Graph-Based Data Clustering with Overlaps

Michael R. Fellows[1⋆], Jiong Guo[2], Christian Komusiewicz[2⋆⋆], Rolf Niedermeier[2], and Johannes Uhlmann[2⋆⋆⋆]

[1] PC Research Unit, Office of DVC (Research), University of Newcastle, Callaghan, NSW 2308, Australia.
michael.fellows@newcastle.edu.au
[2] Institut für Informatik, Friedrich-Schiller-Universität Jena Ernst-Abbe-Platz 2, D-07743 Jena, Germany
{guo,ckomus,niedermr,uhlmann}@minet.uni-jena.de

Abstract. We introduce overlap cluster graph modification problems where, other than in most of previous work, the clusters of the goal graph may overlap. More precisely, the studied graph problems ask for a minimum number of edge modifications such that the resulting graph consists of clusters (that is, maximal cliques) that may overlap up to a certain amount specified by the overlap number s: In the case of s-vertex overlap, each vertex may be part of at most s maximal cliques; s-edge overlap is analogously defined in terms of edges. We provide a complete complexity dichotomy (polynomial-time solvable vs NP-complete) for the underlying edge modification problems, develop forbidden subgraph characterizations of "cluster graphs with overlaps", and study the parameterized complexity in terms of the number of allowed edge modifications, achieving both parameterized hardness (in case of unbounded s-values) as well as fixed-parameter tractability results (in case of constant s-values).

1 Introduction

Graph-based data clustering is an important tool in exploratory data analysis [23,27]. The applications range from bioinformatics [2,24] to image processing [26]. The formulation as a graph-theoretic problem relies on the notion of a *similarity graph*, where vertices represent data items and an edge between two vertices expresses high similarity between the corresponding data items. Then, the computational task is to group vertices into clusters, where a *cluster* is nothing but a dense subgraph (typically, a clique). Following Ben-Dor et al. [2], Shamir et al. [23] initiated a study of graph-based data clustering in terms of *graph modification* problems. More specifically, here the task is to modify (add or

⋆ Supported by the Australian Research Council. Work done while staying in Jena as a recipient of the Humboldt research award of the Alexander von Humboldt foundation, Bonn, Germany.
⋆⋆ Supported by a PhD fellowship of the Carl-Zeiss-Stiftung.
⋆⋆⋆ Supported by the DFG, research project PABI, NI 369/7.

delete) as few edges of an input graph as possible to obtain a *cluster graph*, that is, a *vertex-disjoint* union of cliques. Numerous recent publications build on this concept of cluster graphs, e.g., [4,7,9,11,13,22]. To uncover the *overlapping* community structure of complex networks in nature and society [19], however, the concept of cluster graphs so far misses to model that clusters may overlap. Hence, in this work we introduce a graph-theoretic relaxation of the concept of cluster graphs by allowing to a certain degree overlaps between the clusters (which are cliques). We distinguish between "vertex overlaps" and "edge overlaps" and provide a thorough study of the corresponding cluster graph modification problems.

The two core concepts we introduce are *s-vertex overlap* and *s-edge overlap*, where in the first case we demand that every vertex in the cluster graph is contained in at most s maximal cliques and in the second case we demand that every edge is contained in at most s maximal cliques. Clearly, 1-vertex overlap actually means that there is no overlap between the cliques (clusters). Based on these definitions, we study a number of edge modification problems (addition, deletion, editing) in terms of the two overlap concepts, generalizing and extending previous work that focussed on non-overlapping clusters.

Previous work. Perhaps the best studied cluster graph modification problem is the NP-hard Cluster Editing, where one asks for a minimum number of edges to add or delete in order to transform the input graph into a disjoint union of cliques. Cluster Editing has been intensively studied from a theoretical [1,3,9,11,13,22] as well as a practical side [4,7]. The major part of this work deals with the parameterized complexity of Cluster Editing, having led to efficient search-tree based [3,11] and polynomial-time kernelization [9,13,22] algorithms. One motivation of our work also draws from these intensive studies motivated by the practical relevance of Cluster Editing and related problems. As discussed before, however, Cluster Editing might use a sometimes too strict notion of cluster graphs by disallowing any overlap between the clusters. To the best of our knowledge, relaxed versions of Cluster Editing have been largely unexplored. The only approach studying overlapping cliques in the context of Cluster Editing we are aware of has been presented by Damaschke [6]. He investigated the Twin Graph Editing problem, where now the goal is to obtain a so-called *twin graph* (with a further parameter t specified as part of the input) with a minimum number k of edge modifications. The twin graph is a graph whose "critical clique graph" has at most t edges, where the critical clique graph is the representation of a graph that is obtained by keeping for each set of vertices with identical closed neighborhoods exactly one vertex. Roughly speaking, our model expresses a more local property of the target graph. The main result of Damaschke [6] is fixed-parameter tractability with respect to the combined parameter (t, k). We note that already for $s = 2$ our s-vertex overlap model includes graphs whose twin graphs can have an unbounded number t of edges. Hence, s is not a function of t. Moreover, we expect that for many real-world graphs the number k of necessary edge modifications is much smaller in our model than in the one of Damaschke.

Our results. We provide a thorough study of the computational complexity of clustering with vertex and edge overlaps, significantly extending previous work on Cluster Editing and closely related problems. In particular, in terms of the overlap number s, we provide a complete complexity dichotomy (polynomial-time solvable versus NP-complete) of the corresponding edge modification problems, most of them turning out to be NP-complete (also see Table 1 in Section 3). For instance, somewhat surprisingly, whereas Cluster Editing restricted to only allowing edge additions (also known as Cluster Addition or 1-Vertex Overlap Addition) is trivially solvable in polynomial time, 2-Vertex-Overlap Addition turns out to be NP-complete. Moreover, we also study the parameterized complexity of clustering with overlaps. On the negative side, we show W[1]-hardness results with respect to the parameter "number of edge modifications" in case of unbounded overlap number s. On the positive side, we prove that the problems become fixed-parameter tractable for every constant s. This result is based on forbidden subgraph characterizations of the underlying overlap cluster graphs, which may be of independent graph-theoretic interest. Indeed, it turns out that the "1-edge overlap cluster graphs" are exactly the diamond-free graphs. Finally, we develop polynomial-time data reduction rules for two special cases. More precisely, we show an $O(k^4)$-vertex problem kernel for 1-Edge Overlap Deletion and an $O(k^3)$-vertex problem kernel for 2-Vertex Overlap Deletion, where both times k denotes the number of allowed edge modifications. We conclude with a number of open problems for future work.

Preliminaries. Given a graph $G = (V, E)$, we use $V(G)$ to denote the vertex set of G and $E(G)$ to denote the edge set of G. Let $n := |V|$ and $m := |E|$. The (open) neighborhood $N(v)$ of a vertex v is the set of vertices that are adjacent to v, and the closed neighborhood $N[v] := N(v) \cup \{v\}$. We use $G[V']$ to denote the subgraph of G induced by $V' \subseteq V$, that is, $G[V'] := (V', \{\{u, v\} \mid u, v \in V', \{u, v\} \in E\})$. Moreover, $G - v := G[V \setminus \{v\}]$ for a vertex $v \in V$ and $G - e := (V, E \setminus \{u, v\})$ for an edge $e = \{u, v\}$. For two sets E and F let $E \Delta F := (E \setminus F) \cup (F \setminus E)$ (symmetric difference). Further, for a set X of vertices let $E_X := \{\{u, v\} \mid u, v \in X, u \neq v\}$ denote the set of all possible edges on X. Further, for a graph $G = (V, E)$ and a set $S \subseteq E_V$ let $G \Delta S := (V, E \Delta S)$ denote the graph that results by modifying G according to S. A set of pairwise adjacent vertices is called *clique*. A clique K is a *critical clique* if all its vertices have the same neighborhood and K cannot be extended. A *graph property* is defined as a nonempty proper subset of the set of graphs closed under graph isomorphism. A *hereditary* graph property is a property closed under the operation of taking induced subgraphs.

Formally, for a graph property π, the π Editing problem is defined as follows.

Input: A graph $G = (V, E)$ and an integer $k \geq 1$.
Question: Does there exist a set $S \subseteq V \times V$ with $|S| \leq k$ such that $G \Delta S$ has property π?

In this paper, we focus attention on π being either the s-vertex overlap property or the s-edge overlap property (see Definition 1 in Section 2). The set S is called a solution. Moreover, we say that the vertices that are incident to an edge in S are *affected* by S and that all other vertices are *non-affected*. In the corresponding π Deletion (or π Addition) problem, only edge deletion (or addition) is allowed.

Parameterized complexity is a two-dimensional framework for studying the computational complexity of problems [8,10,18]. One dimension is the input size n (as in classical complexity theory), and the other one is the *parameter* k (usually a positive integer). A problem is called *fixed-parameter tractable* (fpt) if it can be solved in $f(k) \cdot n^{O(1)}$ time, where f is a computable function only depending on k. This means that when solving a combinatorial problem that is fpt, the combinatorial explosion can be confined to the parameter.

A core tool in the development of fixed-parameter algorithms is polynomial-time preprocessing by *data reduction*. Here, the goal is for a given problem instance x with parameter k to transform it into a new instance x' with parameter k' such that the size of x' is upper-bounded by some function only depending on k, the instance (x, k) is a yes-instance iff (x', k') is a yes-instance, and $k' \leq k$. The reduced instance, which must be computable in polynomial time, is called a *problem kernel*, and the whole process is called *reduction to a problem kernel* or simply *kernelization*.

Downey and Fellows [8] developed a formal framework to show *fixed-parameter intractability* by means of *parameterized reductions*. A parameterized reduction from a parameterized language L to another parameterized language L' is a function that, given an instance (x, k), computes in $f(k) \cdot n^{O(1)}$ time an instance (x', k') (with k' only depending on k) such that $(x, k) \in L \Leftrightarrow (x', k') \in L'$. The basic complexity class for fixed-parameter intractability is called $W[1]$ and there is good reason to believe that $W[1]$-hard problems are not fpt [8,10,18].

Due to the lack of space, most proofs are deferred to the Appendix.

2 Forbidden Subgraph Characterization

In this section, we first introduce the two graph properties considered in this work. Then, we present induced forbidden subgraph characterizations for graphs with these properties.

Definition 1 (s-vertex-overlap property and s-edge-overlap property). *A graph $G = (V, E)$ has the* s-vertex-overlap property *(or* s-edge-overlap property*) if every vertex (or edge) of G is contained in at most s maximal cliques.*

Clearly, a graph having 1-vertex-overlap property consists of a vertex-disjoint union of cliques. See Figure 1 for a graph fulfilling the 2-vertex-overlap and the 1-edge-overlap property.

Given a graph and a non-negative integer s, we can decide in polynomial time whether G fulfills the s-vertex-overlap property using a clique enumeration algorithm with polynomial delay. For each $v \in V$, we enumerate the maximal

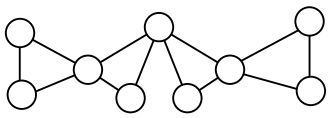

Fig. 1. An example graph with 2-vertex overlap and 1-edge-overlap property.

