To appear in Discrete Optimization, 2010

Graph-Based Data Clustering with Overlaps[☆]

Michael R. Fellows[a,1], Jiong Guo[b,2], Christian Komusiewicz[c,3],
Rolf Niedermeier[c], Johannes Uhlmann[c,4]

[a] *PC Research Unit, Office of DVC (Research), Charles Darwin University, Darwin,
Northern Territory 0909, Australia*
[b] *Universität des Saarlandes, Campus E 1.4, D-66123 Saarbrücken, Germany*
[c] *Institut für Informatik, Friedrich-Schiller-Universität Jena
Ernst-Abbe-Platz 2, D-07743 Jena, Germany*

Abstract

We introduce overlap cluster graph modification problems where, other than in most previous work, the clusters of the target 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 complexity dichotomy (polynomial-time solvable versus NP-hard) 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 fixed-parameter tractability (in case of constant s-values) and parameterized hardness (in case of unbounded s-values).

Key words: Cluster graph modification problems, forbidden subgraph characterization, NP-hardness, fixed-parameter tractability, W[1]-hardness, data reduction, kernelization

[☆] An extended abstract of this paper appeared in the proceedings of the 15th International Computing and Combinatorics Conference (COCOON '09), volume 5609 in LNCS, pages 516–526, Springer 2009. Main work was done while all authors were in Jena.

Email addresses: michael.fellows@cdu.edu.au (Michael R. Fellows), jguo@mmci.uni-saarland.de (Jiong Guo), c.komus@uni-jena.de (Christian Komusiewicz), rolf.niedermeier@uni-jena.de (Rolf Niedermeier), johannes.uhlmann@uni-jena.de (Johannes Uhlmann)

[1] Supported by the Australian Research Council. Work done while staying in Jena as a recipient of a Humboldt Research Award of the Alexander von Humboldt Foundation, Bonn, Germany.

[2] Partially supported by the DFG, research project DARE, GU 1023/1.

[3] Supported by a PhD fellowship of the Carl-Zeiss-Stiftung and the DFG, research project PABI, NI 369/7.

[4] Supported by the DFG, research project PABI, NI 369/7.

1. Introduction

Graph-based data clustering is an important tool in exploratory data analysis [31, 32, 36]. The applications range from bioinformatics [3, 33] to image processing [35]. 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. [3], Shamir et al. [32] initiated a study of graph-based data clustering in terms of *graph modification* problems. Here, the task is to modify (add or delete) as few edges of an input graph as possible to obtain a *cluster graph*, that is, a *vertex-disjoint* union of cliques. The corresponding problem is referred to as Cluster Editing. Numerous recent publications build on this concept of cluster graphs [4, 6, 9, 10, 11, 13, 15, 17, 30]. To uncover the *overlapping* community structure of complex networks in nature and society [28], however, the concept of cluster graphs so far fails to model that clusters may overlap. Consequently, it has been criticized explicitly for this lack of overlaps [11]. In this work, we introduce a graph-theoretic relaxation of the concept of cluster graphs by allowing, to a certain extent, 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. By definition, 1-vertex-overlap means that the cluster graph is a vertex-disjoint union of cliques (that is, there is no overlap of the clusters and, thus, the corresponding graph modification problem is Cluster Editing). 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 most extensively 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 studied from a theoretical [1, 5, 10, 13, 15, 17, 30] as well as a practical side [6, 11]. The majority of these works deals with the parameterized complexity of Cluster Editing, having led to efficient search-tree based [5, 15] and polynomial-time kernelization [9, 13, 15, 17, 30] algorithms. One motivation of our work is drawn from these intensive studies, motivated by the practical relevance of Cluster Editing and related problems. As discussed above, however, Cluster Editing forces a sometimes too strict notion of cluster graphs by disallowing any overlap. To the best of our knowledge, relaxed versions of Cluster Editing and the cluster graph concept have been largely unexplored.[5] There are only two approaches studying overlapping cliques in the context of Cluster Editing that we are aware of. One was proposed by Barthélemy and Brucker [2] under the name t-Zahn Clustering, where the aim is to obtain by a minimum number of edge modifications a graph in which each pair of maximal cliques has at most $t-1$ vertices in common. The base case $t=1$ is thus equivalent to Cluster Editing. Among other things, Barthélemy and Brucker [2] showed that 2-Zahn Clustering is NP-hard. The model of Barthélemy and Brucker [2] allows, for constant t, for vertices and edges to be in an unbounded number of maximal cliques. In contrast, our model limits the number of maximal cliques that a vertex or clique is contained in, but already for constant s there can be maximal cliques that intersect in an unbounded number of vertices. The second approach was presented by Damaschke [10], who investigated the Twin Graph Editing problem, where 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. A t-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 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 [10] 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.

Our results. We provide a thorough study of the computational complexity of clustering with vertex and edge-overlaps, 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-hard) of the corresponding edge modification problems, most of them turning out to be NP-hard (for an overview, see Table 1 in Section 4). For instance, 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-hard. 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 the combined parameter (s,k). This result is based on forbidden subgraph characterizations of the underlying overlap cluster graphs, that may be of independent graph-theoretic interest. In particular, 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.

[5] Two recent exceptions are so-called s-plex cluster graphs [19] and (p,q)-cluster graphs [21].

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 in both cases k denotes the number of allowed edge deletions. We conclude in Section 7 with a number of open problems.

2. Preliminaries

Given an undirected 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 $N_G(v)$ of a vertex v is the set of vertices that are adjacent to v, and the closed neighborhood $N_G[v] := N_G(v) \cup \{v\}$. For a vertex set $S \subseteq V$ let $N_G(S) := \bigcup_{v \in S} N_G(v) \setminus S$ denote the neighborhood of S. The degree of a vertex v, denoted by $\deg_G(v)$, is the cardinality of $N_G(v)$. If G is clear from the context, we omit the subscript G. We use $G[S]$ to denote the subgraph of G induced by $S \subseteq V$, that is, $G[S] := (S, \{\{u, v\} \mid u, v \in S, \{u, v\} \in E\})$. Moreover, $G - v := G[V \setminus \{v\}]$ for a vertex $v \in V$ and $G - e := (V, E \setminus \{e\})$ for an edge $e = \{u, v\}$. For two sets E and F let $E \Delta F := (E \setminus F) \cup (F \setminus E)$ denote the symmetric difference of E and F. 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. Furthermore, 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 from modifying G according to S. A set of pairwise adjacent vertices is called a *clique*. A clique K is a *critical clique* if all its vertices have an identical closed neighborhood and K is maximal under this property. 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 vertex deletion.

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 $S \subseteq E_V$ 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 3). 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 [12, 14, 27]. 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. A core tool in the development of fixed-parameter algorithms is polynomial-time preprocessing by *data reduction* [7, 20]. Here, the goal is for a given problem instance x with parameter k, to transform it into a new

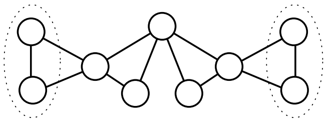

Figure 1: A graph with 2-vertex-overlap and 1-edge-overlap property. The two critical cliques of size two are encircled by dotted lines; all other vertices form critical cliques of size one.

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 if and only if (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 *kernelization*.

Downey and Fellows [12] developed a formal framework for showing *fixed-parameter intractability* by means of *parameterized reductions*. A parameterized reduction from a parameterized problem P to another parameterized problem P' 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) is a yes-instance of problem P if and only if (x', k') is a yes-instance of problem P'. 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 [12, 14, 27]. In this sense, $W[1]$-hardness is the parameterized complexity analog of NP-hardness.

3. Recognition and Forbidden Subgraph Characterization

In this section, we first introduce the two graph properties considered in this work. Then, we show that, for each fixed s, it can be recognized in polynomial time whether a given graph has the respective overlap property. Moreover, we will show that the graph properties, for each fixed s, are characterized by a finite set of forbidden induced subgraphs. More specifically, we show that the forbidden graphs are all of order s^2 and that there is a polynomial-time algorithm that given a graph, either determines that G fulfills the property or identifies an induced subgraph of G that is forbidden.

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 the 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. Note that this graph has one connected component whose critical-clique graph has eight edges. It is thus an 8-twin graph.

For a graph G 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.

Theorem 1. *For a graph $G = (V, E)$ and a non-negative integer s, there is an algorithm that, in $O(s \cdot n^{3.376})$ (or $O(s \cdot m \cdot n^{2.376})$) time, either*

- *finds a vertex (or an edge) that is contained in more than s maximal cliques, or*
- *correctly concludes that G has the s-vertex-overlap (or s-edge-overlap) property.*

Proof. For each $v \in V$, we enumerate the maximal cliques in $G[N[v]]$. If we have found $s + 1$ maximal cliques in $G[N[v]]$ for some $v \in V$, then we abort the enumeration and report that v is in more than s maximal cliques. Otherwise, each $v \in V$ is contained in at most s maximal cliques, and the graph thus fulfills the s-vertex-overlap property. Using for example a polynomial-delay enumeration algorithm by Makino and Uno [24] 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 we consider the common neighborhood of the endpoints of every edge, that is, $N[u] \cap N[v]$ for an edge $\{u, v\}$. □

The next lemma implies the existence of forbidden induced subgraph characterizations for graphs having the s-vertex-overlap or the s-edge-overlap property.

