = S_a^\star$. Now let $\mathcal{H}_b^\star$ be the subset of $\mathcal{H}$ that contains precisely those P_3s P such that $S^\star$ contains at least two edits in $V(P)$. Observe that $|\mathcal{H}_b^\star| \leq \ell - \ell_a$. Thus, in one of the iterations of Algorithm 1, we have $\ell_b = |\mathcal{H}_b^\star|$ and $\mathcal{H}_b = \mathcal{H}_b^\star$. Moreover, in one of the iterations $S_b = S_b^\star$, where $S_b^\star$ is the subset of $S^\star$ that contains precisely those edits that are contained in the P_3s in $\mathcal{H}_b$. Let $S_0^\star = S^\star \setminus (S_a^\star \cup S_b^\star)$. Since each edit in $S_0^\star$ is contained in a unique P_3 in $\mathcal{H} \setminus \mathcal{H}_b^\star$, we have $|S_a| + |S_b| = |S_a^\star| + |S_b^\star| \leq |\mathcal{H}_b^\star| + \ell = |\mathcal{H}_b| + \ell$. Thus, in that iteration the algorithm proceeds to the if-condition in Line 10. Again since each edit in $S_0^\star$ is contained in a unique P_3 in $\mathcal{H} \setminus \mathcal{H}_b^\star$, this set witnesses that $(G_b, \mathcal{H}', 0)$ is a YES-instance and thus the algorithm accepts. Hence, the algorithm is correct. □

5.1 Polynomial-time Algorithm for Zero Excess Edits

Let Cluster Editing Matching Half-Integral Modification-Disjoint P_3 Packing (CEMHMP) be the special case of CEaHMP where $\ell = 0$. That is, an instance of CEMHMP is given by a tuple $(G, \mathcal{H})$ of a graph G and a half-integral P_3 packing $\mathcal{H}$ in G. In this section we give a polynomial-time algorithm for CEMHMP. Again, we use the term *proto-clusters* to denote the connected components of the graph obtained by removing the edges of all packed P_3s.

The intuition behind the polynomial-time result is that, with the constraint that every vertex $v \in V(G)$ is incident with at most two packed P_3s, we cannot freely merge or separate two large proto-clusters without excess edits as in the NP-hardness proof of Section 4. This is because the triangles formed by the packed P_3s cannot cover every vertex pair between two large proto-clusters. Thus we can separate the large proto-clusters and deal with them separately.

The polynomial-time algorithm mainly proceeds by applying reduction rules that simplify the instance step by step. Herein, our first goal is to eliminate proto-clusters of size at least four, which

can be done by a series of straightforward reduction rules (Section 5.1.1). We then look at proto-clusters of size three and observe that their connections to the rest of the graph have quite a limited structure. This observation can be used to eliminate proto-clusters of size three as well (Section 5.1.2). The reduction rules we have developed at this point give more structural observations on smaller proto-clusters which can be used to show that the size of solution clusters is at most four (Section 5.1.3). Afterwards, we show that the only situation in which solution clusters of size four can occur is when there is a certain path-like structure in the instance. A final, quite involved reduction rule takes care of such path-like structures (Section 5.1.4). This then results in an instance with a solution whose clusters have size at most three. Using this cluster-size bound we can finally show that, if there is a solution, then there is also one that only deletes edges. This then leads to a formulation as an instance of 2-SAT (Section 5.1.5), which is well-known to be polynomial-time solvable.

We use the following notation. We say a proto-cluster C is *isolated* from a proto-cluster D if there are no edges of G 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 respectively, then xyz is a *type-γ* P_3; and
- if x, y, z belong to one proto-cluster then xyz is a *type-δ* P_3.

As mentioned, in the following, we present a series of reduction rules, which are algorithms that take an instance of CEMHMP and produce a new instance of CEMHMP. By saying that a reduction rule is *safe*, we mean that the instance before applying this reduction rule is a YES-instance if and only if the instance after applying this reduction rule is a YES-instance. Since the P_3s of $\mathcal{H}$ are modification-disjoint, we have the following handy observation.

Observation 2. *A solution S to an instance of CEMHMP must edit exactly one edge or non-edge of every P_3 of $\mathcal{H}$, and neither non-packed edges nor non-packed non-edges can be edited by S.*

5.1.1 Simple Reduction Rules. We start by getting rid of several simple situations.

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.*

Lemma 5. *Reduction Rule 1 is safe.*

Proof. Given an instance $(G, \mathcal{H})$ of CEMHMP satisfying the condition of Reduction Rule 1, suppose for contradiction that there is a solution S to this instance. Since u, v belong to the same proto-cluster, there is a non-packed path P from u to v. By Observation 2, $uv \notin S$ and none of the edges of P is edited by S. Thus $G \triangle S$ is not a cluster graph, contradicting that the instance has a solution. This completes the proof for the lemma. □

The second reduction rule handles type-β and type-δ P_3s (see Figure 6).

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

Lemma 6. *Reduction Rule 2 is safe.*

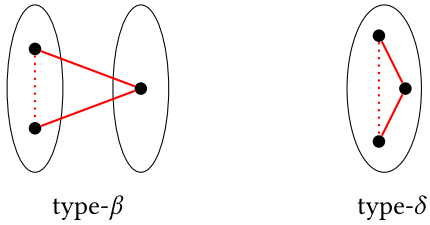

Fig. 6. Examples for Reduction Rule 2.

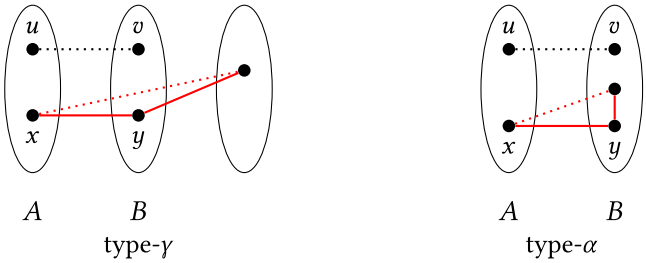

Fig. 7. Examples for Reduction Rule 3.

Proof. Suppose that the given instance of CEMHMP is $(G, \mathcal{H})$ such that there exists a type-β P_3 xyz in G. After inserting the edge xz and removing xyz from $\mathcal{H}$, we get an instance $(G', \mathcal{H}')$. We claim that $(G, \mathcal{H})$ is a YES-instance if and only if $(G', \mathcal{H}')$ is a YES-instance. On one hand, suppose that $(G', \mathcal{H}')$ is a YES-instance and S' is a cluster editing set of G' such that $|S'| = |\mathcal{H}'|$. Obviously, $S' \cup \{xz\}$ is a cluster editing set for G and $|S' \cup \{xz\}| = |\mathcal{H}|$. On the other hand, suppose that $(G, \mathcal{H})$ is a YES-instance and S is a cluster editing set of G such that $|S| = |\mathcal{H}|$. We show that $xz \in S$ and $S \setminus \{xz\}$ is the solution for $(G', \mathcal{H}')$. For contradiction, suppose this is not true. Then either $xy \in S$ or $yz \in S$ holds. Without loss of generality we assume that $xy \in S$. Suppose that after deleting xy from G and removing xyz from $\mathcal{H}$, we get an instance $(G'', \mathcal{H}'')$. Since x, z belong to one proto-cluster of G, there is a non-packed path P from x to z in G. Thus x, z belong to one proto-cluster of G''. Since xyz is removed from $\mathcal{H}$, xz becomes a non-packed non-edge. By Reduction Rule 1, $(G'', \mathcal{H}'')$ is a NO-instance, contradicting that S is a solution to $(G, \mathcal{H})$.

A similar analysis applies to the case that $xyz \in \mathcal{H}$ is a type-δ P_3. This completes the proof for the lemma. □

After applying Reduction Rules 1 and 2 exhaustively, if the algorithm did not return NO, then there is no type-β or type-δ P_3s in the instance. The next reduction rule applies to the case in which there is both a non-packed non-edge and a packed edge between two proto-clusters, see Figure 7 for an illustration.

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 $y \in V(B)$ (not necessarily distinct from u or v), then delete xy and remove the corresponding packed P_3 from $\mathcal{H}$.*

Lemma 7. *Reduction Rule 3 is safe.*

Proof. Given an instance $(G, \mathcal{H})$ of CEMHMP satisfying the condition of Reduction Rule 3 with xy covered by a type-γ P_3 xyz. Without loss of generality, we do not analyze the symmetrical case where x is the center vertex of the P_3 instead of y. We get an instance $(G', \mathcal{H}')$ of CEMHMP

after deleting xy and removing xyz from $\mathcal{H}$. We claim that $(G, \mathcal{H})$ is a YES-instance if and only if $(G', \mathcal{H}')$ is a YES-instance. For the soundness, assume that $(G', \mathcal{H}')$ is a YES-instance and S' is a cluster editing set of size $|\mathcal{H}'|$ for G'. Then obviously $S' \cup \{xy\}$ is a solution to $(G, \mathcal{H})$. For the completeness, assume that $(G, \mathcal{H})$ is a YES-instance and S is a cluster editing set of size $|\mathcal{H}|$ for G. We claim that $xy \in S$. Suppose for contradiction that $xy \notin S$. Then xy becomes a non-packed edge in $G \triangle S$. Since $u, x \in V(A)$ and $v, y \in V(B)$, there is a non-packed path P_A from u to x and a non-packed path P_B from v to y in G. By Observation 2, the edges of P_A and P_B are not edited by S and $uv \notin S$. Thus there is a non-packed path from u to v. Since uv is a non-packed non-edge in $G \triangle S$, $G \triangle S$ is not a cluster graph, contradicting the assumption that S is a solution to $(G, \mathcal{H})$.

A similar analysis applies to the case in which xy is covered by a type-α P_3 xyz (and its symmetrical case where x is the center vertex instead of y). This concludes the proof for the lemma. □

The next reduction rule deals with isolated cliques in graph G.

Reduction Rule 4. *If there is a proto-cluster C which is an isolated clique of G, then remove C from the graph.*

Lemma 8. *Reduction Rule 4 is safe.*

Proof. Given an instance $(G, \mathcal{H})$ of CEMHMP such that there is a proto-cluster C which is an isolated clique, we remove C from G and get an instance $(G', \mathcal{H})$. We claim that $(G, \mathcal{H})$ is a YES-instance if and only if $(G', \mathcal{H})$ is a YES-instance. On one hand, assume that $(G', \mathcal{H})$ is a YES-instance. Then obviously $(G, \mathcal{H})$ is a YES-instance. On the other hand, assume that $(G, \mathcal{H})$ is a YES-instance and S is a solution. Since C is an isolated clique, by Observation 2, neither edges of C nor non-edges between $V(C)$ and $V(G) \setminus V(C)$ are edited by S. Thus S is also a solution to $(G', \mathcal{H})$. This completes the proof for the lemma. □

In later analysis, we will see that some constant-size configurations cannot be connected to the rest of the graph. To remove such configurations, we introduce the following reduction rule.