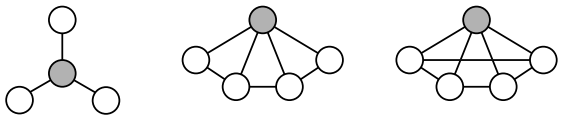

Fig. 2. Forbidden induced subgraphs for graphs with 2-vertex-overlap property. In every graph, the gray vertex is contained in at least three maximal cliques.

cliques in $G[N[v]]$. We abort the enumeration if we have found $s+1$ maximal cliques. Using for example a polynomial delay enumeration algorithm by Makino and Uno [16] that relies on matrix multiplication and enumerates cliques with delay $O(n^{2.376})$, the overall running time of this algorithm is $O(s \cdot n^{3.376})$. For the edge case, a similar approach applies. The only difference is that, here, we consider the common neighborhood of the endpoints of every edge, that is, $N[u] \cap N[v]$ for an edge $\{u, v\}$.

Theorem 1. *Given a graph and a non-negative integer s, it can be decided in $O(s \cdot n^{3.376})$ (or $O(s \cdot m \cdot n^{2.376})$) time whether G has the s-vertex-overlap (or s-edge-overlap) property.*

The next lemma provides the evidence for the existence of induced forbidden subgraph characterizations for graphs having the s-vertex-overlap or the s-edge-overlap property. It is well-known that hereditary graph properties can be characterized by a finite or infinite set of forbidden subgraphs [12].

Lemma 1. *The s-vertex-overlap property and the s-edge-overlap property are hereditary.*

Using Lemma 1, we can show that the forbidden subgraphs have size $O(s^2)$ and, thus, that for fixed s the number of forbidden induced subgraphs is finite. Furthermore, we have an algorithm that finds a forbidden induced subgraph in polynomial time.

Theorem 2. *Given a graph G that violates the s-vertex-overlap (or s-edge-overlap) property, one can find in $O(s \cdot n^{3.376} + s^2 \cdot n)$ (or $O(s \cdot m \cdot n^{2.376} + s^2 \cdot n)$) time a forbidden induced subgraph of size $O(s^2)$.*

See Figure 2 for the induced forbidden subgraphs for graphs with 2-vertex-overlap property. Observe that many important graph classes are contained in the class of graph with s-overlap property. In particular, it is easy to see that the diamond-free graphs are equivalent to graphs with 1-edge-overlap property,

Table 1. Classical computational complexity of graph-based data clustering with overlaps. Herein, "NPC" means that the respective problem is NP-complete and "P" means that the problem can be solved in polynomial time.

	s-vertex-overlap	s-edge-overlap
Editing	NPC for $s \geq 1$	NPC for $s \geq 1$
Deletion	NPC for $s \geq 1$	NPC for $s \geq 1$
Addition	P for $s = 1$, NPC for $s \geq 2$	P for $s = 1$, NPC for $s \geq 2$

as stated in the next lemma. A diamond is the graph that results by deleting one edge from a four-vertex clique. Diamond-free graphs, that is, graphs that contain no diamond as an induced subgraph, form a graph class studied for its own sake [25].

Lemma 2. *A graph G has the 1-edge-overlap property iff G is diamond-free.*

3 A Complexity Dichotomy with Respect to s

This section provides a complete picture of the computational complexity of the introduced problems. The results are summarized in Table 1.

Lemma 3 shows that if one of the problems is NP-hard for some $s \geq 1$, then it is NP-hard for every $s' \geq s$.

Lemma 3. *For $s \geq 1$, there is a polynomial-time many-one reduction from s-*Property Operation *to $(s+1)$-*Property Operation*, where* Property$\in$ { Vertex-Overlap, Edge-Overlap } *and* Operation $\in$ { Editing, Deletion, Addition }*.*

Since Cluster Editing and Cluster Deletion (equivalent to 1-Vertex-Overlap Editing and 1-Vertex-Overlap Deletion) are known to be NP-complete [15,23], we directly arrive at the following theorem.

Theorem 3. *s-*Vertex-Overlap Editing *and s-*Vertex-Overlap Deletion *are NP-complete for $s \geq 1$.*

1-Vertex-Overlap Addition is trivially polynomial-time solvable: one has to transform every connected component into a clique by adding the missing edges. In contrast, for $s \geq 2$, s-Vertex-Overlap Addition becomes NP-complete.

Theorem 4. *s-*Vertex-Overlap Addition *is NP-complete for $s \geq 2$.*

Proof. (Sketch) We present a polynomial-time many-one reduction from the NP-complete Maximum Edge Biclique problem [20] to 2-Vertex-Overlap Addition (2-VOA). Then, for $s \geq 2$, the NP-hardness follows directly from Lemma 3. The decision version of Maximum Edge Biclique is defined as follows: Given a bipartite graph $H = (U, W, F)$ and an integer $l \geq 0$, does H

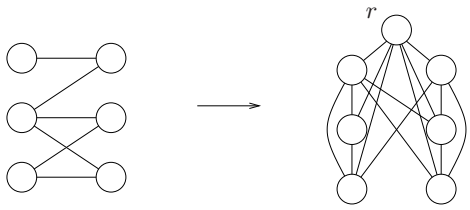

Fig. 3. Example for the reduction from Maximum Edge Biclique to 2-Vertex-Overlap Addition. The left graph is the instance of Maximum Edge Biclique; the right graph is the resulting 2-VOA graph.

contain a biclique with at least l edges? A biclique is a bipartite graph with all possible edges.

The reduction from Maximum Edge Biclique to 2-VOA works as follows: Given a bipartite graph $H = (U, W, F)$, we construct a graph $G = (V, E)$, where $V := U \cup W \cup \{r\}$ and $E := E_{\overline{F}} \cup E_r \cup E_U \cup E_W$. Herein,

- $E_{\overline{F}} := \{\{u, w\} \mid u \in U, w \in W\} \setminus F$,
- $E_X := \{\{x, x'\} \mid x, x' \in X, x' \neq x\}$ for $X \in \{U, W\}$, and
- $E_r := \{\{r, x\} \mid x \in U \cup V\}$.

That is, the graph $(U, V, E_{\overline{F}})$ is the bipartite complement of H, in G both U and V are cliques, and r is adjacent to all other vertices in G. See Figure 3 for an illustration of this construction. The correctness proof is deferred to the Appendix. □

Next, we consider the edge overlap case. First, observe that the reduction given in the proof of Theorem 4 can be easily modified to show the NP-hardness of 2-Edge-Overlap Addition: Simply replace the introduced vertex r by an edge e and connect both endpoints of e to all vertices in the given bipartite graph of the Maximum Edge Biclique. The correspondence between the solutions of both instances can be shown in complete analogy. Note that 1-Edge-Overlap Addition is trivially solvable in polynomial time, since there exists only one possibility to destroy a diamond by adding edges; by Lemma 2, diamonds are the only forbidden subgraph of graphs having 1-edge-overlap property.

Theorem 5. *s-Edge-Overlap Addition is NP-complete for $s \geq 2$.*

Finally, we can show that 1-Edge-Overlap Editing and Deletion are NP-complete by a reduction from Vertex Cover in cubic graphs. For $s > 1$, the NP-hardness follows directly by using Lemma 3.

Theorem 6. *s-Edge-Overlap Editing and s-Edge-Overlap Deletion are NP-complete for $s \geq 1$.*

4 Parameterized Complexity

Here, we consider the parameterized complexity of our overlap clustering problems. First, due to Theorem 2, we have a set of forbidden subgraphs for both properties whose size only depends on s. Thus, using a result of Cai [5], we can conclude that all three problems with overlap properties are fixed-parameter tractable with respect to the combined parameter (s, k).

Theorem 7. *π Editing, π Addition, and π Deletion problems with $\pi \in$ { s-Vertex-Overlap, s-Edge-Overlap } are fixed-parameter tractable with respect to the combined parameter (s, k).*

Next, we consider the parameterization with only k as the parameter. This means that s can have an unbounded value. To show W[1]-hardness, we develop a parameterized reduction from the W[1]-complete Set Packing problem [8].

Theorem 8. *s-Vertex(Edge)-Overlap Deletion(Editing) with unbounded s is W[1]-hard with respect to the parameter k.*

5 Two Kernelization Results for Edge Deletion

Not surprisingly, all nontrivial overlap clustering problems we study here seem to become significantly more demanding than clustering without overlaps. Hence, to start with, we subsequently present two kernelization results for the two most basic NP-hard clustering problems with nontrivial overlaps. Subsequently, we defer the correctness proofs of the considered data reduction rules to the Appendix. It is easy to see that they can be executed in polynomial time.

First, we present a kernelization for 1-edge Overlap Deletion. By Lemma 2, 1-edge Overlap Deletion is equivalent to the problem of destroying *diamonds* by at most k edge deletions. In the following, we introduce four data reduction rules for this problem and can show that an instance that is reduced with respect to all these rules has $O(k^4)$ vertices.

Rule 1. If there is a maximal clique C containing only edges which are not in any other maximal clique, then remove all edges of C.
Rule 2. If there is a matching of size greater than k in the complement graph of the graph that is induced by the common neighbors of the two endpoints of an edge e, then remove e, add e to the solution, and decrease the parameter k by one.
Rule 3. Remove all vertices that are not in any diamond.
Rule 4. If there is a critical clique with more than $k + 2$ vertices, then remove vertices until only $k + 2$ vertices remain.

Theorem 9. *1-Edge Overlap Deletion admits a problem kernel with $O(k^4)$ vertices which can be found in $O(m^3\sqrt{n} + m^2n^2)$ time.*

Proof. Let G denote an input graph reduced with respect to the four reduction rules. Partition the vertices of the graph G', resulting by applying a solution S to the input graph G, into two subsets, one set X containing the vertices that are endpoints of edges deleted by S and $Y := V \setminus X$. Further, construct for each edge $e \in S$ a set Y_e containing the vertices in Y that occur together with e in some diamonds. By Rule 3, $Y = \bigcup_{e \in S} Y_e$.

First, we show that for every maximal clique C in $G'[Y]$ it holds that $C \subseteq Y_e$ for an edge $e \in S$: In G, there is a maximal clique C' containing C and, by Rule 1, C' has an edge $e = \{u, v\}$ which is in two maximal cliques. If $e \in S$, then every vertex in C should be in Y_e; otherwise, there must be a vertex $w \notin C'$ that is adjacent to both u and v. One of the edges $\{u, w\}$ and $\{v, w\}$ must then be in S. Thus, every vertex in C must build a diamond with this deleted edge and C is contained in either $Y_{\{u,w\}}$ or $Y_{\{v,w\}}$.