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

Proof. 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 $w \in N_G(v)$ contained in at least $s+1$ distinct maximal cliques of $G-v$, say $C_1, \ldots, C_{s+1}$. 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 exist vertices $u_i \in C_i$ and $u_j \in C_j$ 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.

In complete analogy one shows that the s-edge-overlap property is hereditary, replacing the vertex $w \in N_G(v)$ with an edge $\{u, w\} \subseteq N_G(v)$ in the argument above. □

Hereditary graph properties can be characterized by a finite or infinite set of forbidden induced subgraphs [16]. Thus, by Lemma 1, such a set must exist

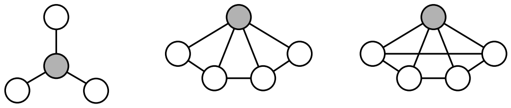

Figure 2: The forbidden induced subgraphs for the 2-vertex-overlap property. In every graph, the gray vertex is contained in at least three maximal cliques.

for "s-vertex-overlap graphs" as well as for "s-edge-overlap graphs". Here, we show that the minimal forbidden induced subgraphs contain $O(s^2)$ vertices. For fixed s, the number of minimal forbidden induced subgraphs is thus finite. Furthermore, we describe an algorithm for efficiently finding a forbidden induced subgraph.

Theorem 2. *For 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 an $O(s^2)$-vertex forbidden induced subgraph.*

Proof. We first show 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 it follows that such a vertex can be found in $O(s \cdot n^{3.376})$ time. Given $s+1$ maximal cliques $K_1, \ldots, K_{s+1}$ containing v, we find a forbidden induced subgraph as follows. To "separate" two maximal cliques K_i and K_j, we need a vertex $v_1 \in K_i \setminus K_j$ and a vertex $v_2 \in K_j \setminus K_i$ with $\{v_1, v_2\} \notin E$. Clearly, such vertices exist since both K_i and K_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. For each pair of cliques, we can find the separating vertices in $O(n)$ time, scanning the vertex-lists of each clique, "marking" the vertices that are contained in both cliques, and then keeping one unmarked vertex of each list. Altogether, we thus need $O(s^2 \cdot n)$ time.

For the edge case, we can find in $O(s \cdot m \cdot n^{2.376})$ time an edge $\{u, v\}$ that is contained in at least $s+1$ maximal cliques. The vertices needed to "separate" the $s+1$ maximal cliques $K_1, \ldots, K_{s+1}$ in $G[N[v] \cap N[u]]$ can be found analogously to the vertex case. □

Figure 2 illustrates the minimal forbidden induced subgraphs for graphs with the 2-vertex-overlap property. Many important graph classes are contained in the class of graphs with some s-overlap property. In particular, it is easy to see that diamond-free graphs are equivalent to graphs with the 1-edge-overlap property. A *diamond* is the graph that results from a four-vertex clique by deleting one edge. Diamond-free graphs, that is, graphs containing no diamond as an induced subgraph, are a natural graph class and have been already studied in earlier work [2, 34].

Proposition 1. *A graph has the 1-edge-overlap property if and only if it is diamond-free.*

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

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

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 contained in at least two maximal cliques. Hence, there is a pair of non-adjacent vertices that are both adjacent to the endpoints of this edge. Therefore, the graph contains at least one induced diamond. □

The property of being diamond-free can also be described as follows: every pair of maximal cliques has at most one vertex in common. Graphs with the 1-edge-overlap property are thus precisely the graphs with 2-Zahn property as defined by Barthélemy and Brucker [2].

4. A Complexity Dichotomy with Respect to Overlap Number s

This section provides a complete picture of the classical computational complexity of the introduced problems. The results are summarized in Table 1. With the exception of the two basic addition problems for $s = 1$, all of the problems turn out to be NP-hard.

First, we show that if one of the problems is NP-hard for some $s \geq 1$, then it is NP-hard for every $s' \geq s$. The basic idea is that, given a problem instance for some value s, we can reduce to an instance for $s+1$ by adding 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 instance only in v (respectively u and v).

Lemma 2. *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}.

Proof. 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 for the deletion and addition variants as well. 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 $G = (V, E)$ and an integer k, we construct an $(s + 1)$-VOE instance

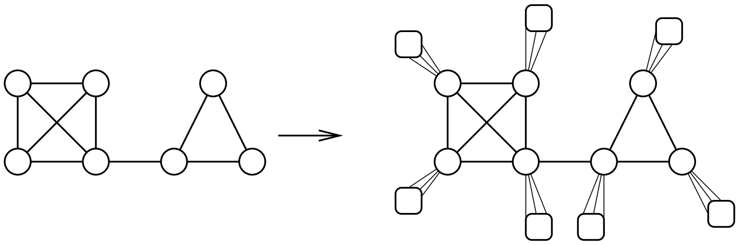

Figure 3: Illustration of the reduction from s-Vertex-Overlap Editing to $(s+1)$-Vertex-Overlap Editing. Herein, every rectangular vertex represents a clique on $2k+2$ vertices.

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 a set C_v of $2k+2$ new vertices to H and we make $\{v\} \cup C_v$ a clique. An illustration of this construction is given in Figure 3.

Next, we show the correctness of the reduction, that is, we show that G has a solution of size at most k for S-VOE if and only if H has a solution of size at most k for $(s+1)$-VOE. First, consider a solution S of size at most k for s-VOE with input graph G. In the graph that results from modifying G according to S, every vertex is contained in at most s maximal cliques. Hence, if we modify H according to S, we obtain a graph in which every vertex is contained in at most $s+1$ maximal cliques. Second, let S' denote a solution of size at most k for $(s+1)$-VOE for H. Moreover, let $H' = H \Delta S'$. Since $|S| \leq k$, there are at most $2k$ vertices that are affected by S. Hence, in H' every vertex $v \in V$ is adjacent to a non-empty set $N_v \subseteq C_v$ of non-affected vertices (since $|C_v| = 2k+2$). 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 $H'[V]$. That is, $H'[V]$ 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. □

The following NP-hardness results can be obtained directly from combining known results with Lemma 2: Since Cluster Editing and Cluster Deletion (equivalent to 1-Vertex-Overlap Editing and 1-Vertex-Overlap Deletion, respectively) are NP-complete [23, 32], the NP-hardness of s-Vertex-Overlap Editing and s-Vertex-Overlap Deletion for all $s > 1$ directly follows. Furthermore, 1-Edge-Overlap Editing has also been shown to be NP-complete by a reduction from Cluster Editing [2] that can also be used to show the NP-hardness of 1-Edge-Overlap Deletion (simply by reducing from Cluster Deletion instead). The NP-hardness for both the editing and the deletion variant and $s > 1$ thus also follows for edge-overlap. Overall, we arrive at the following theorem.

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

It thus remains to determine the classical computational complexity of s-Vertex-Overlap Addition and s-Edge-Overlap Addition. 1-Vertex-Overlap Addition is trivially polynomial-time solvable: one has to transform every connected component into a clique by adding the missing edges. The same observation can be made for 1-Edge-Overlap Addition, since there exists only one possibility to destroy a diamond by adding edges; by Proposition 1, diamonds are the only forbidden subgraph of graphs having the 1-edge-overlap property.

In contrast, for $s \geq 2$, both s-Vertex-Overlap Addition and s-Edge-Overlap Addition become NP-hard, as we will show in the following.

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

Proof. We present a polynomial-time many-one reduction from the NP-hard Maximum Edge Biclique problem [29] to 2-Vertex-Overlap Addition (2-VOA). Then, for $s \geq 2$, the NP-hardness follows directly from Lemma 2. 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 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: For 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,

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

That is, the graph $(U, W, E_{\overline{F}})$ is the bipartite complement of H, in G both U and W are cliques, and r is adjacent to all vertices in G. See Figure 4 a) for an illustration of this construction.

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

> *Claim:* In the 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.

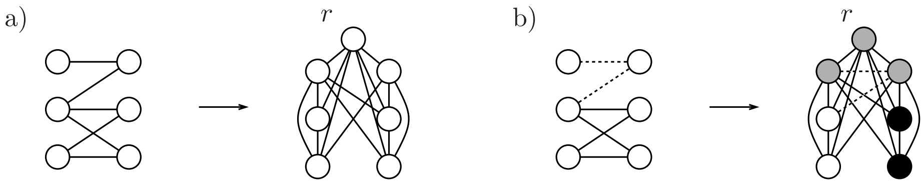

Figure 4: a) Example for the reduction from Maximum Edge Biclique (left graph) to 2-Vertex-Overlap Addition (right graph), b) The graph on the left contains a biclique with four edges (solid edges). Adding the edges not contained in this biclique (dashed edges) to the graph on the right results in a graph that contains two maximal cliques. The gray vertices are in both maximal cliques, the white and black vertices are in one maximal clique.