Reduction Rule 5. *If there is a connected component C in G of size at most 6, then do brute force on C to check if there is a cluster editing set F for 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 C and remove the corresponding packed P_3s from $\mathcal{H}$. Otherwise, if there is no such cluster editing set F, return NO.*

Lemma 9. *Reduction Rule 5 is safe.*

Proof. Given an instance $(G, \mathcal{H})$ of CEMHMP such that there is a connected component C in the graph of size at most 6, suppose that there is a cluster editing set F for C satisfying the condition of Reduction Rule 5. After performing the operations of F, we get an instance $(G', \mathcal{H}')$ of CEMHMP. We claim that $(G, \mathcal{H})$ is a YES-instance if and only if $(G', \mathcal{H}')$ is a YES-instance. On one hand, assume that $(G', \mathcal{H}')$ has a solution S'. Obviously, $S' \cup F$ is a cluster editing set for G and $|S' \cup F| = |\mathcal{H}|$. On the other hand, assume that $(G, \mathcal{H})$ has a solution S. By Observation 2, no vertex pair between $V(C)$ and $V(G) \setminus V(C)$ is edited by S. Let $S_1 \subseteq S$ be the set of vertex pairs which are edges or non-edges of C. Then $S \setminus S_1$ is a solution to $(G', \mathcal{H}')$.

Suppose that there is no such cluster editing set F for C. We claim that $(G, \mathcal{H})$ is a NO-instance. For contradiction, assume that $(G, \mathcal{H})$ has a solution S. Let $S_1 \subseteq S$ be the set of vertex pairs which are edges or non-edges of C. Then S_1 is a cluster editing set for C and $|S_1|$ is equal to the number of packed P_3s incident with a vertex of C by Observation 2, a contradiction. Thus $(G, \mathcal{H})$ is a NO-instance.

The component C is of size at most 6 so we can do brute force in constant time. This completes the proof for the lemma. □

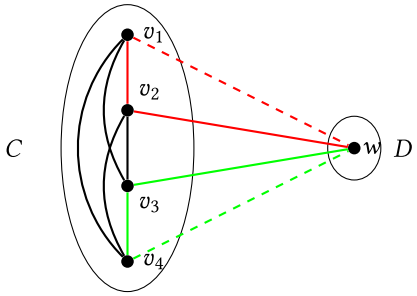

Fig. 8. An example for Lemma 11.

We now move to analyzing the size of the remaining proto-clusters.

Lemma 10. *After applying Reduction Rules 1 to 4 exhaustively, if the algorithm did not return NO, then there is no proto-cluster of size at least* 5.

Proof. Suppose for contradiction that there is a proto-cluster C of size at least 5. If C is a proto-cluster which is isolated from other proto-clusters, then C must be a clique since otherwise Reduction Rule 1 or Reduction Rule 2 can be applied, a contradiction. Then Reduction Rule 4 can be applied and C will be removed from the graph. Thus C is not an isolated proto-cluster.

Let D be a proto-cluster such that there is an edge uv between C and D, say $u \in V(C)$ and $v \in V(D)$. If uv is covered by a type-β P_3, then Reduction Rule 2 can be applied, a contradiction. Thus we assume that uv is covered by a type-α or a type-γ P_3. Since v is incident with at most two packed P_3s, there must be one vertex $w \in V(C)$ such that wv is a non-packed non-edge. Then Reduction Rule 3 can be applied, a contradiction. As a result, there is no proto-cluster of size at least 5. This completes the proof for the lemma. □

Next we focus on proto-clusters of size 4.

Lemma 11. *After applying Reduction Rules 1 to 3 exhaustively, if there is a proto-cluster C of size* 4 *which is not an isolated clique of G, then there is a proto-cluster D of size* 1 *such that the vertex pairs between C and D are covered by two type-α P_3s. In addition, $V(C) \cup V(D)$ forms a connected component in the graph.*

Proof. After applying Reduction Rules 1 to 3 exhaustively, let C be a proto-cluster of size 4 and $V(C) = \{v_1, v_2, v_3, v_4\}$. See Figure 8 for an illustration. Let w be a vertex such that there is an edge between w and $V(C)$. If the vertex pairs between $V(C)$ and w are not covered by two type-α P_3s, then either there is a non-packed non-edge between C and D or there is a type-β P_3 between C and D. Thus Reduction Rule 2 or 3 can be applied, a contradiction. Without loss of generality, suppose that v_1v_2 and v_3v_4 are covered by these two type-α P_3s. Assume for contradiction that there is another vertex u such that u and (without loss of generality) v_1 are adjacent, and uv_1 is a packed edge. Since we have applied Reduction Rule 2 exhaustively, there are neither type-β nor type-δ P_3s in the graph. Thus uv_1 must be covered by a type-α or a type-γ P_3.

We claim that there must be a non-packed non-edge from u to a vertex of C. For contradiction, suppose this is not true. Then either v_1v_4, v_2v_3 are covered by two type-α P_3s, respectively, or v_1v_3, v_2v_4 are covered by two type-α P_3s, respectively. In both cases, v_1, v_2, v_3 and v_4 are not in one proto-cluster anymore since after removing the packed edges, v_1, v_2, v_3 and v_4 are not in one connected component, a contradiction. Thus there must be a non-packed non-edge between $V(C)$ and u. Since uv_1 is a packed edge, Reduction Rule 3 can be applied to C and the proto-cluster containing u, a contradiction. Thus there are no edges between $V(C)$ and any other vertices except w.

Suppose that w belongs to a clique of size at least two. Then there must be a non-packed non-edge and a packed edge between C and D (there cannot be more than two packed P_3s between a proto-cluster of size 4 and another proto-cluster). Thus Reduction Rule 3 can be applied, a contradiction. Thus w belongs to a proto-cluster of size one and let this proto-cluster be D. Since w is already incident with two packed P_3s, w is isolated from any other proto-clusters except C. Obviously, $V(C) \cup V(D)$ forms a connected component in the graph. This completes the proof for the lemma. □

Lemma 12. *After applying Reduction Rules 1 to 5 exhaustively, there is no proto-cluster of size* 4.

Proof. Suppose for contradiction that there is a proto-cluster C of size at least 4. If C is an isolated proto-cluster, C must be a clique since otherwise Reduction Rule 1 or 2 can be applied, a contradiction. Then Reduction Rule 4 can be applied and C will be removed from the graph. Thus C is not an isolated proto-cluster. By Lemma 11, there is a proto-cluster D of size 1 such that $V(C) \cup V(D)$ forms a connected component of size 5 in the graph. Then Reduction Rule 5 can be applied, a contradiction. As a result, there is no proto-cluster of size at least 4. This completes the proof for the lemma. □

Summarizing, using the simple Reduction Rules 1 to 4 we have successfully removed all proto-clusters of size at least four.

5.1.2 Decreasing the Proto-cluster Size and Structural Observations. Next, we focus on the structure of proto-clusters of size three and how to remove them as well. First, we observe how connections around proto-clusters of size three look like. See Figure 9 for an illustration of these connections.

Lemma 13. *After applying Reduction Rules 1 to 4 exhaustively, if there is a proto-cluster C of size* 3*, then there must be a proto-cluster B of size* 1 *and a proto-cluster A of size* 1*, such that the vertex pairs between C and B are covered by a type-α P_3 and a type-γ P_3, and the type-γ P_3 connects C and A via B. In addition, C is isolated from any other proto-clusters except B, and B is isolated from any other proto-clusters except A and C.*

Proof. After applying Reduction Rules 1 to 4 exhaustively, let C be a proto-cluster of size 3. If C is isolated from other proto-clusters, then C must be a clique since otherwise Reduction Rule 1 can be applied. However, then Reduction Rule 4 can be applied, a contradiction. Thus we assume that C is not an isolated proto-cluster.

Let the three vertices of C be u_1, u_2, and u_3. Let v be a vertex such that there is an edge between v and $V(C)$. If the vertex pairs between $V(C)$ and v are not covered by a type-α P_3 and a type-γ P_3, then Reduction Rule 2 or 3 can be applied as v can be incident with at most two packed P_3s, a contradiction. Without loss of generality, suppose that u_1, u_3, and v belong to a type-α P_3. Assume for contradiction that there is another vertex w such that w is adjacent to some vertex of $V(C)$ (w can either belong to the same proto-cluster as v or belong to a different proto-cluster from v). If the vertex pairs between $V(C)$ and w are not covered by a type-α P_3 and a type-γ P_3, then Reduction Rule 2 or 3 can be applied to the corresponding P_3 or proto-clusters, a contradiction. If the vertex pairs between $V(C)$ and w are covered by a type-α P_3 and a type-γ P_3, say u_1, u_2 and w belong to the type-α P_3, then u_1, u_2, and u_3 are not in one proto-cluster, a contradiction. It follows that there is no vertex adjacent to one of the vertices of $V(C)$ except v.

Let B be the proto-cluster to which v belongs. Assume for contradiction that $|B| > 1$ and there is another vertex y belonging to B. As argued above, y is not adjacent to any vertex of $V(C)$ and there is a non-packed non-edge between $V(B)$ and $V(C)$. Thus Reduction Rule 3 can be applied, a contradiction. It follows that $|B| = 1$ and C is isolated from any other proto-clusters except B. We

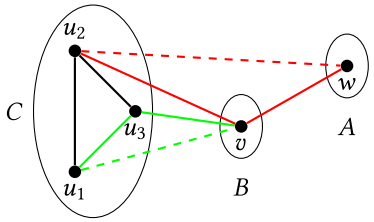

Fig. 9. An example for Lemma 13 and Reduction Rule 6.

have assumed that u_1, u_3 and v belong to a type-α P_3. As argued above, u_2v is covered by a type-γ P_3. Let u_2vx be that type-γ P_3 where x belongs to a proto-cluster A. We claim that $|A| = 1$. Suppose for contradiction that $|A| > 1$ and there is another vertex $z \in V(A)$. Then vz must be a non-packed non-edge since v is already incident with two packed P_3s. Thus Reduction Rule 3 can be applied, a contradiction. It follows that $|A| = 1$. This concludes the proof for the lemma. □

Lemma 13 now suffices to determine a solution around proto-clusters of size three. See Figure 9 for an illustration of the following Rule 6.

Reduction Rule 6. *After applying Reduction Rules 1 to 4 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 C and A 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 14. *Reduction Rule 6 is safe.*

Proof. Given an instance $(G, \mathcal{H})$ of CEMHMP satisfying the condition of Reduction Rule 6, let u_1, u_2, and u_3 be the three vertices of C, let v be the vertex of B and w be the vertex of A. Without loss of generality, let u_1u_3v and u_2vw be two packed P_3s. After applying Reduction Rule 6, we get an instance $(G', \mathcal{H}')$ of CEMHMP. We claim that $(G, \mathcal{H})$ is a YES-instance if and only if $(G', \mathcal{H}')$ is a YES-instance.

For the soundness, suppose that $(G', \mathcal{H}')$ is a YES-instance and S' is a cluster editing set of G' such that $|S'| = |\mathcal{H}'|$. Obviously $S = S' \cup \{u_1v, vw\}$ is a solution to $(G, \mathcal{H})$.