Next, we prove that, for every edge $e = \{u, v\} \in S$, at most $4k$ maximal cliques of $G'[Y]$ are subsets of Y_e. Clearly, all vertices in Y_e must be adjacent to one of u and v. Let $N_{u,v}$ denote the common neighbors of u and v in Y_e. Obviously, $N_{u,v}$ is an independent set and, by Rule 2, $|N_{u,v}| \leq 2k$. Let $N_v := (N(v) \setminus N(u)) \cap Y_e$ and $N_u := (N(u) \setminus N(v)) \cap Y_e$. Since $N_{u,v}$ is an independent set, no vertex from $N_v \cup N_u$ can be adjacent to two vertices in $N_{u,v}$. Then, we can partition the vertices in $N_u \cup N_v$ into at most $4k$ subsets according to their adjacency to the vertices from $N_{u,v} = \{x_1, \ldots, x_l\}$ with $l \leq 2k$, every subset N_{u,x_i} (or $N(v, x_i)$) containing the vertices in $N(u) \cap N(x_i)$ (or $N(v) \cap N(x_i)$). It is easy to see that each of these subsets is a clique, since, otherwise, we would have some undestroyed diamond. With the same argument, there cannot be an edge between N_{u,x_i} and N_{u,x_j} with $i \neq j$. Moreover, the edges between N_{u,x_i} and N_{v,x_j}, if there are any, do not belong to the maximal cliques that are contained in Y_e. The reason is that the two endpoints of such an edge cannot have common neighbors in Y_e; otherwise, there would be some undestroyed diamond. Thus, we have at most $4k$ maximal cliques in $G'[Y]$ which are entirely contained in Y_e.

Finally, we show that if two vertices $u, v \in Y$ are contained in exactly the same sets of maximal cliques in Y, then they have the same neighborhood in G. Assume that this is not true. Then, u and v must have different neighborhoods in X. Let $w \in X$ be a neighbor of u but not of v. Since every two maximal cliques in Y can intersect in at most one vertex (due to the 1-edge overlap property), there can be only one maximal clique in Y containing both u and v. Assume that this maximal clique is contained in Y_e for an edge $e \in S$. Moreover, there must be another clique C in G containing w and u, but not v. By Rule 1, C must contain an edge which is part of two maximal cliques. This implies that the vertices w and u have to be in $Y_{e'}$ for an edge $e' \in S$ and $e \neq e'$. This means that there has to be a maximal clique in $Y_{e'}$ containing u but not v, contradicting that u and v are contained in the same sets of maximal cliques in Y.

Putting all arguments together, we can now show an upper bound for the number of vertices in the reduced instance. Clearly, $|X| \leq 2k$. To bound $|Y|$, note that we have at most k Y_e's. Each of them contains at most $4k$ maximal cliques

of $G'[Y]$. Since every maximal clique of $G'[Y]$ is contained in Y_e for one $e \in S$, we have altogether at most $4k^2$ maximal cliques in $G'[Y]$. It remains to show a size bound for each of these cliques. From the vertices in one clique K, only $4k^2$ of these can be in more than one maximal clique in Y, since every two such cliques overlap only in at most one vertex. The remaining vertices of K then have identical neighborhood. Thus, by Rule 4, K contains at most $4k^2 + k + 2$ vertices. This yields the required size bound on $|Y|$ and, therefore, on the reduced instance. □

Next, we provide a kernelization for 2-Vertex Overlap Deletion. In the following, we say that a vertex is *satisfied* if it is contained in at most two maximal cliques, and a clique is satisfied if all its vertices are satisfied. A clique is a neighbor of an other clique if they share some vertex or edge. Here, the polynomial-time executable data reduction rules read as follows.

Rule 1. If there is a critical clique K with more than $k+1$ vertices, then remove vertices from K until only $k+1$ vertices remain.

Rule 2. If there exists a satisfied maximal clique K and K's neighbors are all satisfied, then remove all edges in K which are not in other maximal cliques.

Rule 3 Let G be a graph reduced with respect to Rule 1. Let K be a maximal clique of G. Consider ℓ maximal cliques $K_1, \ldots, K_\ell$ fulfilling the following two conditions:
1.) $K \cap K_i \neq \emptyset$, $1 \leq i \leq \ell$, and
2.) all vertices in K_i are satisfied, $1 \leq i \leq \ell$.
If $\sum_{i=1}^{\ell} |K_i \cap K| \geq 3k + 4$, then delete all edges between $K_1 \cap K$ and $K \setminus K_1$.

Rule 4. Remove connected components that fulfill the 2-vertex overlap property.

Theorem 10. 2-Vertex Overlap Deletion *admits a problem kernel with* $O(k^3)$ *vertices.*

6 Conclusion

We first-time studied new cluster graph modification problems motivated by the practical relevance of clustering with overlaps [19]. Naturally, studying a so far unexplored set of problems, there remain many challenges for future work, here listing only a few of them. First, it seems conceivable that the forbidden subgraph characterizations we developed for cluster graphs with overlaps can be further refined. Second, it is desirable to improve the upper bounds on our fixed-parameter algorithms (including the kernelization results) and to further extend the list of fixed-parameter tractability results (in particular, achieving kernelization results for problems other than 1-Edge Overlap Deletion and 2-Vertex-Overlap Deletion). Third, corresponding experimental studies (as those undertaken for Cluster Editing, see [4,7]) are a natural next step. Fourth, the polynomial-time approximability of our problems remains unexplored. Fifth and finally, it seems promising to study overlaps also in the context of the more general correlation clustering problems (see [1]) or by relaxing the demand for (maximal) cliques in cluster graphs by the demand for some reasonably dense subgraphs.

References

1. N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1–3):89–113, 2004. 2, 10
2. A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *J. Comput. Biol.*, 6(3/4):281–292, 1999. 1
3. S. Böcker, S. Briesemeister, Q. B. A. Bui, and A. Truß. Going weighted: Parameterized algorithms for cluster editing. In *Proc. 2nd COCOA*, volume 5165 of *LNCS*, pages 1–12. Springer, 2008. 2
4. S. Böcker, S. Briesemeister, and G. W. Klau. Exact algorithms for cluster editing: Evaluation and experiments. In *Proc. 7th WEA*, volume 5038 of *LNCS*, pages 289–302. Springer, 2008. 2, 10
5. L. Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Inf. Process. Lett.*, 58(4):171–176, 1996. 8
6. P. Damaschke. Fixed-parameter enumerability of Cluster Editing and related problems. *Theory Comput. Syst.*, 2009. To appear. 2
7. F. Dehne, M. A. Langston, X. Luo, S. Pitre, P. Shaw, and Y. Zhang. The cluster editing problem: Implementations and experiments. In *Proc. 2nd IWPEC*, volume 4169 of *LNCS*, pages 13–24. Springer, 2006. 2, 10
8. R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999. 4, 8, 19
9. M. R. Fellows, M. A. Langston, F. A. Rosamond, and P. Shaw. Efficient parameterized preprocessing for Cluster Editing. In *Proc. 16th FCT*, volume 4639 of *LNCS*, pages 312–321. Springer, 2007. 2
10. J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006. 4
11. J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier. Graph-modeled data clustering: Exact algorithms for clique generation. *Theory Comput. Syst.*, 38(4):373–392, 2005. 2
12. D. L. Greenwell, R. L. Hemminger, and J. B. Klerlein. Forbidden subgraphs. In *Proc. 4th Southeastern Conf. on Comb., Graph Theory and Computing*, pages 389–394. Utilitas Mathematica, 1973. 5
13. J. Guo. A more effective linear kernelization for Cluster Editing. *Theor. Comput. Sci.*, 410(8-10):718–726, 2009. 2, 24
14. W. Hsu and T. Ma. Substitution decomposition on chordal graphs and applications. In *Proc. 2nd International Symposium on Algorithms*, volume 557 of *LNCS*, pages 52–60. Springer, 1991. 24
15. M. Křivánek and J. Morávek. NP-hard problems in hierarchical-tree clustering. *Acta Inform.*, 23(3):311–323, 1986. 6, 15
16. K. Makino and T. Uno. New algorithms for enumerating all maximal cliques. In *Proc. 9th SWAT*, volume 3111 of *LNCS*, pages 260–272. Springer, 2004. 5
17. S. Micali and V. V. Vazirani. An $O(\sqrt{(|V|)}|E|)$ algorithm for finding maximum matching in general graphs. In *Proc. 21st FOCS*, pages 17–27. IEEE, 1980. 23
18. R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006. 4
19. G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005. 2, 10
20. R. Peeters. The maximum edge biclique problem is NP-complete. *Discrete Appl. Math.*, 131(3):651–654, 2003. 6

21. S. Poljak. A note on stable sets and colorings of graphs. *Commentationes Mathematicae Universitatis Carolinae*, 15(2):307–309, 1974. 16
22. F. Protti, M. D. da Silva, and J. L. Szwarcfiter. Applying modular decomposition to parameterized cluster editing problems. *Theory of Computing Systems*, 44(1):91–104, 2009. 2
23. R. Shamir, R. Sharan, and D. Tsur. Cluster graph modification problems. *Discrete Appl. Math.*, 144(1–2):173–182, 2004. 1, 6, 15
24. R. Sharan, A. Maron-Katz, and R. Shamir. CLICK and EXPANDER: a system for clustering and visualizing gene expression data. *Bioinformatics*, 19(14):1787–1799, 2003. 1
25. M. Talmaciu and E. Nechita. Recognition algorithm for diamond-free graphs. *Informatica*, 18(3):457–462, 2007. 6
26. Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113, 1993. 1
27. R. Xu and D. Wunsch II. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005. 1

Appendix

Proofs of Section 2

Proof of Lemma 1.

Proof. We give here only a proof for the s-vertex-overlap property. The edge case can be shown analogously. To show that the s-vertex-overlap property is hereditary, it suffices to show the following.

> *Claim:* If $G = (V, E)$ has the s-vertex-overlap property, then so does $G - v$ for any $v \in V$.

Assume that G has the s-vertex-overlap property but there exists a vertex $v \in V$ such that $G - v$ does not have the s-vertex-overlap property. Then, there exists a vertex $w \in N_G(v)$ that is contained in at least $s + 1$ distinct maximal cliques of $G - v$, say $C_1, C_2, \ldots, C_{s+1}$. For every $1 \leq i \leq s + 1$, there exists a maximal clique K_i of G with $C_i \subseteq K_i$. Moreover, since in G there are at most s maximal cliques containing w, there exist i and j, $1 \leq i < j \leq s + 1$ such that $K_i = K_j$. However, since C_i and C_j are two distinct maximal cliques of $G - v$, there exists a vertex $u_i \in C_i$ and a vertex $u_j \in C_j$ such that $\{u_i, u_j\} \notin E$: a contradiction to the fact that K_i is a clique containing u_i and u_j. □

Proof of Theorem 2.

