Alternative Parameterizations for Cluster Editing[⋆]

Christian Komusiewicz and Johannes Uhlmann

Institut für Informatik, Friedrich-Schiller-Universität Jena,
Ernst-Abbe-Platz 2, D-07743 Jena, Germany.
[c.komus,johannes.uhlmann]@uni-jena.de

Abstract. Given an undirected graph G and a nonnegative integer k, the NP-hard Cluster Editing problem asks whether G can be transformed into a disjoint union of cliques by applying at most k edge modifications. In the field of parameterized algorithmics, Cluster Editing has almost exclusively been studied parameterized by the solution size k. Contrastingly, in many real-world instances it can be observed that the parameter k is not really small. This observation motivates our investigation of parameterizations of Cluster Editing different from the solution size k. Our results are as follows. Cluster Editing is fixed-parameter tractable with respect to the parameter "size of a minimum cluster vertex deletion set of G", a typically much smaller parameter than k. Cluster Editing remains NP-hard on graphs with maximum degree six. A restricted but practically relevant version of Cluster Editing is fixed-parameter tractable with respect to the combined parameter "number of clusters in the target graph" and "maximum number of modified edges incident to any vertex in G". Many of our results also transfer to the NP-hard Cluster Deletion problem, where only edge deletions are allowed.

1 Introduction

The NP-hard Cluster Editing problem is among the best-studied parameterized problems. It is usually defined as follows:

Input: An undirected graph $G = (V, E)$ and an integer $k \geq 0$.
Question: Can G be transformed into a cluster graph by applying at most k edge modifications?

Herein, an *edge modification* is either the deletion or insertion of an edge and a *cluster graph* is a graph where every connected component is a clique. The cliques of a cluster graph are referred to as *clusters*. Cluster Deletion is defined analogously except that only edge deletions are allowed.

So far, the proposed fixed-parameter algorithms for Cluster Editing almost exclusively examine the parameter solution size k. While several algorithmic improvements have led to impressive theoretical results, it has been observed

[⋆] Supported by the DFG, research project PABI NI369/7.

that the parameter k is often not really small for real-world instances [3]. Still, the fixed-parameter algorithms can solve many of these instances [3], which raises the question whether there are "hidden parameters" that are implicitly exploited by these algorithms. In the spirit of multivariate algorithmics (cf. [15]), this work aims at identifying promising new parameterizations for Cluster Editing that help to separate easy from hard instances.

Related Work. The NP-hardness of Cluster Editing has been shown several times [14, 17, 1]. The problem remains NP-hard even when the solution may contain at most two clusters [17]. The parameterized complexity of Cluster Editing with respect to the parameter k has been extensively studied. After a series of improvements [10, 9, 16, 11, 2, 4], the currently fastest fixed-parameter algorithm for this parameter has running time $O(1.82^k + n^3)$ [2] and the currently smallest problem kernel contains at most $2k$ vertices [4]. Several experimental studies demonstrate that fixed-parameter algorithms can be applied to solve real-world instances of Cluster Editing [6, 3]. Further theoretical studies have dealt with the parameterized complexity of different generalizations of Cluster Editing [5, 12, 8] for example by replacing the clique requirement in the cluster graph with other models for dense graphs. The problem to transform a graph into a cluster graph by a minimum number of vertex deletions is called Cluster Vertex Deletion(CVD). Hüffner et al. [13] presented an $O(2^k k^9 + nm)$-time iterative compression algorithm for CVD.

Our Results. Motivated by the observation that the parameter k is often very large in practice and subsequent calls for "better parameterizations" [7], we consider new parameters for Cluster Editing. Answering an open question by Dehne [7], we show that Cluster Editing is fixed-parameter tractable with respect to the parameter "cluster vertex deletion number" c of the input graph G. Moreover, we consider the parameter t denoting the "maximal number of modified edges incident to any vertex". First, we show that Cluster Editing is NP-hard for maximum degree-six graphs. Since in an optimal solution the number of incident edge modifications of a vertex is bounded by its degree, there is no hope for fixed-parameter tractability with respect to t. However, we can show that a restricted version of Cluster Editing is fixed-parameter tractable when combining t with the parameter d denoting the number of cliques in the final cluster graph. Our methods include enumerative approaches, matching techniques, and problem kernelization.

Due to lack of space, several proofs are deferred to a full version of the paper.