For the completeness, suppose that $(G, \mathcal{H})$ is a YES-instance and S is a cluster editing set of G such that $|S| = |\mathcal{H}|$. If $vw \in S$, then u_2v becomes a non-packed edge between C and B after removing the P_3 u_2vw from $\mathcal{H}$. Thus, in this case we have $u_1v \in S$ as well by Reduction Rule 2, that is, $\{u_1v, vw\} \subseteq S$. Then $S' = S \setminus \{u_1v, vw\}$ is a solution to $(G', \mathcal{H}')$ because by Lemma 13, C and B are isolated from the rest of the graph.

Thus, assume $vw \notin S$ from now on. Then, either $u_2w \in S$ or $u_2v \in S$. First, we assume that $u_2w \in S$, and after inserting u_2w and removing u_2vw from $\mathcal{H}$ we get an instance $(G'', \mathcal{H}'')$ of CEMHMP. Observe that since C is a proto-cluster and u_1u_3 is packed, u_2u_3 is not packed. Thus, u_3u_2w is a non-packed path in G'' and u_3w is a non-packed non-edge. Thus Reduction Rule 1 can be applied to $(G'', \mathcal{H}'')$ and $(G'', \mathcal{H}'')$ is a NO-instance. This contradicts the fact that S is a solution to $(G, \mathcal{H})$. Thus, we have $u_2v \in S$. After deleting u_2v and removing u_2vw from $\mathcal{H}$, u_2v becomes a non-packed non-edge. Thus Reduction Rule 3 can be applied, showing $u_3v \in S$. By Lemma 13, C is isolated from any other proto-clusters except B, and B is isolated from any other proto-clusters except A and C. It follows that in $G \triangle S$, u_1, u_2 and u_3 form a clique of size 3 while v and w form a clique of size 2. Furthermore, $V(G) \setminus \{u_1, u_2, u_3, v, w\}$ forms a cluster graph in $G \triangle S$. Let $\widehat{S} = (S \setminus \{u_2v, u_3v\}) \cup \{vw, u_1v\}$. Obviously $G \triangle \widehat{S}$ is also a cluster graph and $|\widehat{S}| = |\mathcal{H}|$. Thus $\widehat{S}$

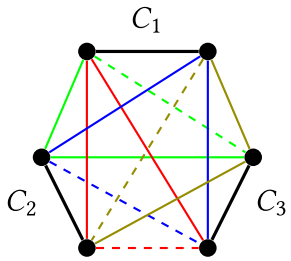

Fig. 10. An example of forming a clique of size 6 in $G \triangle S$. The black edges are non-packed edges. The vertex pairs of the same color which is not black belong to the same packed P_3 and the dashed edges represent non-edges. The same rule of notation applies to the following pictures.

is also a solution to $(G, \mathcal{H})$. It follows that $\widehat{S} \setminus \{vw, u_1v\}$ is a solution for $(G'', \mathcal{H}'')$. This completes the proof for the lemma. □

Corollary 2. *After applying Reduction Rules 1 to 6 exhaustively, there are no isolated cliques in the instance and every proto-cluster of the instance is of size at most 2. Moreover, since the edge in a proto-cluster of size 2 cannot be a packed edge, every packed P_3 in the remaining graph is a type-γ P_3.*

5.1.3 Reducing the Size of Solution Clusters. In the previous section we have successfully removed all proto-clusters of size at least 3. Suppose that after applying Reduction Rules 1 to 6 exhaustively, we have an instance $(G, \mathcal{H})$ of CEMHMP. Suppose that S is a solution to $(G, \mathcal{H})$. Now we consider the size of the clusters in the cluster graph $G \triangle S$. We first show that the largest clique in this graph has size at most 6.

Lemma 15. *After applying Reduction Rules 1 to 6 exhaustively, we have an instance $(G, \mathcal{H})$ of CEMHMP. Suppose that S is a solution to $(G, \mathcal{H})$. Then there is no clique of size larger than 6 in $G \triangle S$.*

Proof. Suppose for contradiction that A is a clique of size at least 7 in $G \triangle S$ and let u be a vertex in A. Then there are at least six vertex pairs between $\{u\}$ and $V(A) \setminus \{u\}$, which are either non-packed edges or covered by packed P_3s. Since u is incident with at most two packed P_3s, at most four vertex pairs between $\{u\}$ and $V(A) \setminus \{u\}$ are covered by a packed P_3. Thus at least two vertex pairs between $\{u\}$ and $V(A) \setminus \{u\}$ are non-packed edges. By Corollary 2, every proto-cluster in G is of size at most 2, a contradiction. This completes the proof for the lemma. □

We can now determine more precisely the structure of potential cliques of size 6 in $G \triangle S$. See Figure 10 as an example.

Lemma 16. *Let $(G, \mathcal{H})$ be an instance of CEMHMP such that the size of every proto-cluster in G is at most 2. Let S be a solution to $(G, \mathcal{H})$ and suppose that A is a clique of size exactly 6 in $G \triangle S$. Then the following statements hold:*

- *The vertices of $V(A)$ belong to three proto-clusters C_1, C_2, and C_3 of size 2 in G.*
- *Every vertex pair between C_1 and C_2, between C_1 and C_3, and 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 in G.*

Proof. Suppose for contradiction that $u \in V(A)$ belongs to a proto-cluster of size 1 in G. Then there are five vertex pairs between $\{u\}$ and $V(A) \setminus \{u\}$, which are covered by packed P_3s. Since u belongs to at most two packed P_3s, at most four vertex pairs between $\{u\}$ and $V(A) \setminus \{u\}$ are covered by a packed P_3, a contradiction.

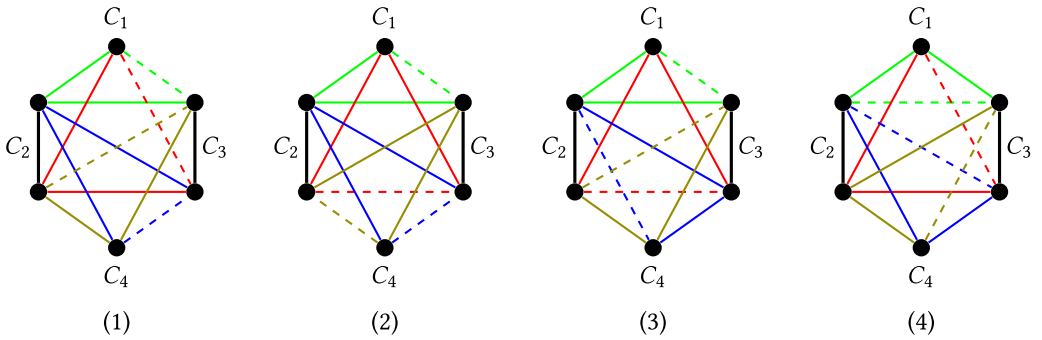

Fig. 11. Some examples of Lemma 18. In Case (1), C_1 is separated from C_2 and C_3, and C_2, C_3, C_4 are merged into a clique of size 5 in $G \triangle S$. In Case (2), C_4 is separated from C_2 and C_3, and C_1, C_2, C_3 are merged into a clique of size 5 in $G \triangle S$. In Case (3), C_1, C_2 are merged into a clique of size 3 and C_3, C_4 are merged into a clique of size 3 such that these two cliques of size 3 are separated from each other. In Case (4), the instance is a NO-instance. Case (3) and Case (4) are not touched by Lemma 18 but they can be handled by Reduction Rule 5 and 4.

Next we show that the vertices of $V(A)$ belong to three proto-clusters C_1, C_2, and C_3 of size 2 in G; see also Figure 10. We see that for every vertex $v \in V(A)$, four of the vertex pairs between $\{v\}$ and $V(A) \setminus \{v\}$ are covered by packed P_3s and the other one is a non-packed edge. Thus every vertex $v \in V(A)$ belongs to two packed P_3s. It follows that for each $i \in [3]$ the proto-cluster C_i is isolated from any other proto-cluster in $G \setminus (V(C_1) \cup V(C_2) \cup V(C_3))$. Note that there are no type-α, type-β, or type-δ P_3s in $\mathcal{H}$ anymore. Thus the edges between the proto-clusters in A are covered by type-γ P_3s. Thus, without loss of generality, let xyz be a P_3 such that $x \in V(C_1)$, $y \in V(C_2)$ and $z \in V(C_3)$. Thus, $V(C_1) \cup V(C_2) \cup V(C_3)$ forms a connected component. This completes the proof for the lemma. □

By the reduction rule that solved small connected components it follows that cliques of size 6 cannot exist in $G \triangle S$.

Lemma 17. *After applying Reduction Rules 1 to 6 exhaustively, we have an instance $(G, \mathcal{H})$ of CEMHMP. Suppose that S is a solution to $(G, \mathcal{H})$. Then there is no clique of size exactly 6 in $G \triangle S$.*

Proof. Suppose for contradiction that A is a clique of size exactly 6 in $G \triangle S$. According to Lemma 16, $V(A)$ induces a connected component of size exactly 6 in the input graph. Then Reduction Rule 5 or Reduction Rule 4 can be applied, a contradiction. This completes the proof for the lemma. □

Now we consider the structure of potential cliques of size 5 in $G \triangle S$. See Figure 11 for examples.

Lemma 18. *After applying Reduction Rules 1 to 3 exhaustively, let $(G, \mathcal{H})$ be an instance of CEMHMP such that the size of every proto-cluster in G is at most 2 and S is a solution to $(G, \mathcal{H})$. Suppose that A is a clique of size exactly 5 in $G \triangle S$. Then there are four proto-clusters C_i for $i \in [4]$ such that the following statements hold:*

- $|C_1| = |C_4| = 1$ *and* $|C_2| = |C_3| = 2$.
- *The vertices of A belong to the three proto-clusters C_1, C_2, and C_3 or to the three proto-clusters C_2, C_3, and C_4.*
- *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.*
- *Furthermore, $V(C_1) \cup V(C_2) \cup V(C_3) \cup V(C_4)$ forms a connected component in G.*

Proof. Suppose for a contradiction that at least three vertices of $V(A)$ belong to proto-clusters of size 1 in G; say $u, v, w \in V(A)$ belong to three distinct proto-clusters of size one, respectively, and two vertices of $V(A)$, say $x, y \in V(A) \setminus \{u, v, w\}$, belong to a proto-cluster of size two or belong to two distinct proto-clusters of size one, respectively. It follows that every vertex pair of $\binom{V(A)}{2}$ is either a non-packed edge or covered by some P_3 of $\mathcal{H}$. Then uv, wv, xv, yv are four vertex pairs that are covered by packed P_3s. Since v is incident with at most two packed P_3s, there are the two following cases: (a) We assume that u, v, x belong to a packed P_3 and w, v, y belong to another packed P_3. We omit the symmetric case that u, v, y belong to a packed P_3 and w, v, x belong to another packed P_3 since the analysis is analogous. (b) We assume that u, v, w belong to a packed P_3 and x, v, y belong to another packed P_3.