Proof. We prove here only the vertex case. Let G be a graph violating the s-vertex-overlap property and let v be a vertex that is contained in more than s maximal cliques. From Theorem 1, we know that such a vertex can be found in $O(s \cdot n^{3.376})$ time. Given $s+1$ maximal cliques $C_1, C_2, \ldots, C_{s+1}$ that contain v, we find a forbidden subgraph as follows. To "separate" two maximal cliques C_i and C_j, we need a vertex $v_1 \in C_i \setminus C_j$ and a vertex $v_2 \in C_j \setminus C_i$ with $\{v_1, v_2\} \notin E$. Clearly, such a vertex always exists since both C_i and C_j are maximal. To "separate" every pair of $s + 1$ maximal cliques we need at most $2\binom{s+1}{2}$ vertices. These vertices and v together induce a subgraph of size at most $(s + 1) \cdot s + 1$ and v is contained in at least $s + 1$ maximal cliques in this graph. Clearly, to find all these separating vertices, we need $O(s^2 \cdot n)$ time. □

Proof of Lemma 2.

Proof. Clearly, a diamond does not satisfy the 1-edge-overlap property. Thus, every 1-edge-overlap graph is diamond-free. Moreover, if a graph does not have the 1-edge-overlap property, then there must be an edge in more than one maximal cliques. These maximal cliques contain at least one induced diamond. □

Proofs of Section 3

Proof of Lemma 3.

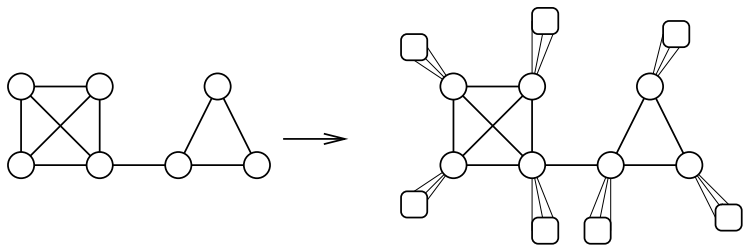

Fig. 4. Illustration of the reduction from s-Vertex-Overlap Editing to $(s+1)$-Vertex-Overlap Editing. Herein, every rectangular vertex represents a clique on $2k+2$ vertices.

Proof. The basic idea of all reductions is to add for every vertex v for vertex-overlap (respectively for every pair of distinct vertices u and v for edge-overlap), one "large" clique that intersects with the original s-Property Operation instance only in v (respectively u and v), resulting in the problem instance for $(s+1)$.

First, we focus on the case of vertex overlap. We show the reduction from s-Vertex-Overlap Editing (s-VOE) to $(s+1)$-Vertex-Overlap Editing ($(s+1)$-VOE). Moreover, we will observe that the same construction yields a reduction from s-Vertex-Overlap Deletion/Addition to $(s+1)$-Vertex-Overlap Deletion/Addition. Second, we will show that the reduction can be adapted to the case of edge overlap.

The reduction from s-VOE to $(s+1)$-VOE works as follows. Given an s-VOE instance consisting of a graph $G=(V,E)$ and an integer k, we construct an $(s+1)$-VOE instance consisting of a graph $H=(U,F)$ and an integer $k' := k$. For the construction of H, initially, we set $H := G$. Then, for every vertex $v \in V$, we add one clique C_v on $2k+2$ new vertices to H and we make v adjacent to every vertex in this clique. In other words, for every vertex we add one clique that intersects with G only in v. An illustration of this construction is given in Figure 4.

Next, we show the correctness of the reduction. On the one hand, given a solution S of size at most k for s-VOE with input graph G, in the graph that results by modifying G according to S, every vertex is contained in at most s maximal cliques. Hence, if we modify H according to S in the resulting graph, by construction of H every vertex is contained in at most $s+1$ maximal cliques.

On the other hand, let S' denote a solution of size at most $k' = k$ for $(s+1)$-VOE for H. Moreover, let H_{opt} denote the graph that results by modifying H according to S'. Since the size of S' is at most k, every vertex $v \in V$ is adjacent in H_{opt} to a non-empty set of non-affected vertices $N_v \subseteq C_v$ (since $|C_v| = 2k+2$ and there are at most $2k$ affected vertices in total). This implies that for every vertex $v \in V$ there exists one maximal clique C'_v containing v with $N_v \subseteq C'_v \subseteq C_v$. Consequently, every vertex $v \in V$ can be contained in at most s further maximal cliques, and, hence, v can be contained in at most s maximal cliques

in the induced subgraph $H_{\text{opt}}[V]$. That is, $H_{\text{opt}}[V]$ fulfills the s-vertex overlap property and $S := S' \cap E_V$ is a solution for s-VOE for G.

It is straightforward to verify that the given construction constitutes a reduction from s-Vertex-Overlap Deletion and s-Vertex-Overlap Addition to $(s+1)$-Vertex-Overlap Deletion and $(s+1)$-Vertex-Overlap Addition, respectively. Moreover, it is not hard to verify that adding for every pair of distinct vertices u and v (instead of every vertex) a clique $C_{u,v}$ with $2k+2$ vertices, which intersects with the original s-Edge-Overlap Operation instance only in u and v, yields a polynomial-time many-one reduction for the edge case. The correctness proof works in complete analogy. □

Proof of Theorem 3.

Proof. For $s = 1$ these problems correspond to Cluster Editing and Cluster Deletion, respectively, both being NP-complete [15,23]. Hence, according to Lemma 3 s-Vertex-Overlap Editing and s-Vertex-Overlap Deletion are NP-hard for $s \geq 1$. The containment in NP of these problem is obvious. □

Proof of the correctness of the reduction given in the proof of Theorem 4.

Proof. For the correctness of the reduction, we show the following.

> *Claim:* In graph H there is a biclique with at least l edges if and only if there exists a solution S with $|S| \leq |F| - l$ for 2-Vertex-Overlap Addition for G.

"⇒": Assume that H contains a biclique with at least l edges. Let $U' \subseteq U$ and $W' \subseteq W$ denote the vertices in such a biclique. Further, let F' denote the edges not contained in this biclique. That is, the removal of F' from H results in a graph that consists of the disjoint union of isolated vertices with one complete bipartite graph with at least l edges. Moreover, $|F'| \leq |F| - l$. Let G' denote the graph that results by adding the edges in F' to G.

Now, we argue that G' fulfills the 2-vertex-overlap property, and, hence, $S := F'$ is a solution for 2-VOA for G. To this end, observe that in G' any two vertices $u, u' \in U'$ have the same closed neighborhood. The same is true for any two vertices in W', $U \setminus U'$, and $W \setminus W'$, respectively. With this observation, it is not hard to see that in G' there are two maximal cliques, namely the clique $U \cup (W \setminus W') \cup \{r\}$ and the clique $W \cup (U \setminus U') \cup \{r\}$. Hence, every vertex in G' is contained in at most two maximal cliques. See Figure 5 for an example.
"⇐": Assume that there exists a solution S with $|S| \leq |F| - l$ for 2-VOA for G. Moreover, let G' denote the graph that results by adding the edges in S to G. First, note that, since S contains only edges not contained in G, all edges in S are between U and W, and, hence, $S \subseteq F$. We show that the graph H' that results by deleting the edges in S from H consists of isolated vertices and a complete bipartite graph with at least l edges. Assume towards a contradiction that H' is not of the claimed form. Then, either H' contains a connected component with more than one vertex that is not a biclique, or H' contains at least two

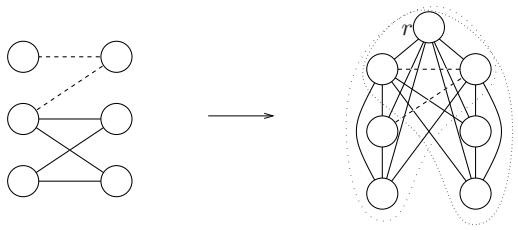

Fig. 5. The graph on the left contains a biclique with four edges (bold edges). Adding the edges not contained in this biclique (dashed edges) to the graph on the right results in a graph that contains two maximal cliques. The two maximal cliques are marked by the pointed lines.

connected components with more than one vertex. We distinguish both cases, and, in each case, derive a contradiction.

First, assume that H' contains a connected component with more than one vertex that is not a biclique. In this case, H' contains a P_4, an induced path on four vertices. Without loss of generality, we can assume that the first and the third vertex, say u and u', are from U and the second and fourth vertex, say w and w', are form W. Since $G'[U \cup W]$ is the (non-bipartite) complement graph of H', in G' we have an induced P_4 $(\{u', u\}, \{u, w'\}, \{w', w\})$. Since r is adjacent to all vertices, this implies that $G'[\{u, u', w, w', r\}]$ is isomorphic to the second graph shown in Figure 2: a contradiction to the fact that G' fulfills the 2-vertex-overlap property.

Second, assume that H' contains at least two connected components with more than one vertex. Let $e = \{u, w\}$ and $e' = \{u', w'\}$ with $u, u' \in U$ and $w, w' \in W$ be two edges from different connected components of H'. This implies that in G' we have that $\{\{u, u'\}, \{u', v\}, \{v, v'\}, \{v', u\}\}$ forms an induced cycle of length four. Further, since r is adjacent to all vertices, $G'[\{u, u', v, v', r\}]$ is isomorphic to the third graph shown in Figure 2: a contradiction to the fact that G' fulfills the 2-vertex-overlap property. □

Proof of Theorem 6.

Proof. First, for the ease of presentation, we show the NP-hardness of 1-Edge-Overlap Deletion. Then, we explain how the given reduction can be extended to work for 1-Edge-Overlap Editing. Then, for $s > 1$, the NP-hardness follows directly from Lemma 3.

We show the NP-hardness of 1-Edge-Overlap Deletion by a reduction from the NP-complete problem Vertex Cover restricted to graphs containing no triangles [21], where one is given a triangle-free graph $G = (V, E)$ and an integer $k \geq 0$, and asks for a set $V' \subseteq V$ with $|V'| \leq k$ such that every edge in E has at least one endpoint in V'. Note that 1-Edge-Overlap Deletion is the same as making a graph diamond-free by edge deletions (Lemma 2).

Given a triangle-free graph $G = (V, E)$ and an integer $k \geq 0$, we construct a graph $H = (W, F)$ as follows. Initially, we set $H := G$. Then, we add a

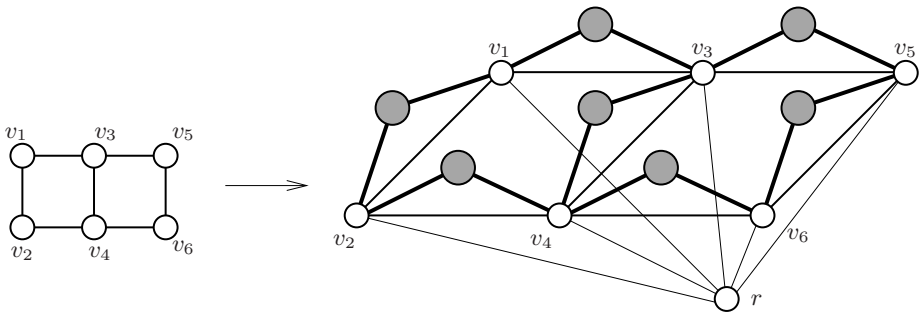

Fig. 6. Example of the reduction from Vertex Cover in triangle-free graphs to 1-Edge-Overlap Deletion. Herein, the gray vertices represent cliques on $3k+1$ vertices and a bold edge between a gray vertex and a vertex v_i represents all possible edges between v_i and the vertices in the respective clique.

new vertex r to G and the edges $\{r, v\}$ for every $v \in V$. Finally, for every edge $e = \{u, v\} \in E$, we add a clique $C_{u,v}$ on $3k + 1$ new vertices to H and the edges $\{u, x\}$ and $\{v, x\}$ for every $x \in C_{u,v}$. An illustration of the construction is given in Figure 6.