"⇒": 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 and one complete bipartite graph with at least l edges. Moreover, $|F'| \leq |F| - l$. Let G' denote the graph that results from 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 follows 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 4 b) 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 from 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 from 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 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 an induced 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 from 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 $G'[\{u, u', w, w'\}]$ is an induced cycle of length four. Further, since r is adjacent to all vertices, $G'[\{u, u', w, w', r\}]$ is isomorphic to the third graph shown in Figure 2, a contradiction to the fact that G' fulfills the 2-vertex-overlap property. □

Finally, we consider s-Edge-Overlap Addition. 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 instance. The correspondence between the solutions of both instances can be shown in complete analogy with the vertex-overlap case.

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

5. Parameterized Complexity

Here, we consider the parameterized complexity of overlap clustering. First, due to Theorem 2, we have a set of forbidden subgraphs for both properties whose size only depends on s. Cai [8] showed that edge modification problems for properties that can be described by forbidden subgraphs of fixed size are fixed-parameter tractable with respect to the parameter "solution size". Hence, we can conclude that for both overlap properties all three problems are fixed-parameter tractable with respect to the combined parameter (s, k).

Theorem 6. *For $\pi \in \{s$-Vertex-Overlap, s-Edge-Overlap$\}$, π Editing, π Addition, and π Deletion 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. For this parameterization, we show that for both overlap properties the deletion and editing problems are W[1]-hard by developing a parameterized reduction from the W[1]-complete Set Packing problem [12]. We leave open the parameterized complexity of the two addition problems with only k as parameter.

Theorem 7. *For $\pi \in \{s$-Vertex-Overlap, s-Edge-Overlap$\}$, π Editing and π Deletion are W[1]-hard with respect to the parameter k.*

Proof. We give the proof details only for s-Vertex-Overlap Deletion (s-VOD), and then discuss how the reduction can be modified to work for s-Vertex-Overlap Editing and edge-overlap. We show the W[1]-hardness of s-VOD by presenting a parameterized reduction from the W[1]-complete Set Packing problem [12], which is defined as follows:

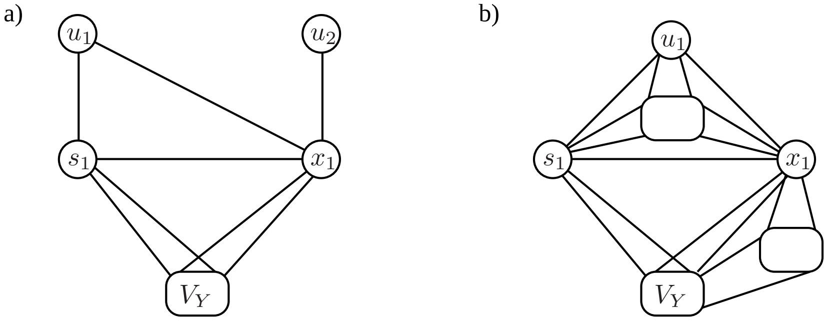

Figure 5: Parts of the graph constructed in the reduction from Set Packing to s-Vertex-Overlap Deletion. Rectangles depict cliques of size at least $k+1$. a) A subgraph containing $u_1, u_2 \in V_U$, $s_1 \in V_S$, $x_1 \in V_X$, and V_Y. Edges are drawn between $s_i \in V_S$ and $u_j \in V_U$ if $j \in S_i$. Here, $1 \in S_1$ but $2 \notin S_1$. b) 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 $x_i \in V_X$.

> 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 $\forall S_i, S_j \in \mathcal{S}'$: $S_i \cap S_j = \emptyset$?

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-VOD 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:

- $V_U := \{u_1, \ldots, u_m\}$ contains one vertex for each element $i \in U$.
- $V_S := \{s_1, \ldots, s_n\}$ contains one vertex for each $S_i \in \mathcal{S}$.
- $V_X := \{x_1, \ldots, x_k\}$ contains k vertices and $V_Y := \{y_1, \ldots, y_{2k+1}\}$ contains $2k+1$ vertices; together they serve as a "selection" gadget.
- V_C contains vertices that are part of some "shielding" cliques. With these cliques, we can enforce that some edges will never be edited.
- 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 $y \in V_Y$ 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 Figure 5a.

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: the maximal clique $K = \{x_i, s_j\} \cup V_Y$ is destroyed and the clique $K' = \{x_i\} \cup V_Y$ which after deleting $\{x_i, s_j\}$ is the maximal subset of K that is a clique is a subset of the clique $\{x_i\} \cup V_Y \cup C^{x_i}$. Furthermore, no additional maximal cliques are created by the deletion of $\{x_i, s_j\}$.

Then, for each edge $\{u_i, s_j\}$, and for each $x_l \in V_x$, we add the 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, one can only delete edges between x_i and V_S. An example of these shielding cliques is shown in Figure 5b.

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. We denote this number for some vertex v by $\#(v)$.

- Each vertex $u_i \in V_U$ is contained in $\#(u_i) = |N(u_i) \cap V_S| \cdot k \leq n \cdot k$ maximal cliques: For each $s_j \in N(u_i) \cap V_S$ and each $x_l \in V_X$ the set $\{u_i, s_j, x_l\} \cup C^{u_i,s_j,x_l}$ is a maximal clique since V_X and V_S are independent sets and by the definition of V_C; no other maximal cliques contain u_i.

- 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: For each $s_j \in V_S$ and for each $u_l \in N(s_j) \cap V_U$ the set $\{x_i, s_j, u_l\} \cup C^{u_l,s_j,x_i}$ is a maximal clique, for each $s_j \in V_S$ the set $\{x_i, s_j\} \cup V_Y$ is a maximal clique, and $\{x_i\} \cup V_Y \cup C^{x_i}$ is a maximal clique; no other maximal cliques contain x_i.

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

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 (by the definition of $\#(u_i)$ and the number of added padding cliques),

- each vertex $x_i \in V_X$ is contained in exactly $n \cdot (m+1) + 1$ maximal cliques (by the definition of $\#(x_i)$ and the number of added padding cliques),

- each vertex $y_i \in V_Y$ is contained in exactly $n \cdot k + k < n \cdot (m+1)$ maximal cliques (one maximal clique for each pair of $x_j \in V_X$ and $s_l \in V_S$, and one maximal clique for each $\{x_j\} \cup C^{x_j}$, $x_j \in V_X$),

- each vertex $s_i \in V_S$ is contained in exactly $k \cdot (|N(s_i) \cap V_U| + 1) < n \cdot (m+1)$ maximal cliques (one maximal clique for each pair of $x_j \in V_X$ and $u_l \in N(s_i) \cap V_U$ and, furthermore, for each $x_j \in V_X$ the maximal clique $\{s_i, x_j\} \cup V_Y$), 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. The main idea of the reduction can be described as follows. For each vertex in V_X we have to reduce the number of maximal cliques it is contained in. This can only be done by deleting edges between V_X and V_S. This corresponds to selecting a set in the Set Packing instance. However, we also force that for each vertex in V_U the number of maximal cliques it is contained in increases at most by one. Hence, at most one of its neighbors in V_S can be "selected". This corresponds to the disjointness of the sets of the Set Packing solution.

To show the W[1]-hardness of s-VOD parameterized by k, we prove the following.

> *Claim:* (I, k) is a yes-instance for Set Packing if and only if (G, k) is a yes-instance for $(n \cdot (m+1))$-VOD.

"$\Rightarrow$": 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 obtain a solution S' of $(n \cdot (m+1))$-VOD by setting $S' := \{\{x_i, s_i\} \mid 1 \leq i \leq k\}$. Let $G' := G \Delta S'$. To see that G' fulfills the $(n \cdot (m+1))$-vertex-overlap property, we only need to consider 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}, namely $C^{u_i, s_j, x_j} \cup \{u_i, s_j\}$ 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, we show that 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, by removing $\{s_i, x_i\}$ we destroy the maximal clique $\{s_i, x_i\} \cup V_Y$, since the 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. For each vertex in $v \in V_Y$ the number of maximal cliques that it is contained in has not increased, since for each $\{x_i, s_i\}$ that was deleted, the maximal clique $\{s_i, x_i\} \cup V_Y$ is destroyed, the clique $V_Y \cup \{s_i\}$ becomes a new maximal clique, and the clique $V_Y \cup \{x_i\}$ is a subset of the clique $V_Y \cup \{x_i\} \cup C^{x_i}$.

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, then 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 $n \cdot (m+1)$-VOD.

"$\Leftarrow$": 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 maximal 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 $s_j \in V_S$. Note that with k edge deletions, we cannot decrease the number of maximal cliques that contain both x_i and some 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, for each shielding or padding clique there remains 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. Suppose S' does not delete any edges between x_i and V_S. We show that in this case G' contains 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, a size-k solution S' contains for each $x_i \in V_X$ an edge from x_i to a vertex from V_S.