For case (a), uw, uy are also covered by one packed P_3 or two distinct packed P_3s. If uw and uy are covered by one packed P_3, then this P_3 is not modification disjoint with the packed P_3 covering w, v, y, a contradiction. If uw and uy are covered by two distinct packed P_3s, then u is incident with three packed P_3s, a contradiction.

For case (b), ux, uy are also be covered by one packed P_3 or two distinct packed P_3s. If ux and uy are covered by one packed P_3, then it is not modification disjoint with the packed P_3 covering x, v, y, a contradiction. If ux and uy are covered by two distinct packed P_3s, then u is incident with three packed P_3s, a contradiction. As all cases lead to a contradiction, it follows that the vertices of $V(A)$ belong to one proto-cluster of size 1 and two proto-clusters of size 2.

Next we show that the vertices in $V(A)$ 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 $|C_2| = |C_3| = 2$; see also Case (1) and Case (2) of Figure 11. Let C_1, C_2, C_3 be the proto-clusters contained in A and without loss of generality let $|V(C_1)| = 1$. Let $V(C_1) = \{x\}, V(C_2) = \{u_1, u_2\}$, and $V(C_3) = \{v_1, v_2\}$. Without loss of generality, let x, u_1, v_1 belong to a packed P_3 and x, u_2, v_2 belong to another packed P_3. Then u_1v_2 and u_2v_1 must be covered by packed P_3s since otherwise Reduction Rule 3 can be applied to C_2 and C_3.

For a contradiction, assume that there are two vertices y_1, y_2 such that y_1, u_1, v_2 belong to one packed P_3 and y_2, u_2, v_1 belong to another packed P_3. Then y_1u_2, y_1v_1 are non-packed non-edges since u_2 and v_1 are each already incident with two packed P_3s. It then follows that Reduction Rule 3 can be applied, a contradiction. It follows that there is a single vertex y such that $\{y, u_2, v_1\}$ and $\{y, u_1, v_2\}$ are vertex sets of P_3s in $\mathcal{H}$. Let C_4 be the proto-cluster to which y belongs.

If $|C_4| > 1$, then there must be a non-packed non-edge between C_4 and C_2 and a non-packed non-edge between C_4 and C_3. Thus Reduction Rule 3 can be applied, a contradiction. Thus $|C_4| = 1$. Since u_1, u_2, v_1, v_2, x, y are all incident with two packed P_3s, the subgraph induced by $V(C_1) \cup V(C_2) \cup V(C_3) \cup V(C_4)$ is isolated from the other parts of the graph. We can view the graph induced by $V(C_1) \cup V(C_2) \cup V(C_3) \cup V(C_4)$ as a complete graph on 6 vertices with five missing edges. Note that the edge between x and y is missing by the condition of this lemma. Suppose that $\{u_1, u_2, v_1, v_2, x, y\}$ does not induce a connected component in G. This is only possible when every edge incident to x (symmetrically, y) is missing because a cut of a complete graph on 6 vertices minus one edge is of size at least 4. However, x (symmetrically, y) is incident with two packed P_3s and thus at most two of the edges incident to x (symmetrically, y) are missing, a contradiction. It follows that $V(C_1) \cup V(C_2) \cup V(C_3) \cup V(C_4)$ forms a connected component in G. This completes the proof for the lemma. □

As for cliques of size 6, the reduction rule that solved small connected components thus took care of cliques of size 5.

Lemma 19. *Let $(G, \mathcal{H})$ be an instance of CEMHMP obtained after applying Reduction Rules 1 to 6 exhaustively. Suppose that S is a solution to $(G, \mathcal{H})$. Then there is no clique of size exactly 5 in $G \triangle S$.*

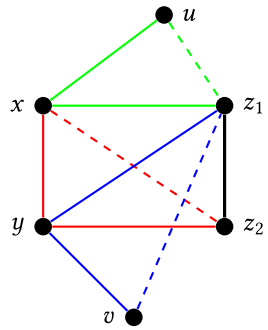

Fig. 12. An example of forming a clique of size 4 in $G \triangle S$. Vertices z_1, z_2 form a proto-cluster of size 2 and each vertex of u, v, x, y belongs to a proto-cluster of size 1.

Proof. Suppose for contradiction that A is a clique of size exactly 5 in $G \triangle S$. According to Lemma 18, $V(A)$ belongs to a connected component of size 6 in the input graph. Then Reduction Rule 5 or Reduction Rule 4 can be applied, a contradiction. This completes the proof for the lemma. □

Summarizing, after applying our reduction rules the cliques in $G \triangle S$ have size at most 4.

5.1.4 Path-like Structures. Next, we aim to get rid of cliques of size 4. This will later enable us to reduce the instance of CEMHMP to 2-SAT. To take care of cliques of size 4, we use a similar strategy as for cliques of size 5 or 6. We first consider the structure of the proto-clusters taking part in the clique (see Figure 12 for an example) and we then devise reduction rules that remove or simplify these proto-clusters. The structure here is more involved. In particular, it is in general not true anymore that cliques of size 4 are contained in small connected components. However, as we will see, these cliques take part in a path-like structure that can either be solved locally, or that behaves analogously to a P_4, see Figure 13 later on. The following lemma formalizes the underlying structure that may contain cliques of size 4.

Lemma 20. *After applying Reduction Rules 1 to 6 exhaustively, let $(G, \mathcal{H})$ be an instance of CEMHMP. Let S be a solution to $(G, \mathcal{H})$. Suppose that A is a clique of size 4 in $G \triangle S$ and $V(A) = \{x, y, z_1, z_2\}$. Then the following statements hold:*

(1) 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, together with z_1 forms a proto-cluster C_1 of size 2 in G.
(2) Vertices x and y form a proto-cluster C_2 of size 1 and a proto-cluster C_3 of size 1 in G, respectively.
(3) There are two vertices u and v such that x, u, z_1 belong to a packed P_3 in G and y, v, z_1 belong to another packed P_3 in G.
(4) Vertices u and v form a proto-cluster C_4 of size 1 and a proto-cluster C_5 of size 1 in G, respectively.
(5) u, v, z_2 cannot belong to the same packed P_3.

Proof. We first show the part of Items (1) and (2) about the partition of $V(A)$ into proto-clusters. For contradiction, suppose that $V(A)$ does not belong to one proto-cluster of size 2 and two proto-clusters of size 1 in G. Then there are two cases: (i) Two vertices of $V(A)$, say x_1, x_2, belong to a proto-cluster C_2 of size two and the other two vertices of $V(A)$, say y_1, y_2, belong to a proto-cluster C_3 of size 2. (ii) All four vertices x_1, x_2, y_1, y_2 of $V(A)$ belong to four distinct proto-clusters C_1, C_2, C_3, and C_4 of size 1, respectively.

Case (i): Since all vertex pairs between C_2 and C_3 need to be covered to form a clique of size 4, without loss of generality, assume that there is a vertex $u \notin V(A)$ such that u, x_1, and y_1 belong to

a packed P_3. Suppose that there is another vertex $u' \notin V(A) \cup \{u\}$ such that u', x_2, and y_2 belong to a packed P_3. Since neither u, x_2, y_1 nor u, x_1, y_2 could belong to a packed P_3 (uy_1 and uy_2 are already covered by the assumed P_3s), one of the vertex pairs ux_2 and uy_2 must be a non-packed non-edge and thus Reduction Rule 3 can be applied, a contradiction. Thus u, x_2 and y_2 belong to a packed P_3. Similarly, we can show that there is another vertex v such that v, x_1, y_2 belong to a packed P_3 and v, x_2, y_1 belong to a packed P_3. It follows that each vertex of $\{x_1, x_2, y_1, y_2, u, v\}$ is incident with two packed P_3s. First we assume that u and v belong to two different proto-clusters, say C_1 and C_4, respectively. If $|C_1| > 1$ or $|C_4| > 1$, then there is a non-packed non-edge involving C_1 or C_4 and thus Reduction Rule 3 can be applied. Thus $|C_1| = |C_4| = 1$. It follows that $V(C_1) \cup V(C_2) \cup V(C_3) \cup V(C_4)$ induces a connected component and Reduction Rule 5 can be applied, a contradiction. Assume that u and v belong to one proto-cluster, say C_1. If $|C_1| > 2$, then Reduction Rule 3 can be applied for the same reason as above. Thus $|C_1| = 2$ and $V(C_1) \cup V(C_2) \cup V(C_3) \cup V(C_4)$ induces a connected component. It follows that Reduction Rule 5 can be applied to this connected component, a contradiction. Therefore, Case (i) does not happen.

Case (ii): Since the vertex pair between each pair of C_1, C_2, C_3, and C_4 needs to be covered to form a clique of size four and each vertex can be in at most two P_3s, without loss of generality, assume that x_1, x_2, y_1 belong to a packed P_3. Pair x_1y_2 also needs to be covered by a packed P_3; observe that by modification-disjointness of the packed P_3s, the third vertex in this P_3 cannot be contained in $V(A)$. Thus, there is another vertex $y_3 \notin V(A)$ such that x_1, y_2, y_3 belong to a packed P_3. The vertex pairs x_2y_2 and y_1y_2 cannot be covered by one packed P_3 since x_2y_1 is already covered by a packed P_3. Thus x_2y_2 and y_1y_2 need to be covered by two distinct P_3s respectively. However, then y_2 is incident with three packed P_3s, a contradiction. Therefore Case (ii) does not happen either. It follows that $V(A)$ consists of one proto-cluster of size 2 and two proto-clusters of size 1.

Next we show that the claims on the P_3s in Item (1) as well as Items (3) and (4) are true. Suppose that A is a clique of size 4 in $G \triangle S$. Let $V(A) = \{x, y, z_1, z_2\}$. By the analysis above, we get that two vertices of A belong to a proto-cluster of size 2 and the other two vertices of A belong to two distinct proto-clusters of size 1 respectively. Without loss of generality, assume that z_1, z_2 form a proto-cluster C_1 of size 2 in G while x and y form a proto-cluster C_2 of size 1 and a proto-cluster C_3 of size 1 in G respectively. See Figure 12 for an illustration.

Since there are three vertex pairs, i.e., $\{xy, xz_1, xz_2\}$, between x and $V(A) \setminus \{x\}$, two of the three vertex pairs are covered by one packed P_3. Moreover, this P_3 cannot contain two vertices of C_1. Without loss of generality, let thus x, y, z_2 belong to a packed P_3. Since xz_1 is also covered by a P_3 and this P_3 is modification-disjoint to the one containing x, y, z_2, there is another vertex $u \notin V(A)$ such that x, u, z_1 belong to a packed P_3 in G.