For the correctness of the reduction we show the following.

> *Claim:* Graph G has a vertex cover V' of size at most k if and only if there exists a solution S of size at most k for 1-Edge-Overlap Deletion for graph H.

"⇒:" Given a vertex cover $V' \subseteq V$ for G, we show that $S := \{\{r, v\} \mid v \in V'\}$ is a solution for 1-Edge-Overlap Deletion for H. This proof crucially uses the fact that G is triangle-free.

Let H' denote the graph that results by deleting the edges in S from H. Note that, since S only contains edges incident to r and since V' is a vertex cover, for every edge $\{u, v\} \in E$ the vertices $C_{u,v} \cup \{u, v\}$ form a maximal clique in H'.

In H (and hence in H') there are three types of edges, namely edges with at least one endpoint in $C_{u,v}$ for some $\{u, v\} \in E$, edges $\{u, v\} \in E$, and edges of the form $\{r, u\}$ with $u \in V$. We show that, for each edge type, every possible edge of this type is contained in at most one maximal clique.

First, let $\{x, y\}$ be an edge with $x \in C_{u,v}$ for some $\{u, v\} \in E$ and $y \in \{u, v\} \cup C_{u,v}$. By the above observation that $C_{u,v} \cup \{u, v\}$ is a maximal clique in H', it follows directly that $\{x, y\}$ is contained in exactly one maximal clique, namely $C_{u,v} \cup \{u, v\}$.

Second, since for every edge $\{u, v\} \in E$ the vertices u and v do not have common neighbors in V (because G is triangle-free) and since by construction at least one of the edges $\{r, u\}$ and $\{r, v\}$ is deleted, the only common neighbors of u and v are the vertices in the clique $C_{u,v}$. Hence, $\{u, v\}$ is contained in exactly one maximal clique, namely $C_{u,v} \cup \{u, v\}$.

Third, consider an edge $\{r, v\} \in E(H')$. Since r is adjacent in H exactly to the vertices in V, the common neighbors of r and v in H are exactly the neighbors of v in G. Moreover, since for every edge $\{r, v\} \in E(H')$ we know that v is not in the vertex cover V', we also know that all neighbors of v in G must be contained in V'. Hence, we can conclude that r and v have no common neighbors in H'. Thus, $\{r, v\}$ is the only maximal clique containing v.

"$\Leftarrow$:" We show that if we have a solution S of size at most k for 1-Edge-Overlap Deletion for H, then for every edge $\{u, v\} \in E$ either the edge $\{r, u\}$ or the edge $\{r, v\}$ is contained in S. Hence, the set $\{v \in V \mid \{r, v\} \in S\}$ is a vertex cover for G.

In the following, let H_{opt} denote the graph that results by modifying H according to S. Since S is a solution, H_{opt} is diamond-free. Assume towards a contradiction that for some edge $\{u, v\} \in E$ neither $\{r, u\} \in S$ nor $\{r, v\} \in S$. Note that in H_{opt} the set $C_{u,v}$ contains at least $3k + 1 - 2k = k + 1$ non-affected vertices. Let x and y denote two distinct non-affected vertices from $C_{u,v}$. Note that in H the vertices x, u, v, r induce a diamond. Since neither $\{r, u\} \in S$ nor $\{r, v\} \in S$ and since x is non-affected, the solution S must contain the edge $\{u, v\}$. However, this implies that x, y, u, v induce a diamond in H_{opt}: Since in H the vertices x, y, u, v form a clique and x and y are non-affected by S, there exist all edges between the vertices x, y, u, v except for the edge $\{u, v\}$ which is contained in S. This is a contradiction to the fact that H_{opt} is diamond-free. This completes the proof of the NP-hardness of 1-Edge-Overlap Deletion.

Next, we argue that the presented reduction can be extended to work for 1-Edge-Overlap Editing. Given a graph without triangles, first, we build a graph H as described above. Then, for every pair of distinct and non-adjacent vertices u, v in H we add a gadget to prevent the addition of $\{u, v\}$. More specifically, for every pair of distinct and non-adjacent vertices u, v, we add $2k+2$ vertices $P_{uv} := \{p_1^{uv}, \ldots, p_{2k+2}^{uv}\}$ and the edges $\{u, p_i^{uv}\}$, $\{v, p_i^{uv}\}$, $1 \leq i \leq 2k + 2$. Let H' be the resulting graph. We claim that this construction provides a reduction from Vertex Cover in triangle-free graphs to 1-Edge-Overlap Editing. For the correctness of the reduction, observe that every solution S' of 1-Edge-Overlap Editing of H' with $|S'| \leq k$ does not contain any edge insertion between two distinct vertices u and v that are non-adjacent in H. This can be seen as follows. Let H'' be the graph modified according to S'. Assume towards a contradiction that $\{u, v\} \in E(H'')$ but $\{u, v\} \notin E$. Since $|S'| \leq k$ there are at most $2k$ affected vertices in H'', and, as a consequence, two vertices in P_{uv} are non-affected. Hence, the edge $\{u, v\}$ is contained in at least two maximal cliques in H''; a contradiction to the fact that S' is a solution for H'. With this observation at hand, the correctness proof works in complete analogy to the deletion case. □

Proof of Section 4

Proof of Theorem 8.

Proof. To show the W[1]-hardness of s-Vertex-Overlap Deletion, we describe a parameterized reduction from the W[1]-complete Set Packing problem [8], which is defined as follows:

> Input: A family of sets $\mathcal{S} = \{S_1, \ldots, S_n\}$ over a universe $U = \{1, \ldots, m\}$ and a nonnegative integer $k \leq n$.
> Question: Is there a set $\mathcal{S}' \subseteq \mathcal{S}$ such that $|\mathcal{S}'| \geq k$ and $\forall S_i, S_j \in \mathcal{S}' : S_i \cap S_j = \emptyset$?

Now, consider an instance $I = (\mathcal{S}, k)$ of Set Packing. Without loss of generality we can assume that $k < m$ and $k < n$. We construct an s-Vertex-Overlap Deletion instance $(G = (V, E), k)$ as follows. The vertex set V is comprised of six subsets V_U, V_S, V_X, V_Y, V_C, and V_P. The set $V_U := \{u_1, \ldots, u_m\}$ contains one vertex for each element $i \in U$. The set $V_S := \{s_1, \ldots, s_n\}$ contains one vertex for each $S_i \in \mathcal{S}$. The set $V_X := \{x_1, \ldots, x_k\}$ contains k vertices; the set $V_Y := \{y_1, \ldots, y_{2k+1}\}$ contains $2k + 1$ vertices; together they serve as a "selection" gadget. The set V_C contains vertices that are part of some "shielding" cliques. With these cliques, we can enforce that some edges will never be edited. Finally, the set V_P contains "padding" cliques that are used to increase the number of maximal cliques for certain vertices. First, we describe the construction of the graph $G[V_U \cup V_S \cup V_X \cup V_Y]$, then we describe how the additional cliques are added to this graph. For a vertex $s_i \in V_S$ corresponding to set S_i and a vertex $u_j \in V_U$ corresponding to an element $j \in U$, we add the edge $\{u_j, s_i\}$ if $j \in S_i$. Furthermore, we connect each $x_i \in V_X$ by edges to all vertices in $V_U \cup V_S$. Finally, we make V_Y a clique and connect each $y \in Y$ to all vertices in $V_X \cup V_S$. This concludes the construction of $G[V_U \cup V_S \cup V_X \cup V_Y]$. An example is shown in Fig. 7a.

Next, the shielding cliques are added. For each $x_i \in V_X$ we add the vertex set $C^{x_i} := \{c_1^{x_i}, \ldots, c_{k+1}^{x_i}\}$ to V_C. Furthermore, we make $C^{x_i} \cup \{x_i\} \cup V_Y$ a clique. This construction ensures that deleting the edge $\{x_i, s_j\}$ for a vertex $x_i \in V_X$ and $s_j \in V_S$ decreases the number of maximal cliques that contain x_i by one. Then, for each edge $\{u_i, s_j\}$, and for each $x_l \in V_x$, we add a vertex set $C^{u_i,s_j,x_l} := \{c_1^{u_i,s_j,x_l}, \ldots, c_{k+1}^{u_i,s_j,x_l}\}$ to V_C and we make $\{u_i, s_j, x_l\} \cup C^{u_i,s_j,x_l}$ a clique. This clique has the following purpose: if we delete an edge $\{s_j, x_l\}$, then we increase the number of maximal cliques that contain u_i. Altogether, the shielding cliques ensure that in order to decrease the number of maximal cliques for a vertex $x_i \in V_X$ with at most k edge deletions, we can only delete edges between x_i and V_S. An example of these shielding cliques is shown in Fig. 7b.

Before we describe how the padding cliques are added, we compute the number of maximal cliques in $G[V_U \cup V_S \cup V_X \cup V_Y \cup V_C]$ that each vertex $v \in V_U \cup V_X$ is contained in. To denote this number for some vertex v, we write $\#(v)$. Each vertex $u_i \in V_U$ is contained in exactly $\#(u_i) = |N(u_i) \cap V_S| \cdot k \leq n \cdot k$ maximal cliques.

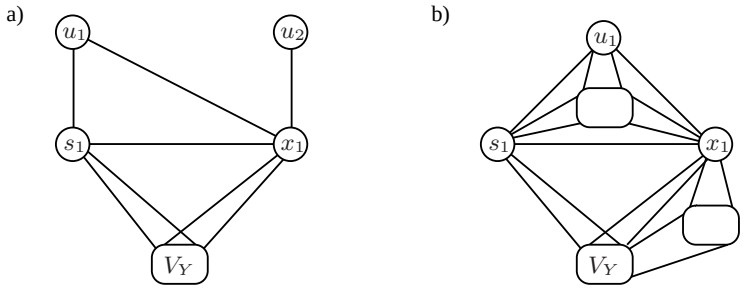

Fig. 7. Parts of the graph constructed in the reduction from Set Packing to s-Vertex Overlap Deletion. Rectangles depict cliques of size at least $k+1$. a) A subgraph containing $u_1, u_2 \in V_U$, $s_1 \in V_S$, $x_1 \in V_X$, and V_Y. Edges are drawn from $s_i \in V_S$ to $u_j \in V_U$ if $j \in S_i$. Here, $1 \in S_1$ but $2 \notin S_1$.) Shielding cliques are added for each triangle in $G[V_U \cup V_X \cup V_S]$ and between x_i and V_Y for all $x_i \in V_X$.