Second, we show that for each $s_i \in V_S$ there is at most one edge incident to s_i that is deleted by S', and thus that S' corresponds to a size-k subset of $\mathcal{S}$. Suppose, otherwise, that for some $s_i \in V_S$, at least two incident edges, say $\{s_i, x_i\}$ and $\{s_i, x_j\}$ have been deleted. Then, for each $u_l \in V_U \cap N(s_i)$ the number of maximal cliques that contain u_l has increased by at least two, since, instead of the two maximal cliques $\{s_i, x_i, u_l\} \cup C^{u_l, s_i, x_i}$ and $\{s_i, x_j, u_l\} \cup C^{u_l, s_i, x_j}$ that are destroyed, there are now four maximal cliques (two for each deleted edge, one that contains s_i and one that contains x_i or x_j, respectively). Then, however, u_j is in more than $n \cdot (m+1)$ maximal cliques, which contradicts that S' is a solution. Hence, we can assume without loss of generality that $S' := \{\{s_i, x_i\} \mid 1 \leq i \leq k\}$.

Finally, we show that the set $\mathcal{S}' := \{S_i \mid 1 \leq i \leq k\}$ is a solution of the Set-Packing instance (I, k). To this end we show that each $u \in V_U$ can have at most one neighbor in $\{s_i \mid 1 \leq i \leq k\}$. Otherwise, the number of maximal cliques that contain u has increased by at least two, a contradiction to the fact that S' is a solution. Hence, $\mathcal{S}'$ is a size-k subset of $\mathcal{S}$ such that every $u \in U$ is contained in at most one $S_i \in \mathcal{S}'$.

Altogether, we have shown the equivalence between the solutions of s-Vertex-Overlap Deletion and Set Packing. This implies that s-Vertex-Overlap Deletion is W[1]-hard, when parameterized only by k.

For s-Vertex-Overlap Editing, the construction has to be modified as follows. Instead of adding only one clique V_Y, we add a clique C_j^i for each pair of vertices x_i and s_j. This ensures that adding edges between distinct $s_j, s_l \in V_S$ does not reduce the number of cliques that each x_i is contained in. Note also that edge additions between distinct $u_i, u_j \in V_U$ and between V_U and V_S do not decrease the number of cliques that a vertex from x_i is contained in because of the large shielding cliques for each triangle in $G[V_U \cup V_S \cup V_X]$. Hence, the only choice to decrease the number of cliques that each $x_i \in V_X$ is contained in is again the deletion of an edge between x_i and V_S. The correctness proof then works in complete analogy with the deletion case.

For s-Edge-Overlap Deletion and Editing, we replace each vertex of $v \in V_U \cup V_X$ with two adjacent vertices, and add further "shielding cliques" that ensure that the edge between these two vertices is not deleted. The correctness proofs work analogously; we omit the details. □

6. Two Kernelization Results for Edge Deletion

Nontrivial overlap clustering problems seem to be algorithmically more demanding than clustering without overlaps. We present polynomial-time kernelization algorithms for the two most basic NP-hard clustering problems with nontrivial overlap.

6.1. An $O(k^4)$-vertex problem kernel for 1-Edge-Overlap Deletion

We present a kernelization for 1-edge-overlap Deletion, which, by Proposition 1, is equivalent to the problem of destroying *diamonds* by at most k edge deletions. We introduce four data reduction rules for this problem and show that a yes–instance reduced with respect to these rules has $O(k^4)$ vertices. Rules 1, 2, and 4 find parts of the graph that need not be modified by optimal solutions, whereas Rule 3 identifies edges that must be in any solution of size at most k.

Rule 1. If there is a maximal clique K containing only edges which are not in any other maximal clique, then remove all edges of K.

Lemma 3. *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 from applying Rule 1 to a maximal clique K in G. To show the correctness of Rule 1, we prove the following.

Claim: (G,k) is a yes-instance if and only if (G',k) is a yes-instance.

"$\Rightarrow$": Let S denote an optimal solution for G. Then S contains no edge from K. To see this, observe that the only possible way to create a diamond containing some edge from K is to delete edges from K. However, since all edges of K are not in a diamond in G, an optimal solution will never delete them. This means that K remains a maximal clique in $G \Delta S$ and no two vertices of K have common neighbors outside of K. Thus, removing the edges of K from $G \Delta S$ does not create any diamond and S is also a solution for G'.
"$\Leftarrow$": Observe that after applying Rule 1 to K, no two vertices of K have common neighbors in G', since otherwise the edge connecting these two vertices would be contained in a diamond in G. Therefore, we can add the edges of K to the graph $G'' := G' \Delta S$, where S is an optimal solution for G', without destroying the 1-edge-overlap property of G''.

To check the applicability of Rule 1, we compute for each edge whether it is in only one maximal clique K. If so, we check further for all edges of K, whether K is the only maximal clique in which these edges are contained. Clearly, this is doable in $O(m^2)$ time. □

Rule 2. Remove all isolated vertices.

Rule 2 is clearly correct and can be performed in linear time. After the exhaustive application of Rule 1, Rule 2 is sufficient to remove all vertices from G that are not in a diamond, as we show in the following.

Proposition 2. *Let G be a graph that is reduced with respect to Rules 1 and 2. Then every vertex in G is contained in a diamond.*

Proof. Assume towards a contradiction that G contains a vertex v that is not contained in any diamond. Since G is reduced with respect to Rule 2, v has at least one neighbor. Furthermore, since v is not contained in any diamond, $G[N(v)]$ is a cluster graph, that is, a disjoint union of cliques. Let K be one of the cliques of $G[N(v)]$. Clearly, $K \cup \{v\}$ is a maximal clique in G. Furthermore, since v is not contained in any diamond, there is no vertex $u \in V \setminus N[v]$ that is adjacent to more than one vertex in K. Hence, none of the edges of $G[K \cup \{v\}]$ is contained in any other maximal clique $K' \neq K \cup \{v\}$. This contradicts G being reduced with respect to Rule 1. □

Rule 3. If there is an edge $e = \{u, v\}$ such that the complement graph of $G[N(u) \cap N(v)]$ contains a matching of size greater than k, then remove e, add e to the solution, and decrease the parameter k by one.

Lemma 4. *Rule 3 is correct and can be carried out in $O(m^2\sqrt{n})$ time.*

Proof. For $e = \{u, v\}$, let G_e denote $G[N(u) \cap N(v)]$. 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 u and v. Therefore, every matching of $\overline{G_e}$ corresponds to a set of diamonds in G, whose edge sets pairwise only have e in common. Hence, to destroy all these diamonds, we either delete e or delete one edge for every diamond. A matching of size greater than k thus forces the deletion of e. Since a maximum matching can be computed in $O(m\sqrt{n})$ time [26], the applicability of Rule 3 can be checked in $O(m^2\sqrt{n})$ time by iterating over all edges of G. □

The final data reduction rule shrinks large cliques whose vertices have identical neighborhoods, so-called critical cliques (see Section 2 for a formal definition).

Rule 4. If there is a critical clique K with more than $k+3$ vertices, then arbitrarily remove vertices from K until $|K| = k+3$.

Lemma 5. *Rule 4 is correct and can be carried out in $O(m+n)$ time.*

Proof. It suffices to prove that, for every critical clique K with at least $k+3$ vertices, every optimal solution of size at most k does not delete edges incident to the vertices of K. Assume towards a contradiction that there is an optimal solution S of size at most k that deletes an edge $\{u, v\}$ with $u \in K$. Let $G' := G\Delta S$ and let G'' be the graph resulting by adding $\{u, v\}$ to G'. Then, since S is optimal, G'' must contain a diamond containing $\{u, v\}$, and thus one vertex x which, in G'', is adjacent to u and v. Since $|K| \geq k+3$, we have $|K \setminus \{u, v, x\}| \geq k$. Moreover, since $|S \setminus \{u, v\}| \leq k-1$, there must be a vertex $y \in K \setminus \{u, v, x\}$ with $N_G(y) \setminus (K \setminus \{u, v, x\}) = N_{G''}(y) \setminus (K \setminus \{u, v, x\})$. Thus, since K is a critical clique, this means y is in G' adjacent to all of u, v, and x. This directly implies that $G''[\{u, v, x, y\}]$ is a clique and thus $G'[\{u, v, x, y\}]$ is a diamond, 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 $O(m+n)$ time [22]. □

Making combined use of Rules 1–4, we obtain a polynomial-size problem kernel for 1-Edge-Overlap Deletion.