Also yz_1 needs to be covered by a packed P_3, so there is another vertex v such that y, v, z_1 belong to a packed P_3 (u and v are different as otherwise the P_3s induced by y, v, z_1 and x, u, z_1 are not modification-disjoint). Suppose that u and v belongs to the same proto-cluster of size at least 2. By Corollary 2, this proto-cluster has size exactly 2. Since x and y are incident with two packed P_3s respectively, uy and vx are two non-packed non-edges. Thus Reduction Rule 3 can be applied to the proto-clusters adjacent to these non-edges, a contradiction. It follows that u and v must belong to two distinct proto-clusters. Assume that there is a vertex u' such that u' and u belong to one proto-cluster of size at least two. Since x, z_1 are already incident with two packed P_3s respectively, $u'x$ and $u'z_1$ must be non-packed non-edges. Then Reduction Rule 3 can be applied since ux or uz_1 is a packed edge. It follows that u belongs to a proto-cluster of size one, say C_4. Similarly, we can show that v belongs to a proto-cluster of size one, say C_5.

Finally we show that Item (5) is true. Suppose for contradiction that u, v, z_2 belong to the same proto-cluster. Then every vertex of $\{u, v, x, y, z_1, z_2\}$ is incident with two packed P_3s. It follows that the subgraph induced by $\{u, v, x, y, z_1, z_2\}$ is a connected component in G, which can be handled

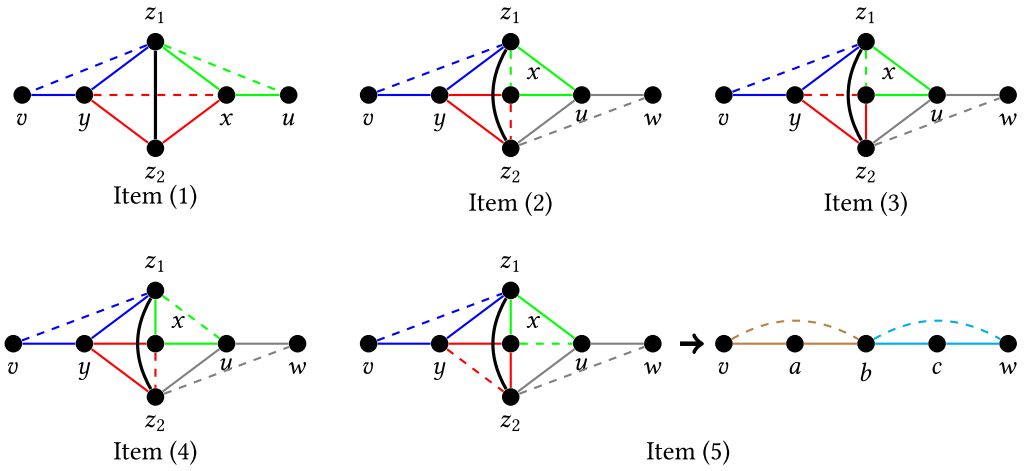

Fig. 13. Examples of Reduction Rule 7. Vertices z_1, z_2 form a proto-cluster of size 2 and each of the other vertices belongs to a proto-cluster of size 1. Note that in Item (3) the P_3 y, x, z_2 is not fully specified by the conditions, that is, its packed non-edge could also be between different vertices.

by Reduction Rule 5. Thus u, v, z_2 cannot belong to the same packed P_3. This completes the proof for the lemma. □

We next leverage the structure observed in Lemma 20 in a reduction rule. Essentially, all the possible ways to realize the structure of Lemma 20 result in a situation that can either be solved directly, or can be replaced by a P_5 with suitable new packed P_3s.

Reduction Rule 7. *After applying Reduction Rules 1 to 6 exhaustively, let C_1, C_2, C_3, C_4, and C_5 be five proto-clusters such that*

- $V(C_1) = \{z_1, z_2\}, V(C_2) = \{x\}, V(C_3) = \{y\}, V(C_4) = \{u\}, V(C_5) = \{v\}$,
- *x, y, z_2 belong to a packed P_3,*
- *x, u, z_1 belong to a packed P_3, and*
- *y, v, z_1 belong to a packed P_3.*

Check which of the following conditions are satisfied and apply the corresponding data reduction.

If uz_2 and vz_2 are non-packed non-edges, then

(1) delete the edges vy and ux, insert an edge to the packed non-edge of the P_3 which covers xy and remove the corresponding packed P_3s from $\mathcal{H}$.

Otherwise, if there is a vertex w such that u, w, z_2 belong to a packed P_3, then do reductions according to the following cases:

(2) xz_1 and xz_2 are packed non-edges: Return NO.
(3) xz_1 is a packed non-edge and xz_2 is a packed edge: Delete the edge uw, delete the edges between y and $\{x, z_1, z_2\}$, and add an edge to the non-edge xz_1. Remove the corresponding packed P_3s from $\mathcal{H}$.
(4) xz_1 is a packed edge and xz_2 is a packed non-edge: Delete the edge vy, delete the edges between u and $\{x, z_1, z_2\}$, and add an edge to the non-edge xz_2. Remove the corresponding packed P_3s from $\mathcal{H}$.

(5) xz_1 and xz_2 are packed edges: Replace the subgraph induced by $\{u, v, w, x, y, z_1, z_2\}$ with two P_3s vab and bcw. Remove the four packed P_3s incident with one vertex of $\{u, x, y, z_1, z_2\}$ from $\mathcal{H}$, and add vab and bcw to $\mathcal{H}$.

Otherwise, if there is a vertex w' such that v, w', z_2 belong to a packed P_3, then do reductions according to the following cases:

(6) yz_1 and yz_2 are packed non-edges: Return NO.
(7) yz_1 is a packed non-edge and yz_2 is a packed edge: Delete the edge vw', delete the edges between x and $\{y, z_1, z_2\}$, and add an edge to the non-edge yz_1. Remove the corresponding packed P_3s from $\mathcal{H}$.
(8) yz_1 is a packed edge and yz_2 is a packed non-edge: Delete the edge ux, delete the edges between v and $\{y, z_1, z_2\}$, and add an edge to the non-edge yz_2. Remove the corresponding packed P_3s from $\mathcal{H}$.
(9) yz_1 and yz_2 are packed edges: Replace the subgraph induced by $\{u, v, w', x, y, z_1, z_2\}$ with two P_3s $w'ab$ and bcu. Remove the four packed P_3s incident with one vertex of $\{v, x, y, z_1, z_2\}$ from $\mathcal{H}$, and add $w'ab$ and bcu to $\mathcal{H}$.

Lemma 21. *Reduction Rule 7 is safe.*

Proof. Note that Items (2) to (5) are symmetric to the Items (6) to (9), respectively. To be more precise, we can relabel the vertices in the subgraph induced by $\{u, v, x, y, z_1, z_2, w\}$ of Item (2) as follows: we exchange the labeling of x and y, exchange the labeling of u and v, and relabel vertex w as w'. Then we get a subgraph induced by $\{u, v, x, y, z_1, z_2, w'\}$ satisfying the condition of Item (6). Similarly we show that Item (3) is symmetric to Item (7), Item (4) is symmetric to Item (8), and Item (5) is symmetric to Item (9). Thus, the safeness of Items (6) to (9) follows from the safeness of Items (2) to (5).

Thus, it is enough to prove the correctness of Items (1) to (5). In the following, to ease arguments we will sometimes successively determine edits that are being made by the solution. We then tacitly assume that the packed P_3 corresponding to the edit is removed from $\mathcal{H}$, obtaining a new instance of CEMHMP. We also sometimes leverage the correctness of the previous reduction rules and apply them to the resulting instances. This implies additional edits that can be assumed to be in the solution without loss of generality.

Item (1). Suppose that $(G, \mathcal{H})$ is an instance of CEMHMP satisfying the condition of Item (1) of Reduction Rule 7. First, we claim that vz_1 and uz_1 must be packed non-edges. Suppose for contradiction that uz_1 or vz_1 is a packed edge. Since vz_2 and uz_2 are non-packed non-edges as in the assumption, Reduction Rule 3 can be applied, a contradiction. Let F be the set of vertex pairs edited by Item (1). Observe that F contains exactly one vertex pair of each of the packed P_3s incident with one of the vertices of $\{x, y, z_1, z_2\}$. After applying the operations of Item (1) we get an instance $(G' = G \triangle F, \mathcal{H}')$. We claim that $(G, \mathcal{H})$ is a YES-instance if and only if $(G', \mathcal{H}')$ is a YES-instance.

First assume that $(G', \mathcal{H}')$ has a solution S'. Observe that $\{x, y, z_1, z_2\}$ is a cluster in G' and no packed P_3 is incident with any vertex of this cluster. Thus, $S' \cap F = \emptyset$ and, moreover, $S' \cup F$ is a solution to $(G, \mathcal{H})$.

Now assume that $(G, \mathcal{H})$ has a solution S. Then $G \triangle S$ is a cluster graph. We claim that z_2 is not incident with any other packed P_3s except the one covering xy. Suppose that there is another vertex z_3 such that z_3z_2 is a packed edge. Then z_3z_1 is a non-packed non-edge since z_1 is already incident with two packed P_3s. Then Reduction Rule 3 can be applied because z_1 and z_2 are together in a proto-cluster, a contradiction. Thus z_2 is not incident with any other packed P_3s except the one covering xy.

Let $S_1 \subseteq S$ be the set of vertex pairs which are packed edges or packed non-edges in the subgraph of G induced by $\{u, v, x, y, z_1, z_2\}$. We claim that $\widehat{S} = (S \setminus S_1) \cup F$ is also a solution to $(G, \mathcal{H})$. Since S is a solution to $(G, \mathcal{H})$, S_1 must contain exactly one vertex pair of each of the packed P_3s incident with one of the vertices of $\{x, y, z_1, z_2\}$. Since $F \cap (S \setminus S_1) = \emptyset$, $S_1 \subseteq S$ and $|F| = |S_1|$, we get that $|\widehat{S}| = |S|$. Since $G \triangle S$ is a cluster graph, the subgraph of $G \triangle S$ induced by $V(G) \setminus \{x, y, z_1, z_2\}$ is also a cluster graph $\widetilde{G}$. x, y, z_1 are not incident with any other packed P_3s except the ones covering xy, yz_1, xz_1 since each of them is incident with two packed P_3s in G. We have shown that z_2 is not incident with any other packed P_3s except the one covering xy in the previous paragraph. Since x, y belong to two proto-clusters of size one and z_1, z_2 belong to a proto-cluster of size two, there is no edge between $V(G) \setminus \{u, v, x, y, z_1, z_2\}$ and $\{x, y, z_1, z_2\}$. It follows that $G \triangle \widehat{S}$ is a cluster graph consisting of two isolated parts, i.e., $\widetilde{G}$ and the clique of size four formed by $\{x, y, z_1, z_2\}$. It follows that $S \setminus S_1$ is a solution to $(G', \mathcal{H}')$. As a result, Item (1) is safe.

Preparation for Items (2) to (5). For the proof of the correctness of Items (2)–(5), we claim that vz_1 and wz_2 are packed non-edges. Suppose for contradiction that vz_1 or wz_2 is a packed edge. Since z_1, z_2 are already incident with two packed P_3s, vz_2 and wz_1 must be non-packed non-edges. Then Reduction Rule 3 can be applied, a contradiction. Thus vz_1 and wz_2 are packed non-edges. Also, we claim that vz_2 and wz_1 are non-packed non-edges since z_1, z_2 are both incident with two packed P_3s and there is no proto-cluster of size more than 2 by Corollary 2.