Each vertex $x_i \in V_x$ is contained in $\#(x_i) = \sum_{s_j \in V_S}(|N(s_j) \cap V_U| + 1) + 1 \leq n \cdot (m+1) + 1$ maximal cliques.

We add padding cliques of size $k+1$ that are "attached" to the vertices in V_U and V_X as follows. For each $x_i \in V_X$, we add $n \cdot (m+1) + 1 - \#(x_i)$ size-$(k+1)$ vertex sets $C_l^{x_i}$ to V_P, where $1 \leq l \leq n \cdot (m+1) + 1 - \#(x_i)$. For each $C_{x_i}^l$, we make $\{x_i\} \cup C_l^{x_i}$ a clique. Then, for each $u_i \in V_U$, we add $n \cdot (m+1) - 1 - \#(u_i)$ size-$k+1$ vertex sets $C_l^{u_i}$ to V_P, where $1 \leq l \leq n \cdot (m+1) - \#(u_i)$. Note that since $k < m$ and $k < n$, the number of added cliques is positive. Again, for each $C_l^{u_i}$, we make $\{u_i\} \cup C_l^{u_i}$ a clique.

This concludes the construction of G. Note that in G

- each vertex $u_i \in V_U$ is contained in exactly $n \cdot (m+1) - 1$ maximal cliques,
- each vertex $x_i \in V_X$ is contained in exactly $n \cdot (m+1) + 1$ maximal cliques,
- each vertex $y_i \in V_Y$ is contained in exactly $n \cdot k + k < n \cdot (m+1)$ maximal cliques,
- each vertex $s_i \in V_S$ is contained in exactly $k \cdot (|N(s_1) \cap V_U| + 1) < n \cdot (m+1)$ maximal cliques, and
- each vertex $v \in V_P \cup V_C$ is contained in exactly one maximal clique.

Finally, we set $s := n \cdot (m+1)$. Clearly, the construction can be performed in polynomial time. Hence, for showing W[1]-hardness of s-Vertex-Overlap Deletion parameterized by k, it is sufficient to prove the following

> Claim: (I, k) is a yes-instance for Set Packing iff (G, k) is a yes-instance for $(n \cdot (m+1))$-Vertex-Overlap Deletion.

"$\Rightarrow$" Let $\mathcal{S}'$ be a size-k solution of Set Packing, and assume without loss of generality, that $\mathcal{S}' = \{S_1, \ldots, S_k\}$.We construct a solution S' of $(n \cdot (m+1))$-Vertex-Overlap Deletion as follows: we delete the edge $\{x_i, s_i\}$ for each $1 \leq$

$i \leq k$. To see that this set of edge deletions is a solution consider the following. Let G' be the graph that is created by applying S' to G. Clearly, we need to consider only vertices $v \in V$ such that there is at least one edge that has been removed from $G[N[v]]$, since for the other vertices the number of maximal cliques containing them has not changed.

First, since $\mathcal{S}'$ is a solution for Set Packing, for each $u_i \in V_U$, there is at most one $s_j \in N[u_i]$ with $1 \leq j \leq k$. Hence, at most one edge in $G[N[u_i]]$ has been removed. Let $\{s_j, x_j\}$ denote such an edge. There is one maximal clique in G that contains u_i, s_j, and x_j, namely, $\{u_i, s_j, x_j\} \cup C^{u_i,s_j,x_j}$. After the deletion of $\{s_j, x_j\}$, we have two maximal cliques that contain the vertices from C^{u_i,s_j,x_j}: $C^{u_i,s_j,x_j} \cup \{u_i, s_j\}$, and $C^{u_i,s_j,x_j} \cup \{u_i, x_j\}$. Hence, for each $u_i \in V_U$ the number of maximal cliques has increased by at most one. Therefore, each $u_i \in V_U$ is in at most $n \cdot (m+1)$ maximal cliques.

Next, for each vertex $x_i \in V_x$, the number of maximal cliques has decreased by one. This can be seen as follows. For each $x_i \in V_X$, we have removed only the edge $\{s_i, x_i\}$ in $G[N[x_i]]$. This means, that the number of maximal cliques that contain x_i cannot increase. Furthermore, with removing $\{s_i, x_i\}$ we destroy the maximal clique $\{s_i, x_i\} \cup V_Y$, and the "new" clique $\{x_i\} \cup V_Y$ is a subset of the existing shielding clique $\{x_i\} \cup V_Y \cup C^{x_i}$. The number of maximal cliques that contain x_i has thus decreased by one. Hence, each $x_i \in V_x$ is now in exactly $n \cdot (m+1)$ maximal cliques. A similar argument can be used to show that for each vertex in V_Y the number of maximal cliques that it is contained in has not increased.

For each vertex $s_i \in V_S$, the number of maximal cliques that contain s_i has not increased, since if an edge in $G[N[s_i]]$ has been deleted, it is the edge $\{s_i, x_i\}$. Since this edge is incident to s_i, its deletion does not increase the number of maximal cliques that contain s_i. Hence, each $s_i \in V_S$ is still in at most $n \cdot (m+1)$ maximal cliques.

Finally, each $v \in V_P \cup V_C$ is contained in at most two maximal cliques in G', since each $v \in V_P \cup V_C$ is contained in at most one maximal clique in G, and at most one edge in $G[N[v]]$ has been deleted.

Altogether, each vertex in G' is contained in at most $n \cdot (m+1)$ maximal cliques and S' is thus a size-k solution for $n \cdot (m+1)$-Vertex-Overlap Deletion.

"$\Leftarrow$"Let S' be a size-k solution for (G, k) and $G' = G \Delta S'$. First, we show that for each $x_i \in V_X$, at least one edge between x_i and V_S must be deleted. To see this, consider the following. There are $n \cdot (m+1) + 1$ maximal cliques that contain x_i. Hence, the number of cliques containing x_i must be reduced by at least one. The vertex x_i is contained in two types of maximal cliques: those that contain a shielding or a padding clique and those that contain x_i, V_Y, and one vertex $s_j \in V_S$.

First, note that with k edge deletions, we cannot decrease the number of maximal cliques that contain x_i and a shielding or padding clique. This can be seen as follows. The shielding and padding cliques are pairwise vertex-disjoint

in G and with k edge deletions, there remains for each shielding or padding clique at least one vertex that is adjacent to x_i in G'.

Next, consider the cliques that contain x_i, V_Y, and one vertex from V_S. In G, there are exactly $|V_S|$ cliques of this type. We now show that in G' there are at least $|V_S|$ cliques of this type. Consider some $s_j \in V_S$. Since V_Y has size $2k+1$, there is in G' at least one vertex $y \in V_Y$ that is adjacent to x_i and s_j. Hence, for each s_j there is at least one maximal clique that contains x_i, s_j, and y. Since V_S is an independent set, this means that there are at least $|V_S|$ maximal cliques of this type in G'. Hence, the number of cliques that contain x_i has not decreased in the case that we do not delete any edge between x_i and V_S. Therefore, each edge in a size-k solution S' contains exactly one vertex from V_S and can thus be interpreted as a subset of the input set $\mathcal{S}$ of the Set Packing instance. To see that this set is indeed a solution of Set Packing, consider the following: for each $u \in V_U$, there can be at most one pair of vertices $s_i \in V_S \cap N(u)$, $x_j \in V_X$ such that $\{s_i, x_j\}$ has been deleted by S'. Otherwise, u is contained in too many maximal cliques. In particular, each $s_i \in V_S$ is incident to at most one edge of S'. This corresponds exactly to a size-k subset $\mathcal{S}' \subseteq \mathcal{S}$ such that no $u \in U$ is contained in more than one element of $\mathcal{S}'$.

Altogether, we have shown the equivalence between the solutions of s-Vertex-Overlap Deletion and Set Packing. This implies that s-Vertex-Overlap Deletion with unbounded s is W[1]-hard, when parameterized by k. Using "tricks" such as increasing the size of the shielding cliques etc. one can modify the reduction in order to show W[1]-hardness with respect to the parameter k also for s-Vertex-Overlap Editing, s-Edge-Overlap Deletion, and s-Edge-Overlap Editing; we omit the details of the proofs. □

Proofs of Section 5

Kernelization for 1-Edge-Overlap Deletion

Lemma 4. *Rule 1 is correct and can be carried out in $O(m^2)$ time.*

Proof. Let G denote the input instance and G' be the graph resulting by applying Rule 1 to a maximal clique C in G. To show the correctness of Rule 1, we prove that (G, k) is a yes-instance iff (G', k) is yes-instance. For the "$\Rightarrow$"-direction, let S denote an optimal solution for G. Then S contains no edge from C. To see this, observe that the only possible way to create a diamond containing some edge from C is to delete edges from C. However, since all edges of C are not in a diamond in G, an optimal solution will never delete them. This means that C remains a maximal clique in $G \Delta S$ and no two vertices of C have common neighbors outside of C. Thus, removing the edges of C from $G \Delta S$ does not destroy the 1-edge-overlap property and S is also a solution for G'. For the other direction, observe that after applying Rule 1 to C, no two vertices of C have common neighbors in G'. Therefore, we can add the edges of C to the graph $G'' := G' \Delta S$, where S is an optimal solution for G', without destroying the 1-edge-overlap property of G''. Therefore, the rule is correct. To check the

applicability of Rule 1, we go through all edges and check whether this edge is in only one maximal clique. If so, we check further whether all edges of this clique are only contained in this clique. Clearly, this is doable in $O(m^2)$ time. □

Lemma 5. *Rule 2 is correct and can be carried out in $O(m^2\sqrt{n})$ time.*

Proof. Let G_e denote the subgraph of the input graph G induced by the common neighbors of the two endpoints of an edge e. Every edge e' in the complement graph $\overline{G_e}$ of G_e implies a diamond in G consisting of the endpoints of e' and the endpoints of e. Therefore, every matching of $\overline{G_e}$ corresponds to a set of diamonds in G, which pairwise only have e in common. This means that to destroy all these diamonds, we either delete e or delete one edge for every diamond. Then, a matching greater than k forces the deletion of e. Since a maximum matching can be computed in $O(m\sqrt{n})$ time [17], the applicability of Rule 2 can thus be checked in $O(m^2\sqrt{n})$ time. □

Lemma 6. *Rule 3 is correct and can be carried out in $O(n^2m)$ time.*

Proof. The correctness of Rule 3 follows directly from the claim that for a vertex v that is not in any diamond, every optimal solution does not delete edge incident to v. It remains to show the claim. Consider an arbitrary optimal solution S for an input graph G. Clearly, there is an ordering of the edges in S such that e_1 is in a diamond in $G_0 := G$, e_2 is in a diamond in $G_1 := G - e_1$, and so on. Suppose that the claim is not true. Then, there is some edges in S incident to v. This means that there is a minimal index i such that G_i has a diamond containing v but G_j not for all $j < i$. Therefore, deleting edge $e_{i-1} = \{u, w\}$ creates this diamond containing v. Since i is minimal, e_{i-1} is not incident to v. By the fact that the deletion of e_{i-1} creates this diamond, v has to be adjacent to both u and w. Moreover, there must be a diamond in G_{i-2} leading to the deletion of e_{i-1}. Let x and y denote the other two vertices of this diamond in G_{i-2}. Both x and y have to be adjacent to v; otherwise, we would have a diamond containing v in G_{i-2}, contradicting the minimality of i. However, the edges $\{v, x\}$ and $\{v, y\}$ then create a diamond which contains u, v, x, y or v, w, x, y, again contradiction to the minimality of i.