Theorem 8. 1-Edge-Overlap Deletion *admits a problem kernel with $O(k^4)$ vertices which can be found in $O(m^3\sqrt{n})$ time.*

Proof. Let G denote an input graph reduced with respect to the above four data reduction rules, and let S be a solution of size at most k. Partition the vertices of the graph $G' := G\Delta S$ into two subsets, one set X containing the vertices that are endpoints of edges deleted by S, and $Y := V \setminus X$. Clearly, $|X| \leq 2k$. It thus remains to show that $|Y| = O(k^4)$. Define for each edge $e \in S$ the set Y_e containing the vertices in Y that, in G, occur together with e in at least one diamond. By Proposition 2, $Y = \bigcup_{e \in S} Y_e$. First, we show that every maximal clique K in $G'[Y]$ is contained in Y_e for some $e \in S$. Second, we show that for each $e \in S$ at most $4k$ maximal cliques of $G'[Y]$ are contained in Y_e, which means that there can be at most $4k^2$ maximal cliques in $G'[Y]$. Finally, we show that each of these cliques contains $O(k^2)$ vertices, which yields the claimed overall bound on the number of vertices.

First, we show that for every maximal clique K in $G'[Y]$ there is an edge $e \in S$ with $K \subseteq Y_e$. In G, there is a maximal clique K' containing K and, by Rule 1, K' has an edge $\{u, v\}$ which is in two maximal cliques, and thus there is a vertex $x \in K$ and a vertex $w \in V \setminus K'$ such that $G[\{u, v, w, x\}]$ is a diamond. Note that if $|K \cap \{u, v\}| = 2$, then no edge of $G[\{u, v, w, x\}]$ is contained in $G[X]$, contradicting the fact that S is a solution. We distinguish the cases that $|K \cap \{u, v\}|$ is either 1 or 0. First, consider the case that $|K \cap \{u, v\}| = 1$. Without loss of generality, let $u \in K$ and $v \in K \setminus K'$. Note that $\{v, w\}$ is the only edge of $G[\{u, v, w, x\}]$ with both endpoints in X, and hence $\{v, w\} \in S$. We show that for every $x' \in K$ it holds that $G[\{u, v, w, x'\}]$ is a diamond, and, hence, $K \subseteq Y_{v,w}$. Assume towards a contradiction that there is a vertex $x' \in K$ such that $G[\{u, v, w, x'\}]$ is not a diamond. Observe that $G[\{u, v, w, x'\}]$ is a clique and, hence, $G'[\{u, v, w, x'\}]$ is a diamond, contradicting the fact that S is a solution. Second, consider the case that $|K \cap \{u, v\}| = 0$, that is, $u, v \in K' \setminus K$. If for every vertex x' it holds that $G[\{u, v, w, x'\}]$ is a diamond, then $K \subseteq Y'_e$ for at least one $e' \in \{\{u, v\}, \{v, w\}, \{u, w\}\}$. Otherwise, there is a vertex $x' \in K$ such that $G[\{u, v, w, x'\}]$ is a clique. Then, however $G[\{v, w, x, x'\}]$ is a diamond and the first case applies since $\{x', v\}$ is contained in two maximal cliques and $x' \in K$ and $v \in K' \setminus K$.

Second, we show 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 at least one of u and v. Let $N_{u,v}$ denote the common neighbors of u and v in Y_e. Since G' is diamond-free, $N_{u,v}$ is an independent set and, by Rule 3, $|N_{u,v}| \leq 2k$. Let $N_u := (N(u) \setminus N(v)) \cap Y_e$ and $N_v := (N(v) \setminus N(u)) \cap Y_e$. Since $N_{u,v}$ is an independent set, no vertex from $N_u \cup N_v$ 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 $N(v) \cap N(x_i)$). Each subset N_{u,x_i} is a clique, since otherwise two non-adjacent vertices from N_{u,x_i} would form a diamond with x_i and u. The same holds for each N_{v,x_i}. Furthermore, there cannot be an edge between N_{u,x_i} and N_{u,x_j} with $i \neq j$, since otherwise two adjacent vertices $w \in N_{u,x_i}$ and $y \in N_{u,x_j}$ would form a diamond with u and x_i. Moreover, there is no maximal clique K of $G'[Y]$ completely contained in Y_e and containing an edge $\{a, b\}$ such that $a \in N_{u,x_i}$ and $b \in N_{v,x_j}$ for $i, j \in \{1, \ldots, \ell\}$. Suppose that such a clique K exists. Note that a and b must have a common neighbor in G—otherwise, the edge $\{a, b\}$ is a maximal clique to which Rule 1 applies, and thus it would have been removed. Hence, any maximal clique containing a and b also contains at least one further vertex w. In case $K \subseteq Y_e$, this further vertex is in $N(u) \cup N(v)$. Suppose without loss of generality that $w \in N(u)$. Then $G'[\{u, a, b, w\}]$ is a diamond, contradicting the diamond-freeness of G'.

In summary, we have at most $4k$ maximal cliques in $G'[Y]$ which are entirely contained in Y_e. Since there are at most k different Y_e's, and since every maximal clique in $G'[Y]$ is completely contained in at least one Y_e, there can be at most $4k^2$ maximal cliques in $G'[Y]$.

Finally, we show that every maximal clique K in $G'[Y]$ contains $O(k^2)$ vertices. This can be seen as follows. From the vertices of K, only $4k^2$ many can be in more than one maximal clique in $G'[Y]$, since every two cliques in $G'[Y]$ overlap in at most one vertex. Moreover, as argued above, $K \subseteq Y_e$ for some $e = \{u, v\} \in S$ and there is exactly one vertex in K which is adjacent to both u and v. Let K' denote the remaining vertices of K, that is, each vertex of K' has no neighbors in $Y \setminus K$ and is adjacent to at most one of u and v. We show that $|K'| \leq 2k+k+3$. Clearly, we can assume that $|K'| > 2$, since otherwise the claim is trivially fulfilled. Note that $K' \subseteq N(u)$ or $K' \subseteq N(v)$, since otherwise there would be a vertex $a \in K'$ that is adjacent to u but not to v and a vertex $b \in K'$ that is adjacent to v but not to u. Moreover, since $|K'| > 2$ there is a vertex $x' \in K'$ that is either adjacent to u or v. Assume without loss of generality that $\{x', u\} \in E$. Then, however $G'[\{a, b, u, x'\}]$ is a diamond. We now claim that for every vertex $w \in X \setminus \{u, v\}$, either $K' \subseteq N(w)$ or $|N(w) \cap K'| \leq 1$. Assume the claim is not true. Then we have two vertices $a, b \in K' \cap N(w)$ and one vertex $c \in K' \setminus N(w)$. This implies that there is a diamond consisting of a, b, c, w in G', contradicting that G' is diamond-free. This claim implies that all except for at most $2k$ vertices in K' have the same neighborhood in X. This means that they have the same neighborhood in G and thus they form a critical clique. By Rule 4, there can be at most $k+3$ of such vertices. Hence, K contains altogether at most $4k^2 + 1 + 2k + k + 3$ vertices.

Summarizing, we have at most $4k^2$ maximal cliques in $G'[Y]$, and each clique contains at most $4k^2 + 3k + 4$ vertices. Hence, $|Y| = O(k^4)$. Since each of the four data reduction rules is performed at most $O(m)$ times, the running time follows from Lemmas 3–5. □

6.2. *An $O(k^3)$-vertex kernel for 2-Vertex-Overlap Deletion*

We present four polynomial-time data reduction rules for 2-Vertex-Overlap Deletion and show that a yes–instance reduced with respect to these rules has $O(k^3)$ vertices. 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. Moreover, a maximal clique all whose vertices are satisfied is called a *satisfied maximal clique*. The first data reduction rule reads as follows.

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

The following lemma helps in showing the correctness of Rule 1. It says that there always exists an optimal solution S such that all vertices having the same closed neighborhood in the input graph G also have the same closed neighborhood in $G\Delta S$, and, hence, are contained in the same maximal cliques in $G\Delta S$. Thus, the following lemma is a stronger claim than Guo's corresponding result [17, Lemma 2], which says that for a specific critical clique K of the input graph G there exists an optimal solution $S \subseteq E$ such that K is part of a critical clique in $G\Delta S$.

Lemma 6. *There is an optimal edge modification set $S \subseteq E$ such that every critical clique K of G is part of a critical clique in $G\Delta S$.*

Proof. Let $K_1, \ldots, K_\ell$ denote the critical cliques of G and let $S' \subseteq E$ denote an optimal edge deletion set. We show that one can transform S' into an edge deletion set S with $|S| \leq |S'|$ such that for every i, $1 \leq i \leq \ell$, there is a critical clique K' in $G\Delta S$ with $K_i \subseteq K'$.