Item (2). For a contradiction, suppose that an instance $(G, \mathcal{H})$ of CEMHMP satisfying the condition of Item (2) of Reduction Rule 7 has a solution S. Observe that vx is a non-packed non-edge because x is in two packed P_3s with other vertices. Thus, at least one packed edge of vy, xy belongs to S since otherwise $\{v, x, y\}$ would induce a P_3 in $G \triangle S$.

Suppose that $xy \in S$. Then yz_2 cannot be removed by S and thus $yz_1 \notin S$ because otherwise $\{y, z_1, z_2\}$ would induce a P_3 in $G \triangle S$. Furthermore, $vz_1 \notin S$, because otherwise $\{v, z_1, z_2\}$ would induce a P_3 in $G \triangle S$. Thus, $vy \in S$. Consider the instance resulting from making the edits xy an vy. Then, the subgraph induced by $\{y, z_1, z_2\}$ is a proto-cluster. Since uy is a non-packed non-edge and uz_1 is a packed edge, Reduction Rule 3 can be applied, which deletes uz_1 and makes ux become a non-packed edge. Now u and x form a proto-cluster of size two. Observe that now xz_2 is a non-packed non-edge. Thus again by Reduction Rule 3, uz_2 is deleted and uw becomes a non-packed edge. Then Reduction Rule 1 can be applied to the proto-cluster formed by u, w, x, a contradiction to S being a solution.

Suppose that $vy \in S$. After making this modification, yz_1 becomes a non-packed edge, so $yz_2 \notin S$ since otherwise Reduction Rule 1 can be applied. If $xz_2 \in S$, then $xz_1 \in S$ since otherwise $\{x, z_1, z_2\}$ would induce a P_3. After making these two modifications, uz_1 and ux become non-packed edges. Since uy is a non-packed non-edge, Reduction Rule 1 can be applied to the proto-cluster formed by u, x, y, z_1, z_2. Thus $xz_2 \notin S$. It follows that $xy \in S$ because $yz_2 \notin S$ as argued above. After making this modification, by Reduction Rule 3, uz_1 is deleted and ux becomes a non-packed edge. Now u, x form a proto-cluster of size two. Again by Reduction Rule 3, uz_2 is deleted and uw becomes a non-packed edge. Then Reduction Rule 1 can be applied to the proto-cluster formed by u, w, x, a contradiction.

As a result, $(G, \mathcal{H})$ is a NO-instance and Item (2) is safe.

Item (3). Given an instance $(G, \mathcal{H})$ of CEMHMP satisfying the condition of Item (3) of Reduction Rule 7, let F be the set of vertex pairs edited by Item (3). Observe that F contains exactly one vertex pair of each of the packed P_3s incident with one of the vertices of $\{u, x, y, z_1, z_2\}$. After applying the operations of Item (3) we get an instance $(G' = G \triangle F, \mathcal{H}')$. We claim that $(G, \mathcal{H})$ is a YES-instance if and only if $(G', \mathcal{H}')$ is a YES-instance.

First assume that $(G', \mathcal{H}')$ has a solution S'. Observe that y is not in a packed P_3 in $\mathcal{H}'$ and thus $\{v, y\}$ forms a cluster in $G' \triangle S'$. Similarly, w is not adjacent to any vertex of u, x, z_1, z_2. Thus $S' \cup F$ is a solution to $(G, \mathcal{H})$.

Now assume that $(G, \mathcal{H})$ has a solution S. We claim that $F \subseteq S$. Suppose for contradiction that $xz_1 \notin S$. Then there are two cases: (1) $ux \in S$ and (2) $uz_1 \in S$.

Case (1): If $ux \in S$, then uz_1 becomes a non-packed edge and xz_1 becomes a non-packed non-edge. It follows that $uz_2 \notin S$ and $xz_2 \in S$ since otherwise Reduction Rule 1 can be applied. Since wz_1 is a non-packed non-edge, $wz_2 \notin S$. Thus $uw \in S$. We now distinguish whether xy is an edge or a non-edge. If xy is a non-edge, then yz_2 is an edge. Thus, yz_2 becomes a non-packed edge after xz_2 is deleted. Since uy is a non-packed non-edge, Reduction Rule 1 can be applied, a contradiction. Otherwise, if xy is an edge, then yz_2 is a non-edge. Then y and x form a cluster of size two. Recall that vz_1 is a packed non-edge. Thus, yz_1 is a packed edge and uy is a non-packed non-edge, meaning that Reduction Rule 3 can be applied and $yz_1 \in S$. Thus vy becomes a non-packed edge. Since vx is a non-packed non-edge, Reduction Rule 1 can be applied, a contradiction.

Case (2): If $uz_1 \in S$, then $uz_2 \in S$ since otherwise Reduction Rule 1 can be applied. Thus ux and uw become non-packed edges after uz_1 and uz_2 are deleted. Since xw is a non-packed non-edge, Reduction Rule 1 can be applied, a contradiction.

Since both cases lead to a contradiction it follows that $xz_1 \in S$. Thus, $xz_2, uz_2 \notin S$. Thus $uw \in S$ since otherwise Reduction Rule 1 can be applied.

It remains to show that all edges between y and $\{x, z_1, z_2\}$ are in S. Now u, x, z_1, z_2 belong to one proto-cluster. Then by Reduction Rule 3, indeed $yz_1 \in S$. Thus u, x, z_1, z_2 are in one proto-cluster and y is in a different proto-cluster. Since uy is a non-packed non-edge, if xy is an edge, $xy \in S$ since otherwise Reduction Rule 1 can be applied. Then by Reduction Rule 3, $yz_1 \in S$. Similarly, if yz_2 is an edge, $yz_2 \in S$ since otherwise Reduction Rule 1 can be applied. Then by Reduction Rule 3, $yz_1 \in S$. As a result, $F \subseteq S$.

We claim that $\widehat{S} = S \setminus F$ is a solution to $(G', \mathcal{H}')$. Since u, x, y, z_1, z_2 are already incident with two packed P_3s, $\{u, x, y, z_1, z_2\}$ are isolated from $V(G) \setminus \{u, v, w, x, y, z_1, z_2\}$ in G. It follows that in $G \triangle S$, v, y belong to a clique of size two, u, x, z_1, z_2 belong to a clique of size four and $V(G) \setminus \{u, v, x, y, z_1, z_2\}$ induces a cluster graph such that there are no edges between $\{u, v, x, y, z_1, z_2\}$ and $V(G) \setminus \{u, v, x, y, z_1, z_2\}$. Thus $G' \triangle \widehat{S}$ is a cluster graph and $|\widehat{S}| = |\mathcal{H}'|$. As a result, Item (3) is safe.

Item (4). Note that Item (4) is symmetric to Item (3). To be more precise, we can relabel the vertices in the subgraph of Item (4) as follows: we exchange the labeling of z_1 and z_2, exchange the labeling of u and y, and exchange the labeling of w and v. Then we get a subgraph satisfying the condition of Item (3). Thus, the safeness of Item (4) follows from the safeness of Item (3).

Item (5). Let $(G, \mathcal{H})$ be an instance of CEMHMP satisfying the condition of Item (5) of Reduction Rule 7. Since xz_1, xz_2 are packed edges, there are two packed edges between y and $\{x, z_1, z_2\}$; let the set of the two packed edges be W_y. Also, there are two packed edges between u and $\{x, z_1, z_2\}$; let the set of the two packed edges be W_u. After applying the operations of Item (5) we get an instance $(G', \mathcal{H}')$. We claim that $(G, \mathcal{H})$ is a YES-instance if and only if $(G', \mathcal{H}')$ is a YES-instance.

For soundness, suppose that $(G', \mathcal{H}')$ has a solution S'. Thus $G' \triangle S'$ is a cluster graph. There are nine possible cases for which vertex pairs in the two P_3s are contained in S'. However, only the following five cases are valid as in the other cases it is easy to see that S' is not a cluster editing set:

(1) $\{ab, cw\} \subseteq S'$. Since $G' \triangle S'$ is a cluster graph, the subgraph of $G' \triangle S'$ induced by $V(G) \setminus \{b, c\}$ is also a cluster graph. Let $S = (S' \setminus \{ab, cw\}) \cup W_y \cup (\{ux, uz_1, uz_2\} \setminus W_u) \cup \{uw\}$. Note that, by construction of G', there are no edges or packed non-edges from a, b, c to other vertices except to

v, w. In particular, this means that $\{v, a\}$ is a cluster in $G'\triangle S'$. Thus $G\triangle S$ is also a cluster graph and $|S| = |\mathcal{H}|$. Thus $(G, \mathcal{H})$ is a YES-instance.

(2) $\{va, bc\} \subseteq S'$. We can show that $(G, \mathcal{H})$ has a solution in a similar way to that of Case (1).

(3) $\{vb, bc\} \subseteq S'$. Since vertices a, b and c are not adjacent to any vertex of $V(G') \setminus \{a, b, c, v, w\}$ in G', $\{v, a, b\}$ induces a triangle which is a connected component and $\{c, w\}$ induces a clique of size two which is also a connected component in $G'\triangle S'$. Let $S = (S' \setminus \{vb, bc\}) \cup W_u \cup (\{yx, yz_1, yz_2\} \setminus W_y) \cup \{vy\}$. It follows that $G\triangle S$ is a cluster graph and $|S| = |\mathcal{H}|$. Thus $(G, \mathcal{H})$ is a YES-instance.

(4) $\{ab, bw\} \subseteq S'$. We can show that $(G, \mathcal{H})$ has a solution in a similar way to that of Case (3).

(5) $\{ab, bc\} \subseteq S'$. Since $G'\triangle S'$ is a cluster graph, the subgraph of $G'\triangle S'$ induced by $V(G) \setminus \{b, c\}$ is also a cluster graph. Let $S = (S' \setminus \{ab, bc\}) \cup \{yx, yz_1, uz_1, uz_2\}$. Note that, by construction of G', there are neither edges nor packed non-edges from a, b, c to other vertices except to v, w. It follows that $\{v, a\}$ and $\{c, w\}$ are two clusters in $G'\triangle S'$. Thus $G\triangle S$ is also a cluster graph and $|S| = |\mathcal{H}|$. Thus $(G, \mathcal{H})$ is a YES-instance.

For completeness, suppose that $(G, \mathcal{H})$ has a solution S. We can check that there are only three possible cases ((1) $vy \in S$, $uw \notin S$; (2) $uw \in S$, $vy \notin S$ (3) $vy \notin S$, $uw \notin S$). Readers can easily check that the the other case in which $vy \in S$, $uw \in S$ is invalid as there is no such cluster editing set S.