The running time follows directly from the fact that all diamonds containing a vertex v can be found by finding a missing edge in the intersection of $N(v)$ and the neighborhoods of the vertices not in $N(v)$ or by finding an induced path of three vertices in $N(v)$. Both can be done in $O(nm)$ time. □

Lemma 7. *Rule 4 is correct and can be carried out in $O(n + m)$ time.*

Proof. To show the claim, it suffices to prove the claim that, for every critical clique C with more than $k + 1$ vertices, every optimal solution does not delete edge incident to the vertices of C. Suppose that the claim is not true. Let S denote an optimal solution deleting the edge $\{u, v\}$ with $u \in C$. Let G' denote the graph resulting by applying S to the input graph G and G'' be the graph resulting by adding $\{u, v\}$ to G'. Then, since S is optimal, G'' must contain a diamond containing $\{u, v\}$; let x, y be the other two vertices of this diamond.

Since $|C| > k+1$ and $|S| \leq k$, there must be a vertex w in C which, in G'', has the same neighborhood as u. This implies that G'' and, thus, G' contains a diamond consisting of v, w, x, y, contradicting that S is a solution.

The running time of Rule 4 follows from the fact that all critical cliques of a graph can be computed in $O(n+m)$ time [14]. □

Kernelization for 2-Vertex-Overlap Deletion

Correctness of Rule 1. The following lemma says that all vertices that have the same closed neighborhood in the input graph also have the same closed neighborhood in the resulting solution, and, hence, are contained in the same maximal cliques in the solution. The proof works in analogy to a proof of [13, Lemma 2] and is omitted.

Lemma 8. *Let K denote a critical clique of $G = (V, E)$. Then, there exists a minimum solution $S \subseteq E$ such that K is part of a critical clique in $G \Delta S$.*

According to Lemma 8, if one deletes an edge incident to a vertex in a critical clique, then one has to delete an edge for every vertex in a critical clique. Hence, if there exists a critical clique of size greater than $k+1$, then we need only $k+1$ of these vertices. Hence, Rule 1 os correct. Note that all critical cliques of a graph can be found in linear time [14]. Hence, Rule 1 can be carried out in $O(n+m)$ time.

Correctness of Rule 2. To prove the correctness of Rule 2, we need the following two lemmas.

Lemma 9. *Let K and K' denote two maximal cliques of a graph G with $K \cap K' \neq \emptyset$. If all vertices in $K \cap K'$ are satisfied, then it holds that*

1. *K' is vertex-disjoint to all other maximal cliques intersecting with K, and*
2. *there is no edge between $K' \setminus K$ and $K \setminus K'$.*

Proof. Assume towards a contradiction that there exists a maximal clique K'' with $K \cap K'' \neq \emptyset$ and $K' \cap K'' \neq \emptyset$. Let $x \in K' \cap K''$. Clearly, $x \notin K$ since, otherwise, x is contained in three maximal cliques. Let $u \in K \cap K'$ and $v \in K'' \cap K$. Clearly, $\{u, v, x\}$ forms a clique. Let X denote an arbitrary maximal clique containing $\{u, v, x\}$. Note that X is neither K (since $x \in K' \setminus K$) nor K' (since $v \in K'' \setminus K'$) nor K'' (since $u \in K' \setminus K''$). Hence, u is contained in at least three maximal cliques, a contradiction to the fact that all vertices in $K \cap K'$ are satisfied. □

Lemma 10. *Let K denote a satisfied maximal clique. If there exists a vertex $v \in K$ that is contained in exactly one maximal clique (namely K), then, for every minimum solution S, v is contained in exactly one maximal clique in $G \Delta S$.*

Proof. Let S' denote a minimum solution for G. In the following, we always assume that every critical clique of G is also (part of) a critical clique in $G_{S'} := G \Delta S'$ (see Lemma 8). Assume towards a contradiction that v is contained in two maximal cliques in $G_{S'}$.

First, we describe the structure of $N_G[v]$ in G. Second, we describe how the structure of $N_G[v]$ is changed when deleting the edges in S. Finally, we show that we can undo some edge deletions in $G_{S'}$, yielding a contradiction to the fact that S' is minimal.

Consider the closed neighborhood of v in G. Let A denote the critical clique of G containing v and let $\mathcal{B} := \{B_1, B_2, \ldots, B_\ell\}$ denote the critical cliques of G in $N_G(v)$. Note that for every B_i there exists exactly one further (besides K) maximal clique K_i in G with $B_i \subseteq K_i$. Since all vertices in K are satisfied, clearly all vertices in all B_i's from $\mathcal{B}$ are satisfied. Hence, by Lemma 9, the K_i's are pairwise vertex disjoint and every vertex $x \in K_i \setminus K$ is adjacent to all vertices in B_i but not to any other vertex in K.

Next, consider the vertices of K in $G_{S'}$. By Lemma 8, we know that a critical clique B_i is not split by S'. Let $\mathcal{B}' \subseteq \mathcal{B}$ be the set of all $\mathcal{B}_i \in \mathcal{B}$ such that there exists a vertex $x \in K \setminus B_i$ adjacent in $G_{S'}$ to a vertex in B_i. Without loss of generality, let $\mathcal{B}' := \{B_1, B_2, \ldots, B_t\}$ and define $K' := A \cup \bigcup_{i=1}^{t} B_i$. Note that S' contains edge deletions between vertices in K', since otherwise, v would be contained in at most one maximal clique in $G_{S'}$. We show that we can undo all edge deletions between vertices of K' without creating unsatisfied vertices. More specifically, let G'' denote the graph that results by undoing all edge deletions within $G_{S'}[K']$. We show that G'' fulfills the 2-vertex overlap property. That is, we claim that $S := S' \setminus E_{K'}$ is a solution, contradicting the fact that S' is a minimum solution for G.

Observe that for every $B_i \in \mathcal{B}'$ in $G_{S'}$ there exists a maximal clique $Q_i \subseteq K$ containing B_i: By definition, for every $B_i \in \mathcal{B}'$ there exists a vertex $x \in K \setminus B_i$ adjacent in $G_{S'}$ to a vertex $u \in B_i$. Note that, since the common neighborhood of x and u in G is K, the edge $\{u, v\}$ is contained in a maximal clique $Q_i \subseteq K$ in $G_{S'}$. Since all vertices in B_i have the same closed neighborhood in $G_{S'}$, B_i is completely contained in one maximal clique $Q_i \subseteq K$ and, hence, in at most one further maximal clique $K'_i \subseteq K_i$.

Consider the graph G''. We argue that all vertices of G'' are satisfied. Clearly, all vertices in K' are contained in at most two maximal cliques in G'', namely in K' and in at most one further clique $K'_i \subseteq K_i$ (note that the K'_i's are pairwise vertex disjoint). Note that a vertex $w \in V \setminus K'$ can only become "unsatisfied" by undoing the edge deletions within $G_{S'}[K']$ if it has two neighbors in K'. However, only a vertex $w \in K'_i \setminus K'$ can have two neighbors in K' in $G_{S'}$. Since a vertex $w \in K'_i \setminus K'$ is adjacent in $G_{S'}$ to all vertices in B_i but not to any other vertex of K' and since B_i is a clique in $G_{S'}$, no edge deletion is "undone" between any two neighbors of w. In summary, G'' fulfills the 2-vertex overlap property, a contradiction to the fact that S' is a minimal solution. □

Lemma 11. *Rule 2 is correct and can be carried out in $O(n^{3.376})$ time.*

Proof. Let $G = (V, E)$ denote the graph containing a maximal clique K that is satisfied and K's neighbors are all satisfied. Moreover, let $G' = (V', E')$ denote the graph that results by removing all edges in K contained only in K.

To show the correctness of the rule, we need the following definitions. Let $\mathcal{B} := \{B_1, B_2, \ldots, B_\ell\}$ denote the critical cliques of G contained in K. Note that for every B_i there exists at most one further maximal clique K_i in G with $B_i \subseteq K_i$. Further, by Lemma 9, it follows that the K_i's are pairwise vertex disjoint.

For the correctness, we show the following.

Claim: G has a solution of size k iff G' has a solution of size k.

"$\Rightarrow$": Let S denote a minimum solution of size at most k for G and let $G_S := (V, E \setminus S)$. We show that $S' := S \setminus E_K$ is a solution for G'. Let $G'_{S'} := (V, E' \setminus S')$. Note that according to Lemma 8, every B_i is contained completely in at most two maximal cliques in G_S. Moreover, every vertex $x \in V \setminus K$ that is adjacent in G_S to a vertex of B_i is adjacent in G_S to every vertex in B_i but not to any other vertex in K. Since $G'_{S'}$ differs from G_S in the way that all edges between different B_i's are deleted, the graphs induced by the closed neighborhood of every vertex in $x \in V \setminus K$ in G_S and $G'_{S'}$ are isomorphic. Hence, these vertices are satisfied. Next, we argue that every B_i is contained in at most two maximal cliques in $G'_{S'}$. Every B_i is contained in at most two maximal cliques in G_S. First, consider the case that B_i is contained in two maximal cliques C_i^1 and C_i^2 in G_S. If $C_i^1 \subseteq K_i$ and $C_i^2 \subseteq K_i$, then there cannot be any edges between B_i and $K \setminus B_i$ in G_S (since otherwise B_i would be contained three maximal cliques). Hence, the graphs induced by the closed neighborhood of a vertex in B_i in G_S and $G'_{S'}$ are identical. If $C_i^1 \subseteq K$ and $C_i^2 \subseteq K_i$, then we have $C_i^1 \cap C_i^2 = B_i$, since a vertex in C_i^2 is adjacent in G_S to all vertices in B_i but not to any other vertex in K. Hence, after deleting the edges between B_i, the vertices of B_i are contained in exactly one maximal clique, namely $C_i^2 \subseteq K_i$. For the case that B_i is contained in exactly one maximal clique, the argumentation works in analogy. In summary, $G'_{S'}$ fulfills the 2-vertex overlap property.
"$\Leftarrow$": Let S' denote a minimum solution of size at most k for G'. We show that S' is a minimum solution for G as well. By Lemma 10, we can assume that every B_i is completely contained in exactly one maximal clique $K'_i \subseteq K_i$ in $(V', E' \setminus S')$. Further, recall that all K'_i's are pairwise vertex disjoint and every vertex in $K'_i \setminus B_i$ is adjacent in $(V', E' \setminus S')$ to all vertices B_i but not to any other vertex in K. Hence, if we add all missing edges between the vertices of K in $(V, E' \setminus S')$ (resulting in $(V, E \setminus S')$), then there is no edge added between two neighbors of vertices in $K'_i \setminus K$. Hence, these vertices are satisfied $(V, E \setminus S')$. Moreover, since these are the only vertices in $V \setminus K$ that have at least two neighbors in $(V', E' \setminus S')$ in K all vertices in $V \setminus K$ are satisfied $(V, E \setminus S')$. Finally, all vertices in K are clearly contained in at most two maximal cliques in $(V, E \setminus S')$, namely in K and in at most one further clique $K'_i \subseteq K_i$ (note that for two vertices from different B_i's their common neighborhood in $(V, E \setminus S')$ is K).