If each K_i is contained in a critical clique in $G\Delta S'$, then the lemma trivially holds for $S = S'$. Hence, in the following we consider the case that there is an i, $1 \leq i \leq \ell$, such that K_i is not contained in a critical clique in $G\Delta S'$. We show that we can iteratively apply the following "local modification" until each K_i is contained in a critical clique. Consider an arbitrary critical clique K_i such that K_i is not contained in one critical clique in $G\Delta S'$. Let $A_1, \ldots, A_p$ denote the critical cliques in $G\Delta S'$ with $A_j \cap K_i \neq \emptyset$ for all $1 \leq j \leq p$. Note that $p > 1$. Let $B_j := A_j \cap K_i$ for all $1 \leq j \leq p$. Moreover, let $v \in K_i$ denote a vertex such that $S'_v := S' \cap \{\{v, w\} \mid w \in V \setminus K_i\}$ has minimum cardinality among all vertices in K_i and without loss of generality let $v \in B_1$. Transform $G\Delta S'$ as follows into a graph $G' = (V, E')$. In G', K_i is a clique and $N_{G'}(K_i) = N_{G\Delta S'}(B_1) \setminus K_i$, that is, undo all edge deletions between the vertices in K_i, and, for each vertex $x \in K_i \setminus B_1$, delete the edges between x and $N_{G\Delta S'}(x) \setminus (K_i \cup N_{G\Delta S'}(B_1))$ and undo the edge deletions between x and $N_{G\Delta S'}(B_1) \setminus K_i$ (note that in G we have $\{x, y\} \in E$ for each $y \in N_{G\Delta S'}(B_1) \setminus K_i$). Observe that, by the choice of v, for each x the number of edge deletions is at most the number of edges deletions that are undone. Thus, in G', every vertex in K_i is incident to at most $|S'_v|$ edge modifications, and, hence, the edit distance of G' to G is at most the edit distance of $G\Delta S'$ to G. Moreover, by construction, all vertices in K_i have an identical closed neighborhood, and, hence, K_i is part of a critical clique in G'.

Next, we show the correctness of this transformation, that is, we show that G' has the 2-vertex-overlap property. Then, we show that the modification does not increase the number of critical cliques intersecting with K_j ($j \neq i$). For both proofs the following interpretation is helpful. Observe that the transformation from $G\Delta S'$ to G' can be seen as follows. First, all vertices in $K_i \setminus B_1$ are deleted. Then they are added one-by-one, making each vertex adjacent to each vertex in the (current) closed neighborhood of v. Next, we show that G' has the 2-vertex-overlap property. To this end, first note that since the 2-vertex-overlap property is hereditary (see Lemma 1) deleting a vertex does not destroy the 2-vertex-overlap property. Moreover, adding a vertex x and making it adjacent to the vertices in the closed neighborhood of an existing vertex v does not destroy the 2-vertex-overlap property.

Finally, we show that in G' the number of critical cliques intersecting with K_j ($j \neq i$) does not increase. To this end, recall the alternative interpretation of the transformation (that is, first deleting the vertices in $K_i \setminus B_1$ and subsequently adding these vertices and making them adjacent to the vertices in the closed neighborhood of v). Since two vertices with an identical neighborhood have an identical neighborhood after the deletion of a third vertex, deleting a vertex does not increase the number of critical cliques. Moreover, adding a vertex and making it adjacent to each vertex in the closed neighborhood of an existing vertex does not change the critical cliques of the graph (except for the critical clique containing the new vertex whose size is increased by one).

In summary, after the modification K_i is contained in one critical clique of G' and for every other critical clique K_j, the number of critical cliques of G' intersecting with K_j does not increase. Moreover, the resulting graph G' has the 2-vertex-overlap property and the edit distance of G' to G is at most the edit distance of $G \Delta S'$ to G. Hence, we can apply the local modification described above until every critical clique is contained in one critical clique of the resulting graph. □

According to Lemma 6, if one deletes an edge incident to a vertex in a critical clique, then one has to delete an edge for every vertex in this critical clique. Hence, if there exists a critical clique of size greater than $k+1$, then we need to keep only $k+1$ of these vertices, because at most k edges may be deleted. Therefore, Rule 1 is correct. Note that all critical cliques of a graph can be found in linear time [25]. Hence, Rule 1 can be carried out in $O(m+n)$ time.

Rule 2. If there exists a satisfied maximal clique K such that all vertices in $N(K)$ are satisfied, then remove every edge e such that K is the only maximal clique containing e.

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

Lemma 7. *Let K and K' be two maximal cliques with $K \cap K' \neq \emptyset$. If all vertices in $K \cap K'$ are satisfied, then*

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

Proof. First, we prove part 1. Assume towards a contradiction that there exist two vertices $v \in K \setminus K'$ and $u \in K' \setminus K$ such that v and u are adjacent. Let $x \in K \cap K'$. 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 $u \notin K$) nor K' (since $v \notin K'$). Hence, x is contained in at least three maximal cliques, a contradiction to the fact that all vertices in $K \cap K'$ are satisfied.

Next, we prove part 2. Assume towards a contradiction that there exists a maximal clique K'' with $K \cap K'' \neq \emptyset$ and $K' \cap K'' \neq \emptyset$. Since the vertices in $K \cap K'$ are satisfied, K'' intersects with K and K' only in $K \setminus K'$ and $K' \setminus K$, respectively. Hence, there exists a vertex in $K \setminus K'$ and a vertex in $K' \setminus K$ both contained in K'', a contradiction to part 1 of the lemma. □

Lemma 8. *Let K be a satisfied maximal clique in G. If there exists a vertex $v \in K$ such that $N[v] = K$, then there exists an optimal solution S such that v is contained in exactly one maximal clique in $G \Delta S$.*

Proof. Let S be an optimal solution for G such that every critical clique of G is part of a critical clique in $G_S := G \Delta S$. By Lemma 6, such a solution must exist.

Assume towards a contradiction that v is contained in two maximal cliques K_1 and K_2 in G_S. Let W denote the set of vertices in the connected component of $G_S[K]$ containing v and let $X \subseteq S$ denote the edge deletions between vertices of W. Note that $X \neq \emptyset$ since $K_1 \cup K_2 \subseteq W$. We show that $S' := S \setminus X$ is a solution, which contradicts the optimality of S. More precisely, we show that in $G_{S'} := G \Delta S'$ every vertex is satisfied.

Since in $G_{S'}$ there is no edge $\{x, y\}$ with $x \in W$ and $y \in K \setminus W$ (otherwise, y would be in a connected component with v in G_S) it holds that $N_{G_{S'}}(W) \subseteq V \setminus K$. Moreover, since we only undo edge deletions between vertices of W it suffices to show that the vertices in $N_{G_{S'}}(W) \cup W$ are satisfied (for all other vertices the graph induced by their closed neighborhood does not change).

First, consider a vertex $u \in N_{G_{S'}}(W)$. Recall that $u \in V \setminus K$. Let $B := K \cap N_G(u)$. Observe that since K is satisfied B is a critical clique in G. Hence, by Lemma 6, B is part of a critical clique in G_S and u is adjacent to all vertices of B (that is, $B = N_{G_S}(u) \cap W$). Clearly, this implies that $B = N_{G_{S'}}(u) \cap W$. Thus, the graphs induced by the closed neighborhoods of u in G_S and $G_{S'}$ are identical. Hence, u is satisfied.

Second, consider a vertex $w \in W$. We argue that w is contained in a maximal clique completely contained in W. Let B denote the critical clique of G containing w. Note that $B \subseteq K$. By Lemma 6 it follows that $B \subseteq W$ and that all vertices in B have an identical closed neighborhood in G_S. Since W contains more than one critical clique in G_S (note that v is contained in two maximal cliques in G_S), by definition of W there exists a vertex $x \in W \setminus B$ adjacent to w in G_S. Let Q denote a maximal clique of G_S with $\{x, w\} \subseteq Q$. We show that $Q \subseteq W$. Assume towards a contradiction that $Q \setminus W \neq \emptyset$ and let $z \in Q \setminus W$. Let $B' := N_G(z) \cap K$. Since K is satisfied, B' forms a critical clique in G. Moreover, since $w \in B'$ we have $B' = B$, contradicting the fact that $x \in W \setminus B$. Hence, there exists a maximal clique Q in G_S contained in W. This means that there exists at most one further maximal clique K' in G_S containing w and vertices from $V \setminus K$. Hence, w is in $G_{S'}$ contained in at most two maximal cliques, namely W and K' (if existent). □

Lemma 9. *Rule 2 is correct and can be carried out in $O(m \cdot n)$ time.*

Proof. Let $G = (V, E)$ be a graph containing a satisfied maximal clique K such that all vertices in $N_G(K)$ are satisfied. Moreover, let $G' = (V', E')$ denote the graph that results from removing all edges contained only in K.

To show the correctness of Rule 2, we need the following. Let $\mathcal{B} := \{B_1, \ldots, B_\ell\}$ denote the critical cliques of G contained in K. Note that, since K is satisfied, for every B_i there exists at most one further maximal clique K_i in G with $B_i \subseteq K_i$. Furthermore, by Lemma 7, it follows that the K_i's are pairwise vertex-disjoint.

Claim: (G, k) is a yes-instance if and only if (G', k) is a yes-instance.