(1) $F_1 = W_u \cup (\{yx, yz_1, yz_2\} \setminus W_y) \cup \{vy\} \subseteq S$. Since vertices u, x, y, z_1, z_2 are not adjacent to any vertex of $V(G) \setminus \{u, v, w, x, y, z_1, z_2\}$ in G, $\{x, y, z_1, z_2\}$ induces a clique of size four which is a connected component and $\{u, w\}$ induces a clique of size two which is also a connected component in $G\triangle S$. Let $S' = S \setminus F_1 \cup \{va, bc\}$. It follows that $G'\triangle S'$ is a cluster graph and $|S'| = |\mathcal{H}'|$. Thus $(G', \mathcal{H}')$ is a YES-instance.

(2) $F_2 = W_y \cup (\{ux, uz_1, uz_2\} \setminus W_u) \cup \{uw\} \subseteq S$. Since vertices u, x, y, z_1, z_2 are not adjacent to any vertex of $V(G) \setminus \{u, v, w, x, y, z_1, z_2\}$ in G, $\{u, x, z_1, z_2\}$ induces a clique of size four which is a connected component and $\{v, y\}$ induces a clique of size two which is also a connected component in $G\triangle S$. Let $S' = S \setminus F_2 \cup \{ab, cw\}$. It follows that $G'\triangle S'$ is a cluster graph and $|S'| = |\mathcal{H}'|$. Thus $(G', \mathcal{H}')$ is a YES-instance.

(3) $F_3 = W_u \cup W_y \subseteq S$. Since vertices u, x, y, z_1, z_2 are not adjacent to any vertex of $V(G) \setminus \{u, v, w, x, y, z_1, z_2\}$ in G, $\{x, z_1, z_2\}$, $\{v, y\}$ and $\{u, w\}$ are three clusters in $G\triangle S$. Let $S' = (S \setminus F_3) \cup \{ab, bc\}$. It follows that $G'\triangle S'$ is a cluster graph and $|S'| = |\mathcal{H}'|$. Thus $(G', \mathcal{H}')$ is a YES-instance.

As a result, Item (5) is safe. This completes the proof for the lemma. □

After applying Reduction Rule 7, Reduction Rule 4 can be applied to remove the isolated cliques.

Lemma 22. *After applying Reduction Rules 1 to 7 exhaustively, let $(G, \mathcal{H})$ be an instance of CEMHMP which has a solution S. Then there is no clique of size at least 4 in $G\triangle S$.*

Proof. By Lemma 15, 17 and 19, there is no clique of size at least 5 in $G\triangle S$. Suppose for contradiction that A is a clique of size 4 in $G\triangle S$. Let $V(A) = \{x, y, z_1, z_2\}$. Then by Lemma 20, 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, forms with z_1 a proto-cluster C_1 of size two in G. Meanwhile, 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, and u and v form a proto-cluster C_4 of size one and C_5 of size one in G respectively. There are five cases:

(1) uz_2 and vz_2 are non-packed non-edges. Then Item (1) of Reduction Rule 7 can be applied.

(2) uz_2 is a packed edge and vz_2 is a non-packed non-edge. Then one of Items (2)–(5) can be applied.

(3) uz_2 is a packed non-edge and vz_2 is a non-packed non-edge. By Item (5) of Lemma 20, u, v, z_2 cannot belong to one packed P_3. Thus there is another vertex w such that u, w, z_2 belong to a packed P_3 and uz_2 is a packed non-edge. Thus wz_2 is a packed edge. Since z_1 is in a proto-cluster

of size one and z_1 is already incident with two packed P_3s, wz_1 must be a non-packed non-edge. Since z_1, z_2 belong to one proto-cluster, Reduction Rule 3 can be applied.

(4) vz_2 is a packed edge and uz_2 is a non-packed non-edge. Then one of Items (6)–(9) of Rule 7 can be applied.

(5) vz_2 is a packed non-edge and uz_2 is a non-packed non-edge. By Item (5) of Lemma 20, u, v, z_2 cannot belong to one packed P_3. Thus there is another vertex w' such that v, w', z_2 belong to a packed P_3 and vz_2 is a packed non-edge. Thus $w'z_2$ is a packed edge. Since z_1 is in a proto-cluster of size one and z_1 is already incident with two packed P_3s, $w'z_1$ must be a non-packed non-edge. Since z_1, z_2 belong to one proto-cluster, reduction Rule 3 can be applied.

It follows that there is no clique of size 4 in $G \triangle S$. This completes the proof for the lemma. □

5.1.5 Reduction to 2-SAT. First, we introduce a new problem called Cluster Deletion above modification-disjoint P_3 packing. The formal definition is as follows:

Cluster Deletion above modification-disjoint P_3 packing (CDaMP)

Input: A graph $G = (V, E)$, a modification-disjoint packing $\mathcal{H}$ of induced P_3s of G, and a nonnegative integer ℓ.

Question: Is there a *cluster deletion set*, i.e., a set of edges $S \subseteq E$ such that $G' = (V, E \setminus S)$ is a disjoint union of cliques, with $|S| - |\mathcal{H}| \leq \ell$?

Note that in the definition of CDaMP, the P_3s of $\mathcal{H}$ are still modification-disjoint although the solution to the problem contains only edge deletions. Since in this paper we only need to consider the special case of CDaMP where $\ell = 0$, we use the tuple $(G, \mathcal{H})$ to represent an instance of CDaMP in which $\ell = 0$.

Lemma 23. *Given an instance $(G, \mathcal{H})$ of CEMHMP, after applying Reduction Rules 1 to 7 exhaustively, we get an instance $(G', \mathcal{H}')$ of CEMHMP. Then $(G, \mathcal{H})$ is a YES-instance of CEMHMP if and only if $(G', \mathcal{H}')$ is a YES-instance of CDaMP.*

Proof. *Soundness.* Assume that $(G', \mathcal{H}')$ is a YES-instance of CDaMP and S' is a cluster deletion set of size $|\mathcal{H}'|$. Obviously S' is also a cluster editing set of G'. Thus $(G', \mathcal{H}')$ is a YES-instance of CEMHMP. It follows that $(G, \mathcal{H})$ is a YES-instance of CEMHMP.

Completeness. Assume that $(G, \mathcal{H})$ is a YES-instance of CEMHMP. Then $(G', \mathcal{H}')$ is a YES-instance of CEMHMP and let S' be its solution. By Lemma 22, there is no clique of size at least four in $G' \triangle S'$. By Observation 2, every non-edge of S' is a packed non-edge. Let $uw \in S'$ be a non-edge of G' which is covered by a P_3 uvw of $\mathcal{H}'$. Then in $G' \triangle S'$, $\{u, v, w\}$ induces a triangle which is a connected component. It follows that $S' \setminus \{uw\} \cup \{uv\}$ is also a solution to $(G', \mathcal{H}')$. Let $S_1 \subseteq S'$ be the set of non-edges of S'. Then there is a set S_2 of packed edges of G' such that $(S' \setminus S_1) \cup S_2$ is a cluster deletion set for G' of size $|\mathcal{H}'|$. Thus $(G', \mathcal{H}')$ is a YES-instance of CDaMP. This completes the proof for the lemma. □

Given an instance $(G, \mathcal{H})$ of CEMHMP, after applying Reduction Rules 1 to 7 exhaustively, we get an instance $(G', \mathcal{H}')$ 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 $e_0, e_1, \ldots, e_{\lambda-1}$ according to this ordering. We construct an instance of 2-SAT with λ variables $x_0, x_1, \ldots, x_{\lambda-1}$ as follows. First, initialize the 2-SAT formula $\Phi = \text{true}$. 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, let $uv = e_p$ and $vw = e_q$ and update $\Phi \leftarrow \Phi \wedge (x_p \vee x_q)$. This completes the construction of the 2-SAT instance.

We now show that formula Φ and $(G, \mathcal{H})$ are equivalent instances.

Lemma 24. *Given an instance $(G, \mathcal{H})$ of CEMHMP, after applying Reduction Rules 1–7 exhaustively, we get an instance $(G', \mathcal{H}')$ of CDAMP. We construct a 2-SAT formula Φ as described above. Then $(G, \mathcal{H})$ is a YES-instance if and only if Φ is satisfiable.*

Proof. *Completeness.* Assume that $(G, \mathcal{H})$ is a YES-instance. By Lemma 23, $(G', \mathcal{H}')$ is a YES-instance of CDAMP and let S' be a cluster deletion set for G' of size $|\mathcal{H}'|$. Let α be an assignment to Φ such that $\alpha(x_i) = \text{true}$ if and only if $e_i \in S'$ for $i = 0, \ldots, \lambda - 1$. We claim that α is a satisfying assignment to Φ. Since $|S'| = |\mathcal{H}'|$ and the P_3s of $\mathcal{H}'$ are modification-disjoint, S' contains exactly one edge of every packed P_3 of $\mathcal{H}'$. It follows that for every P_3 $xyz \in \mathcal{H}'$ ($xy = e_i, yz = e_j$), the two clauses $(x_i \vee x_j)$ and $(\neg x_i \vee \neg x_j)$ are satisfied. Since S' is a solution to $(G', \mathcal{H}')$, there is no induced P_3 in $G' \triangle S'$. Thus for every induced P_3 uvw in G' such that uv and vw belong to two distinct packed P_3s ($uv = e_p, vw = e_q$) respectively, at least one edge of $\{uv, vw\}$ belongs to S' and the clause $(x_p \vee x_q)$ is satisfied. As a result, α is a satisfying assignment to Φ.

Soundness. Assume that Φ is satisfiable and let α be a satisfying assignment to Φ. Let $S' = \{e_i \mid \alpha(x_i) = \text{true}\}$. We will show that S' is a cluster deletion set for G' of size $|\mathcal{H}'|$. First we claim that for every induced P_3 $xyz \in \mathcal{H}'$, exactly one edge of xy and yz belongs to S'. Assume that $e_i = xy$ and $e_j = yz$ for some $i, j \in \{0, \ldots, \lambda - 1\}$. Since $(x_i \vee x_j)$ and $(\neg x_i \vee \neg x_j)$ are two clauses of Φ and α is a satisfying assignment to Φ, either $x_i = \text{false}, x_j = \text{true}$ or $x_i = \text{true}, x_j = \text{false}$ holds. Thus the claim is true and $|S'| = |\mathcal{H}'|$.

It remains to show that S' is indeed a cluster deletion set, that is, there is no induced P_3 in $G' \triangle S'$. We show this by going over the possibilities of such an induced P_3 for whether its edges are packed or not. Before that, for every induced P_3 uvw in G' such that uv and vw belong to two distinct P_3s of $\mathcal{H}'$, let $uv = e_p$ and $vw = e_q$ for some $p, q \in \{0, \ldots, \lambda - 1\}$. By the construction, $(x_p \vee x_q)$ is a clause of Φ so it is satisfied by α. Thus at least one edge of uvw belongs to S'.

First, by Corollary 2, there is no proto-cluster of size at least three in G'. Thus there is no induced P_3 abc in $G' \triangle S'$ such that ab and bc are non-packed edges in G'.