For the running time, note that we can compute the set U of all satisfied vertices in $O(n^{3.376})$ time. After that, we consider the vertices in U one by one.

Every vertex $u \in U$ is contained in at most two maximal cliques K_1 and K_2. These two cliques can be computed in $O(n^{2.376})$ time for every $u \in U$. Finally, we check whether K_1 or K_2 fulfills the precondition of Rule 2. This is clearly doable in $O(n^2)$ time. Hence, the overall running time for one application of Rule 2 is bounded by $O(n^{3.376})$. □

Correctness of Rule 3. To prove the correctness of Rule 3, we need the following lemma.

Lemma 12. *Let $G = (V, E)$ denote a graph reduced with respect to Rule 1. Let K and $K_1, \ldots, K_\ell$ be maximal cliques in G fulfilling the two conditions of Rule 3 and $\sum_{i=1}^{\ell} |K_i \cap K| \geq 2k + 2$. Then, there exists a minimum solution S of size at most k such that $S \cap E_K = \emptyset$.*

Proof. Let S' be a minimum solution for G and let $G_{S'} := (V, E \setminus S')$. Assume that $S' \cap E_K \neq \emptyset$. Let $u_1, u_2, \ldots, u_t$ with $t \geq 2k+2$ denote the vertices of $\bigcup_{i=1}^{\ell} K_i$. Clearly, since $t \geq 2k+2$, one of these vertices is non-affected by S'. Without loss of generality, assume $u_1 \in K_1$ is one of these non-affected vertices. Let $B_1 := K \cap K_1$. Clearly, B_i is a critical clique in G. By Lemma 8 we can assume that B_1 is (part of) a critical clique in $G_{S'}$, and, hence, all vertices in B_1 are non-affected. Let $\{v, w\}$ be an edge in $S' \cap E_K$. Thus, neither v nor w is contained in B_1. Let $z \in K_1 \setminus K$. Since u_1 is non-affected by S' we know that $\{z, u_1\} \in E(G'_S)$. Moreover, by Lemma 9 (and condition two of Rule 3) it follows that $\{z, v\}$ and $\{z, w\}$ are not contained in E. Hence, z is contained in at least three maximal cliques in $G_{S'}$; a contradiction to the fact that S' is a solution. □

Lemma 13. *Rule 3 is correct and can be carried out in $O(n^{3.376})$ time.*

Proof. We show that $G = (V, E)$ has a minimum solution of size at most k iff the graph $G' = (V', E')$ that results by deleting all edges between between $K_1 \cap K$ and $K \setminus K_1$ has a minimum solution of size at most k. In the following, let $B_i := K_i \cap K$.

Let S denote a minimum solution of size at most k for G and let $G_S := (V, E \setminus S)$. We show that S is a solution for G'. Let $G'_S := (V', E' \setminus S)$. By Lemma 12, S does not delete any edge within K. In G_S, B_i is contained in K and in at most one further maximal clique $K'_i \subseteq K$. Hence, we can imagine that G'_S results from G_S by first deleting all vertices in B_i (which does not create unsatisfied vertices) and then adding the vertices of B_i to the resulting graph and making each adjacent to all vertices in K'_i (which clearly does not create any unsatisfied vertices, too).

Let S' denote a minimum solution of size at most k for G' and let $G'_{S'} := (V', E' \setminus S')$. We show that $G_{S'} := (V, E \setminus S')$ fulfills the 2-vertex overlap property. Note that in G', K_1 forms a clique all whose vertices are satisfied and that the vertices in B_1 are only contained K_1. Hence, according to Lemma 10, B_1 is contained in exactly one maximal clique $K'_1 \subseteq K_1$ in $G'_{S'}$. Moreover, note that since G' is reduced with respect to Rule 1 and since B_1 is a critical clique, it holds that $|B_1| \leq k + 1$ and hence $\sum_{i=2}^{\ell} |B_i| \geq 2k + 2$. For the same reason,

we have $\ell \geq 3$. Moreover, it is not hard to verify that $K \setminus B_1$ forms a maximal clique in G'. Thus, by Lemma 12, S' does not delete any edge between two vertices from $K \setminus B_1$. Hence, K' is a maximal clique in $G'_{S'}$. Note that $G_{S'}$ results from $G'_{S'}$ by inserting all edges between a vertex in B_i and the vertices in $K \setminus B_i$. Clearly, this does not change the number of maximal cliques for a vertex in $V \setminus K$, since by Lemma 9 none of these has a neighbor in B_i and another neighbor in $K \setminus B_i$. Finally, all vertices in K clearly are satisfied.

For the running time, note that we can compute the set U of all satisfied vertices in $O(n^{3.376})$ time. Every vertex $u \in U$ is contained in at most two maximal cliques K_1 and K_2. These two cliques can be computed in $O(n^{2.376})$ time for every $u \in U$. Finally, we check whether K_1 or K_2 fulfills the precondition of Rule 4. This is clearly doable in $O(n^2)$ time by checking for every satisfied vertex x in K_1 (K_2 respectively) whether all vertices in the second maximal clique K' containing x are satisfied. Hence, the overall running time for one application of Rule 2 is bounded by $O(n^{3.376})$. □

Proof of Theorem 10.

Proof. Let $G = (V, E)$ denote a graph that is reduced with the above reduction rules. We show that if G has a solution of size at most k, then the number of vertices of G is bounded by $O(k^3)$.

Assume that G has a size-k solution S. We need the following notation. Let G_S denote the graph that results by deleting the edges in S from G. Further, let X denote the vertices that are affected by S and let $Y := V \setminus X$. First, note that $|X| \leq 2k$. Hence, it remains to bound $|Y|$.

Let $K_1, \ldots, K_t$ denote the maximal cliques of G_S containing at least one vertex of X. Note that $t \leq 4k$ since a vertex $x \in X$ is contained in at most two maximal cliques in G_S. Further, let $K'_i := K_i \cap Y$, $1 \leq i \leq t$. Moreover, let $Z := \{Z_1, \ldots, Z_q\}$ denote all other maximal cliques of G_S. For every $1 \leq i < j \leq t$ let $K'_{i,j} := K'_i \cap K'_j$. Clearly, the sets $K'_{i,j}$ form critical cliques in G_S. Since the vertices in $K'_{i,j}$ are non-affected, they also form critical cliques in G. As a consequence, we have $|K_{i,j}| \leq k + 1$ since G is reduced with respect to Rule 1. Let $K'_{i,cc}$ denote the vertices of K'_i that are contained only in the maximal clique K_i in G_S. By the same argument as above, we have $|K'_{i,cc}| \leq k + 1$.

Next, we show that

a) every vertex in $A_i := K'_i \setminus (K_{i,cc} \cup \bigcup_{i \neq j} K'_{i,j})$ is contained in at most two maximal cliques in G,
b) for every $1 \leq j \leq q$ every vertex in Z_j is contained in at most two maximal cliques in G,
c) every Z_j intersects with at least one A_i, $1 \leq j \leq q$, $1 \leq i \leq t$,
d) $|A_i| \leq 3k + 3$ for every $1 \leq i \leq t$, and
e) $|Z| \leq (3k+4) \cdot 4k$ and $|I_j| \leq 4k+4$, with $I_j := Z_j \setminus (\bigcup_{i=1}^{t} A_i)$ and $1 \leq j \leq q$.

a) Consider an arbitrary vertex $y \in A_i$. In X, y has only the vertices of $K_i \setminus K'_i$ as neighbors. Since $K_i \setminus K'_i$ is a clique in G_S, no edge between the vertices in

$K_i \setminus K'_i$ is deleted. Hence, no edge between any two neighbors of y is deleted, and, hence, y is contained in the same number of maximal cliques in G as in G'.

b) Note that a vertex in $y \in Z_j$ is either contained in A_i for some i, $1 \leq i \leq t$, or all its neighbors are non-affected. In the first case, y is satisfied according to a). In the second case, y is clearly contained in at most two maximal cliques in G.

c) Assume that there exists a Z_j that does not intersect with any A_i for some i, $1 \leq i \leq t$. Then, Z_j intersects only with other elements from Z. Hence, Z_j and all Z_j's neighbors are satisfied and, as a consequence, Rule 2 would apply.

d) Assume that there exists an $|A_i| \geq 3k+4$. Let $Z'_1, \ldots, Z'_p$ denote the sets in Z intersecting with A_i. Clearly, every vertex in A_i is contained in some Z'_j and, hence, $\sum_{j=1}^{p} |Z'_j \cap A_i| \geq 3k+3$. Moreover, according to b) all Z'_j's are satisfied. Thus, Rule 3 would apply: a contradiction.

e) Since every Z_j intersects with some A_i and since any other Z_h cannot intersect with A_i in the same vertices as Z_j ($j \neq h$), we have $|Z| \leq (3k+3) \cdot 4k$. Moreover, assume that there exists an I_j with $|I_j| > 4k+4$. Since G is reduced with respect to Rule 1 there exists at most $k+1$ vertices in I_j that are contained in the single maximal clique Z_j. All other vertices of I_j are contained in some Z_h with $h \neq j$. Let $Z'_1, \ldots, Z'_p$ denote these sets in Z intersecting with Z_j. Since $|I_j| > 4k+4$, it holds that $\sum_{r=1}^{p} |Z_j \cap Z'_r| > 3k+3$, and, as a consequence Rule 3 would apply: a contradiction to the fact that G is reduced.

Putting all together, we have

$$\begin{aligned}
|Y| &\leq \sum_{i=1}^{t} |K'_i| + \sum_{j=1}^{q} |I_j| \\
&\leq \sum_{i=1}^{t} (|K'_{i,cc}| + |A_i| + \sum_{j=1}^{t} |K'_{i,j}|) + |Z| \cdot (4k+4) \\
&\leq 4k \cdot (k+1+(3k+3)+4k \cdot (k+1)) + (3k+3) \cdot 4k \cdot (4k+4) \\
&\in O(k^3)
\end{aligned}$$

□