"$\Rightarrow$": Let S be an optimal solution of size at most k for G and let $G_S := G \Delta S$. We show that $S' := S \setminus E_K$ is a solution for G', where E_K denotes the set of all possible edges between two vertices of K. Let $G'_{S'} := G' \Delta S'$. According to

Lemma 6, we can assume that every B_i is completely contained in at most two maximal cliques in G_S. In particular, this means that S does not contain edge deletions between two vertices of the same B_i. Hence, $G'_{S'}$ differs from G_S in that all edges between different B_i's are deleted. Moreover, every vertex $x \in V \setminus K$ being 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 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 identical. Hence, these vertices are satisfied and it remains to show that all vertices in the B_i's are satisfied.

To this end, we argue that 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 edge between B_i and $K \setminus B_i$ in G_S since otherwise B_i would be contained in three maximal cliques (note that by Lemma 7 there is no edge between $C_i^j \setminus K$ and $K \setminus C_i^j$, $j \in \{1, 2\}$). 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 $C_i^1 \cap C_i^2 = B_i$, since a vertex in $C_i^2 \setminus B_i$ 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 and $K \setminus B_i$, the vertices of B_i are contained in exactly one maximal clique, namely $C_i^2 \subseteq K_i$. Second, for the case that B_i is contained in exactly one maximal clique in G_S, the argumentation works in analogy. In summary, $G'_{S'}$ fulfills the 2-vertex-overlap property.

"⇐": Let S denote an optimal solution of size at most k for G', and let $G'_S := G' \Delta S$. We show that S is solution for G as well, that is, we show that in $G_S := G \Delta S$ every vertex is satisfied. Note that for every B_i every vertex of B_i is contained in exactly one satisfied maximal clique in G', namely K_i. Thus, by Lemma 8, we can assume that every B_i is completely contained in exactly one satisfied maximal clique $K'_i \subseteq K_i$ in G'_S. Further, recall that all K'_i's are pairwise vertex-disjoint and every vertex in $K'_i \setminus B_i$ is adjacent in G'_S to all vertices in B_i but not to any other vertex in K. Hence, if we add all missing edges between the vertices of K in G'_S (resulting in G_S), then none of the added edges is between two neighbors of vertices in $K'_i \setminus K$. Hence, these vertices are satisfied in G_S. Moreover, since these are the only vertices in $V \setminus K$ having in G'_S at least two neighbors in K, all vertices in $V \setminus K$ are satisfied in G_S. Finally, all vertices in K are clearly contained in at most two maximal cliques in G_S, namely in K and in at most one further clique $K'_i \subseteq K_i$ (note that, in G_S, the common neighborhood for two vertices from different B_i's is K).

For the running time, note that one can compute the set U of all satisfied vertices in $O(m \cdot n)$ time as follows. For each $v \in V$, build $G[N[v]]$ and then check in $O(|N[v]|^2)$ time whether $G[N[v]]$ contains at most two maximal cliques. The running time for computing U hence sums up to

$$O(\sum_{v \in V} \deg(v)^2) = O(n \cdot \sum_{v \in V} \deg(v)) = O(n \cdot m).$$

After that, 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 $O(|N[v]|^2)$ time for every $u \in U$. Finally, check in $O(m)$ time whether K_1 or K_2 fulfills the precondition of Rule 2. Hence, the overall running time for one application of Rule 2 is bounded by $O(n \cdot m + \sum_{u \in U}(\deg(v)^2 + m)) = O(m \cdot n)$. □

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

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

Lemma 10. *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 Conditions 1 and 2 of Rule 3 and suppose that $\sum_{i=1}^{\ell} |K_i \cap K| \geq 2k + 2$. If (G, k) is a yes–instance, then there exists an optimal solution of size at most k not deleting any edge between vertices of K.*

Proof. Suppose that there exists an optimal solution S of size at most k for G and let $G_S := G \Delta S$. Assume towards a contradiction that S contains an edge $\{v, w\}$ with $v, w \in K$. In the following, we refer by $\{u_1, \ldots, u_t\}$ to the vertices in $\bigcup_{i=1}^{\ell}(K_i \cap K)$. Since all K_i's are satisfied, according to Lemma 7 the K_i's are pairwise vertex-disjoint. Because $t \geq 2k + 2$, one of the u_i's is non-affected by S. Without loss of generality, assume that u_1 is one of these non-affected vertices and $u_1 \in K_1$. Let $B_1 := K \cap K_1$. Clearly, B_1 is a critical clique in G. By Lemma 6, we have that B_1 is (part of) a critical clique in G_S, and, hence, all vertices in B_1 are non-affected. This implies that neither v nor w is contained in B_1. Let $z \in K_1 \setminus K$. Since u_1 is non-affected by S, this implies $\{z, u_1\} \in E(G_S)$. Moreover, by Lemma 7 (and Condition 2 of Rule 3), it follows that $\{z, v\}$ and $\{z, w\}$ are not contained in E and hence not in $E(G_S)$. This implies that u_1 is contained in at least three maximal cliques in G_S: the vertices u_1, v, w, and z induce a star with center vertex u_1 and three leaves (see first graph in Figure 2). This is a contradiction to the fact that S is a solution. □

Lemma 11. *Rule 3 is correct and can be carried out in $O(m \cdot n)$ time.*

Proof. Let $G = (V, E)$, K, and $K_1, \ldots, K_\ell$ be as described in Rule 3. Furthermore, let $B_i := K_i \cap K$ for every $1 \leq i \leq \ell$. Again, since all K_i's are satisfied, the B_i's are critical cliques and according to Lemma 7 the K_i's are pairwise vertex-disjoint. Let $G' = (V, E')$ be the graph resulting from one application of Rule 3. We show the following.

Claim: (G, k) is a yes–instance if and only if (G', k) is a yes–instance.

"$\Rightarrow$": Let S denote an optimal solution of size at most k for G and let $G_S := G \Delta S$. We show that S is a solution for G'. Let $G'_S := G' \Delta S$. By Lemma 10, S does not delete any edge within K. Together with Lemma 6 this implies that in G_S, B_1 is contained in K and in at most one further maximal clique $K'_1 \subseteq K_1$. Note that G'_S results from G_S by deleting all edges between B_1 and $K \setminus B_1$. Since by Lemma 7, there is no edge between $K'_1 \setminus K$ and $K \setminus K'_1$, this does not create any unsatisfied vertices.

"$\Leftarrow$": Let S' denote an optimal solution of size at most k for G' and let $G'_{S'} := G' \Delta S'$. We show that in $G_{S'} := G \Delta S'$ all vertices are satisfied. Note that in G', K_1 forms a clique whose vertices are all satisfied and that the vertices in B_1 are contained in exactly one maximal clique, namely K_1. Hence, according to Lemma 8, we can assume that B_1 is contained in exactly one maximal clique $K'_1 \subseteq K_1$ in $G'_{S'}$. Moreover, note that for every $1 \leq i \leq \ell$, since G is reduced with respect to Rule 1 and since B_i is a critical clique in G, it holds that $|B_i| \leq k + 1$. In particular, $|B_1| \leq k + 1$ and since $\sum_{i=1}^{\ell} |B_i| \geq 3k + 4$, we have $\sum_{i=2}^{\ell} |B_i| \geq 2k+2$ and $\ell \geq 3$. Hence, $\sum_{i=2}^{\ell} |(K \setminus B_1) \cap K_i| \geq 2k+2$. Moreover, it is not hard to verify that $K \setminus B_1$ forms a maximal clique in G'. Thus, by Lemma 10, S' does not delete any edge between two vertices from $K \setminus B_1$. Hence, $K \setminus B_1$ is a maximal clique in $G'_{S'}$. Note that $G_{S'}$ results from $G'_{S'}$ by inserting all edges between a vertex in B_1 and the vertices in $K \setminus B_1$. Clearly, this does not change the number of maximal cliques for a vertex in $V \setminus K$, since, by Lemma 7, none of these has neighbors in both B_1 and $K \setminus B_1$. Finally, all vertices in K clearly are satisfied.

For the running time note the following. First, as argued in the proof of Lemma 9, we can compute the set U of all satisfied vertices in $O(n \cdot m)$ time. Hence, in the following we assume that for each vertex in the graph, we can determine in $O(1)$ time whether it is satisfied or not. Then, for every vertex $u \in U$ we proceed as follows. Vertex u is contained in at most two maximal cliques K' and K''. These two cliques can be computed in $O(\deg(u)^2)$ time. Next, we check whether K' and K'' can play the role of K and K_1 in Rule 3. Consider the case that $K = K'$ and $K_1 = K''$. Clearly, we can check in $O(\deg(u))$ time whether all vertices in K'' are satisfied. It remains to verify that there are at least $3k+4$ vertices in the intersections of satisfied maximal cliques with K'. We argue that this is possible in $O(m)$ time. We first label all vertices in K' that are contained in exactly two maximal cliques by '+'. All other vertices in K' are labeled by '−'. Next, we iterate over the edge set. For an edge $\{x, y\} \in E$ if $y \notin K'$ and not satisfied and x is labeled '+' then mark x with '−'. After that, if a satisfied vertex $v \in K'$ is contained in a second maximal clique containing non-satisfied vertices, then this vertex clearly is labeled '−'. Hence, all vertices labeled by '+' are contained in the intersections of satisfied maximal cliques with K. Thus, to check whether Rule 3 can be applied we just need to count the number of '+'-vertices in K. In summary, the overall running time is $O(m \cdot n + \sum_{u \in U}(\deg(u)^2 + m)) = O(m \cdot n)$. □

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

Theorem 9. 2-Vertex-Overlap Deletion *admits a problem kernel with* $O(k^3)$ *vertices.*

Proof. Let $G = (V, E)$ be a graph reduced with respect to Rules 1–4. We show that if G has a solution of size at most k, then the number of vertices of G is $O(k^3)$.