Second, we claim that there is no induced P_3 xyz in $G' \triangle S'$ such that both xy and yz are packed edges in G'. Suppose for a contradiction that there is an induced P_3 xyz in $G' \triangle S'$ such that both xy and yz are packed edges in G'. Then xy and yz must be covered by two distinct packed P_3s, since otherwise xy or yz belongs to S' by the definition of S'. We contend that xz must be a packed edge covered by another packed P_3 in G', i.e., xy, yz and xz are covered by three distinct packed P_3s in G'. First of all, xz is an edge of G', because otherwise xyz would be an induced P_3 in G'. Then xy or yz would belong to S' by the definition of S', a contradiction. If xz is a non-packed edge in G', then xz is an edge in $G' \triangle S'$ since S' can only contain vertex pairs covered by packed P_3s. However, this contradicts the assumption that xyz is an induced P_3 in $G' \triangle S'$. Therefore, xz is indeed a packed edge in G'.

By the construction of S', no two of xy, yz, and xz are covered by the same packed P_3 as otherwise one of the three edges belongs to S'. Thus xy, yz and xz are covered by three distinct packed P_3s in G'. Suppose that without loss of generality, xz is covered by $uxz \in \mathcal{H}'$. Note that $ux, xy, yz \notin S'$ as by our assumption, xyz is an induced P_3 in $G' \triangle S'$. Since y is already incident with two packed P_3s, uy is either a non-packed non-edge in G' or a non-packed edge in G'. If uy is a non-packed non-edge in G', then uxy is an induced P_3 in G'. Let $ux = e_i$ and $xy = e_j$, then the clause $(x_i \vee x_j)$ of Φ is not satisfied, a contradiction. Thus uy is a non-packed edge.

By the analysis above, there is a vertex w such that x, y, w belong to a packed P_3 and there is a vertex w' such that y, z, w' belong to a packed P_3. We have the following subcases: (1) the subgraph induced by $\{x, y, z, u, w, w'\}$ is isolated from $G' \setminus \{x, y, z, u, w, w'\}$. Then Reduction Rule 5 can be applied; (2) Either wy is a non-packed non-edge and wu is a packed edge, or $w'y$ is a non-packed

non-edge and $w'u$ is a packed edge in G'. Then Reduction Rule 3 can be applied as uy is a proto-cluster of size 2 in G' by our analysis above; (3) the subcases (1) and (2) do not hold. Then we can check that one of the items of Reduction Rule 7 can be applied (note that which item can be applied depends on the structure of the subgraph we are considering): There could be another vertex a such that a, w, u belong to one packed P_3 or another vertex a' such that a', w', u belong to one packed P_3. If no such vertices a and a' exist, then Item (1) of Reduction Rule 7 applies. Otherwise, one of the other items applies. To see this more clearly, we relabel the vertices as follows: $y \leftarrow z_1, u \leftarrow z_2, w \leftarrow u, z \leftarrow y, x \leftarrow x, w' \leftarrow v, a \leftarrow w, a' \leftarrow w'$. Thus, all three subcases above contradict the assumption that no reduction rules can be applied in G'. Therefore, the claim holds, that is, there is no induced P_3 xyz in $G'\triangle S'$ such that both xy and yz are packed edges in G'.

Third and finally, we claim that there is no induced P_3 in $G'\triangle S'$ such that one edge of this P_3 is a non-packed edge in G' and the other edge is a packed edge in G'. Suppose for a contradiction that there is such a P_3 uvw in $G'\triangle S'$ such that uv is a non-packed edge and vw is a packed edge in G'. Then there is another vertex x such that v, w, x belong to a packed P_3 in G'. Since Reduction Rule 3 cannot be applied to $(G', \mathcal{H}')$, uw must be covered by a packed P_3 in G', i.e., there is a vertex y such that u, w, y belong to a packed P_3 in G'. We contend that at least one of vy and ux are covered by a packed P_3. Suppose for contradiction that both vy and ux are non-packed non-edges. Then, if uy is a packed edge, Reduction Rule 3 could be applied. Thus we can assume that uy is a packed non-edge. Since uvw is an induced P_3 in $G'\triangle S'$, $uw, wx \in S$. Then vwy is an induced P_3 in G'. Assume that $vw = e_p$ and $wy = e_q$. Then the assignment α cannot satisfy $(x_p \vee x_q)$, which is a clause of Φ, contradicting that α is a satisfying assignment to Φ. Thus we can assume that there is a vertex z such that v, y, z belong to a packed P_3 in G' (the analysis for the case that there is a vertex z' such that u, x, z' belong to a packed P_3 in G' is similar).

We have the following subcases: (1) the subgraph induced by $\{x, y, z, u, v, w\}$ is isolated from $G' \setminus \{x, y, z, u, v, w\}$. Then Reduction Rule 5 can be applied; (2) vz is a non-packed non-edge and uz is a packed edge. Then Reduction Rule 3 can be applied as uv is a proto-cluster of size 2 in G'; (3) the subcase (1) and (2) do not hold. Then we can check that one of the items of Reduction Rule 7 can be applied (note that which item can be applied depends on the structure of the subgraph we are considering). There could be another vertex a such that a, x, u belong to one packed P_3 or another vertex a' such that a', z, u belong to one packed P_3. If no such vertices a and a' exist, then Item (1) of Reduction Rule 7 can be applied. Otherwise, one of the other items applies. To see more clearly that Reduction Rule 7 applies, we relabel the vertices as follows: $v \leftarrow z_1, u \leftarrow z_2, z \leftarrow v, w \leftarrow x, x \leftarrow u, y \leftarrow y, a \leftarrow w, a' \leftarrow w'$. All three subcases above contradict the assumption that no reduction rules can be applied in G'. It follows that there is no induced P_3 in $G'\triangle S'$ such that one edge of this P_3 is a non-packed edge in G' and the other edge of this P_3 is a packed edge in G'.

As a result, S' is a solution to the instance $(G', \mathcal{H}')$ of CDAMP. By Lemma 23, $(G, \mathcal{H})$ is a YES-instance. This concludes the proof for the lemma. □

The above lemma shows that there is a polynomial-time algorithm for the special instances of CDAMP with $\ell = 0$ that our reduction rules produces.

We can now prove that, without excess edits, CEMHMP can be solved in polynomial time.

THEOREM 3 (RESTATED). *CLUSTER EDITING ABOVE HALF-INTEGRAL MODIFICATION-DISJOINT P_3 PACKING can be solved in polynomial time when $\ell = 0$, that is, when no excess edits are allowed.*

PROOF. By Lemma 24, given an instance $(G, \mathcal{H})$ of CEMHMP, after applying Reduction Rules 1 to 7 exhaustively, we reduce it to an equivalent instance of 2-SAT in polynomial time. Then we can decide the 2-SAT instance by invoking the algorithm for 2-SAT. It is well-known that 2-SAT can be solved in polynomial time. This completes the proof for the theorem. □

6 CONCLUSIONS

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 Modification-Disjoint 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 combines settling the clustering around the vertices with large number of P_3s and applying reduction rules from Section 5 could be efficient.

ACKNOWLEDGMENTS

The authors acknowledge TU Wien Bibliothek for financial support through its Open Access Funding Programme. The authors would like to thank anonymous referees for their valuable comments and helpful suggestions.

REFERENCES

[1] Nir Ailon, Moses Charikar, and Alantha Newman. 2008. Aggregating inconsistent information: Ranking and clustering. *Journal of the ACM* 55, 5 (2008), 23:1–23:27. https://doi.org/10.1145/1411509.1411513

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

[3] Noga Alon, Konstantin Makarychev, Yury Makarychev, and Assaf Naor. 2006. Quadratic forms on graphs. *Inventiones Mathematicae* 163, 3 (2006), 499–522.

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

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

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

[7] Sebastian Böcker. 2012. A golden ratio parameterized algorithm for cluster editing. *Journal of Discrete Algorithms* 16 (2012), 79–89. https://doi.org/10.1016/j.jda.2012.04.005

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

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

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

[11] Sebastian Böcker and Peter Damaschke. 2011. Even faster parameterized cluster deletion and cluster editing. *Inform. Process. Lett.* 111, 14 (2011), 717–721. https://doi.org/10.1016/j.ipl.2011.05.003

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

[13] Nicolas Bousquet, Jean Daligault, and Stéphan Thomassé. 2018. Multicut is FPT. *SIAM J. Comput.* 47, 1 (2018), 166–207. https://doi.org/10.1137/140961808

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

[15] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. 2005. Clustering with qualitative information. *J. Comput. System Sci.* 71, 3 (2005), 360–383. https://doi.org/10.1016/j.jcss.2004.10.012

[16] Jianer Chen and Jie Meng. 2012. A $2k$ kernel for the cluster editing problem. *J. Comput. System Sci.* 78, 1 (2012), 211–220. https://doi.org/10.1016/j.jcss.2011.04.001

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

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

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

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

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

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

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

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

[25] Vincent Froese. 2018. *Fine-Grained Complexity Analysis of Some Combinatorial Data Science Problems*. Ph.D. Dissertation. Technische Universität Berlin. https://doi.org/10.14279/depositonce-7123

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

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

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

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

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

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

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

[33] Bart M. P. Jansen, Christian Schulz, and Hisao Tamaki. 2019. NII Shonan Meeting Report no. 144 Parameterized Graph Algorithms and Data Reduction. (2019). https://shonan.nii.ac.jp/docs/No.144.pdf

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

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

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

[37] Shaohua Li, Marcin Pilipczuk, and Manuel Sorge. 2021. Cluster editing parameterized above modification-disjoint P_3-packings. In *Proceedings of the 38th International Symposium on Theoretical Aspects of Computer Science (STACS 2021) (Leibniz International Proceedings in Informatics (LIPIcs))*, Markus Bläser and Benjamin Monmege (Eds.), Vol. 187. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 49:1–49:16. https://doi.org/10.4230/LIPIcs.STACS.2021.49

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

[39] Meena Mahajan and Venkatesh Raman. 1999. Parameterizing above guaranteed values: MaxSat and MaxCut. *Journal of Algorithms* 31, 2 (1999), 335–354. https://doi.org/10.1006/jagm.1998.0996

[40] Dániel Marx and Igor Razgon. 2014. Fixed-parameter tractability of multicut parameterized by the size of the cutset. *SIAM J. Comput.* 43, 2 (2014), 355–388. https://doi.org/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. 2011. clusterMaker: A multi-algorithm clustering plugin for cytoscape. *BMC Bioinformatics* 12, 1 (2011), 436. https://doi.org/10.1186/1471-2105-12-436

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

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

[44] René van Bevern, Vincent Froese, and Christian Komusiewicz. 2018. Parameterizing edge modification problems above lower bounds. *Theory of Computing Systems* 62, 3 (2018), 739–770. https://doi.org/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. 2010. Partitioning biological data with transitivity clustering. *Nature Methods* 7, 6 (2010), 419–420. https://doi.org/10.1038/nmeth0610-419

Received 17 September 2022; revised 19 July 2023; accepted 24 September 2023