Assume that G has a solution S of size at most k and let $G_S := G \Delta S$. Further, let X denote the vertices affected by S and let $Y := V \setminus X$. First, note that $|X| \leq 2k$. Hence, it remains to show $|Y| = O(k^3)$.

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. Furthermore, define $K'_i := K_i \cap Y$, $1 \leq i \leq t$ and let $Z := \{Z_1, \ldots, Z_q\}$ denote the set of all other maximal cliques of G_S. For every $1 \leq i < j \leq t$ let $K'_{i,j} := K'_i \cap K'_j$. Note that every $K'_{i,j}$ is part of a critical clique in G_S, since it belongs to two maximal cliques. Furthermore, since the vertices in $K'_{i,j}$ are non-affected, they are also part of a critical clique 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, $|K'_{i,cc}| \leq k + 1$. Finally, let $A_i := K'_i \setminus (K'_{i,cc} \cup \bigcup_{j \neq i} K'_{i,j})$ denote the other vertices of K'_i. Note that $A_i \subseteq \bigcup_{i=1}^{q} Z_i$.

Next, we show that

a) every vertex in A_i is contained in at most two maximal cliques in G,

b) for $1 \leq j \leq q$, every vertex in Z_j is contained in at most two maximal cliques in G,

c) for $1 \leq j \leq q$, every Z_j has a nonempty intersection with at least one A_i, $1 \leq i \leq t$,

d) for $1 \leq \ i \leq t$, $|A_i| \leq 3k + 3$, and

e) $q \leq (3k+4) \cdot 4k$ and, for $1 \leq j \leq q$, $|I_j| \leq 4k+4$, with $I_j := Z_j \setminus (\bigcup_{i=1}^{t} A_i)$.

a) Consider an arbitrary vertex $y \in A_i$. Note that y is adjacent in G_S only to the vertices $K_i \setminus K'_i$ of X. 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, therefore, y is contained in the same number of maximal cliques in G as in G_S.

b) A vertex $y \in Z_j$ is either contained in A_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 of Z_j's neighbors are satisfied and, as a consequence, Rule 2 applies, contradicting the fact that G is reduced.

d) Assume that there exists an i with $|A_i| \geq 3k+4$. Without loss of generality, let $Z_1, \ldots, Z_p$ be the sets in Z intersecting with A_i. Hence, $A_i \subseteq \bigcup_{j=1}^{p} Z_j$ (recall that $A_i \subseteq \bigcup_{j=1}^{q} Z_j$) and as a consequence $|A_i| = \sum_{j=1}^{p} |Z_j \cap A_i| \geq 3k+4$. Moreover, according to b), all Z_j's are satisfied. Thus, Rule 3 applies to a maximal clique K with $A_i \subseteq K$ in G, contradicting the fact that G is reduced.

e) First, since every Z_j has nonempty intersection with some A_i and since any other Z_h, $h \neq j$, cannot intersect with A_i in the same vertices as Z_j, it follows that $|Z| \leq (3k+3) \cdot 4k$. Second, assume that there exists an I_j with $|I_j| > 4k+4$. Since G is reduced with respect to Rule 1, there are at most $k+1$ vertices in I_j that are contained in the single maximal clique Z_j (these vertices form a critical clique in G). All other vertices of I_j are contained in some Z_h with $h \neq j$. Let $Z'_1, \ldots, Z'_p$ denote the sets in Z having nonempty intersection 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 applies, contradicting the fact that G is reduced.

Putting everything together, one obtains

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

□

7. Conclusion

We have provided here a first theoretical study of a set of new cluster graph modification problems motivated by the practical relevance of clustering with overlaps [11, 28]. Naturally, studying a set of problems that is so far barely explored, there remain many challenges for future work. We list only a few of them. First, it is 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 (like those undertaken for Cluster Editing, see [6, 11]) are a natural next step. Fourth, the polynomial-time approximability of our problems remains unexplored. Fifth and finally, it seems promising to study overlaps 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 (as recently considered in the context of clustering without overlaps [18, 19, 21]).

Acknowledgement. We are grateful to anonymous referees of *Discrete Optimization* for feedback improving our presentation.

References

[1] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56(1–3):89–113, 2004.

[2] J.-P. Barthélemy and F. Brucker. NP-hard approximation problems in overlapping clustering. *Journal of Classification*, 18(2):159–183, 2001.

[3] A. Ben-Dor, R. Shamir, and Z. Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3/4):281–292, 1999.

[4] S. Böcker, S. Briesemeister, Q. B. A. Bui, and A. Truß. A fixed-parameter approach for weighted cluster editing. In *Proc. 6th APBC*, volume 5 of *Advances in Bioinformatics and Computational Biology*, pages 211–220. Imperial College Press, 2008.

[5] S. Böcker, S. Briesemeister, Q. B. A. Bui, and A. Truß. Going weighted: Parameterized algorithms for cluster editing. *Theoretical Computer Science*, 410(52):5467–5480, 2009.

[6] S. Böcker, S. Briesemeister, and G. W. Klau. Exact algorithms for cluster editing: Evaluation and experiments. *Algorithmica*, 2009. To appear, electronically available.

[7] H. L. Bodlaender. Kernelization: New upper and lower bound techniques. In *Proc. 4th IWPEC*, volume 5917 of *LNCS*, pages 17–37. Springer, 2009.

[8] L. Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Information Processing Letters*, 58(4):171–176, 1996.

[9] J. Chen and J. Meng. A $2k$ kernel for the cluster editing problem. In *Proc. 16th COCOON*, volume 6196 of *LNCS*, pages 459–468. Springer, 2010.

[10] P. Damaschke. Fixed-parameter enumerability of cluster editing and related problems. *Theory of Computing Systems*, 46(2):261–283, 2010.

[11] 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.

[12] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Springer, 1999.

[13] 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.

[14] J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[15] J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier. Graph-modeled data clustering: Exact algorithms for clique generation. *Theory of Computing Systems*, 38(4):373–392, 2005.

[16] 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.

[17] J. Guo. A more effective linear kernelization for Cluster Editing. *Theoretical Computer Science*, 410(8-10):718–726, 2009.

[18] J. Guo, I. A. Kanj, C. Komusiewicz, and J. Uhlmann. Editing graphs into disjoint unions of dense clusters. In *Proc. 20th ISAAC*, volume 5878 of *LNCS*, pages 583–593. Springer, 2009.

[19] J. Guo, C. Komusiewicz, R. Niedermeier, and J. Uhlmann. A more relaxed model for graph-based data clustering: s-plex editing. In *Proc. 5th AAIM*, volume 5564 of *LNCS*, pages 226–239. Springer, 2009. Long version to appear in *SIAM Journal on Discrete Mathematics*.

[20] J. Guo and R. Niedermeier. Invitation to data reduction and problem kernelization. *ACM SIGACT News*, 38(1):31–45, 2007.

[21] P. Heggernes, D. Lokshtanov, J. Nederlof, C. Paul, and J. A. Telle. Generalized graph clustering: recognizing (p, q)-cluster graphs. In *Proc. 36th WG*, LNCS. Springer, 2010. To appear.

[22] 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.

[23] M. Křivánek and J. Morávek. NP-hard problems in hierarchical-tree clustering. *Acta Informatica*, 23(3):311–323, 1986.

[24] 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.

[25] R. M. McConnell and J. Spinrad. Linear-time modular decomposition and efficient transitive orientation of comparability graphs. In *Proc. 5th SODA*, pages 536–545. ACM/SIAM, 1994.

[26] 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.

[27] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Oxford University Press, 2006.

[28] 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.

[29] R. Peeters. The maximum edge biclique problem is NP-complete. *Discrete Applied Mathematics*, 131(3):651–654, 2003.

[30] 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.

[31] S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007.

[32] R. Shamir, R. Sharan, and D. Tsur. Cluster graph modification problems. *Discrete Applied Mathematics*, 144(1–2):173–182, 2004.

[33] 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.

[34] M. Talmaciu and E. Nechita. Recognition algorithm for diamond-free graphs. *Informatica*, 18(3):457–462, 2007.

[35] 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.

[36] R. Xu and D. Wunsch II. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005.