Preliminaries. Given a graph $G = (V, E)$, we use $V(G)$ to denote the vertex set of G and $E(G)$ to denote the edge set of G. Let $n := |V|$ and $m := |E|$. The (open) neighborhood $N(v)$ of a vertex v is the set of vertices adjacent to v, and the closed neighborhood is $N[v] := N(v) \cup \{v\}$. For a vertex set W let $E_W := \{\{v, w\} \mid \{v, w\} \subseteq W\}$ denote the set of all size-two subsets of W. We use $G[V']$ to denote the subgraph of G induced by $V' \subseteq V$, that is, $G[V'] := (V', E_{V'} \cap E)$. Moreover, $G - v := G[V \setminus \{v\}]$ for a vertex $v \in V$ and $G - e := (V, E \setminus \{e\})$.

Let $E \Delta F := (E \setminus F) \cup (F \setminus E)$ denote the symmetric difference of two sets E and F. The *edit distance* between two graphs G_1 and G_2 on the same vertex set is $|E(G_1) \, \Delta \, E(G_2)|$. Given three graphs G_1, G_2, and G_3, we say that G_2 is *closer* to G_1 than G_3 if the edit distance between G_1 and G_2 is strictly smaller than the edit distance between G_1 and G_3. For a graph $G = (V, E)$ and a set $S \subseteq E_V$ let $G \Delta S := (V, E \Delta S)$ denote the graph that results by modifying G according to S. A set of pairwise adjacent vertices is called *clique*.

2 Cluster Vertex Deletion Number

In this section, we present fixed-parameter algorithms for Cluster Editing (CE) and Cluster Deletion (CD) parameterized by the size of a minimum-cardinality vertex set Y such that removing Y from the input graph G results in a cluster graph. In the following, we will refer to this parameter as cluster vertex deletion number c of G. Note that c is bounded from above by the size k of a minimum-cardinality edge-modification set: deleting for each edge modification one of the two vertices (arbitrarily chosen) clearly results in a cluster graph. Our algorithms solve the optimization version of CE and CD, that is, they find a minimum-cardinality edge modification or edge deletion set, respectively. Both algorithms make use of the following observation for cliques that are large in comparison to the size of their neighborhood. These cliques are preserved to large extent by any optimal solution for CE or CD.

Lemma 1. *Let K denote a clique in G of size at least $2 \cdot |N_G(K)|$. Then, for every optimal edge modification set (optimal edge deletion set) S, the graph $G \, \Delta \, S$, contains a cluster K' with*

$$|K \cap K'| \geq |K| - 2|N_G(K)|.$$

Proof. We show the lemma for the case of an optimal edge modification set S. Let $K'_1, \ldots, K'_\ell$ denote the clusters in $G \, \Delta \, S$ with $K'_i \cap K \neq \emptyset$. Furthermore, define $B_i := K'_i \cap K$ and observe that $K = \bigcup_{i=1}^{\ell} B_i$. Note that in the case that $|K| \leq 2|N_G(K)| + 1$ the lemma trivially holds since $|B_1| \geq 1$ and $|K| - 2|N_G(K)| \leq 1$.

Assume towards a contradiction that all B_i's contain less than $|K| - 2|N_G(K)|$ vertices. This implies that—in order to separate the B_i's from each other—the solution contains at least

$$1/2 \sum_{i=1}^{\ell} |B_i|(|K| - |B_i|) > 1/2 \sum_{i=1}^{\ell} (|B_i| \cdot 2|N_G(K)|) = |N_G(K)| \cdot |K|$$

edge deletions. Hence, one obtains a cluster graph that is closer to G by deleting in $G \, \Delta \, S$ all edges between K and $N(K)$ (at most $|N(K)| \cdot |K|$) and undoing all edge deletions between vertices in K (at least $|N(K)| \cdot |K| + 1$); a contradiction to the fact that S is optimal. It is easy to verify that all steps of the proof hold for an optimal edge deletion set, too. □

Next, we present our fixed-parameter algorithm for Cluster Editing.

2.1 Cluster Editing

Given an input graph G and a size-c cluster vertex deletion set Y of G, a key observation used by our algorithm is that clusters in $G-Y$ that are much larger than Y will not be split by any optimal edge modification set. Recall that an isolated clique of a cluster graph is called a *cluster*.

Lemma 2. *Let Y denote a size-c cluster vertex deletion set and let K denote a cluster in $G-Y$ of size at least $3c+1$. Then, for every optimal edge modification set S, the graph $G \,\Delta\, S$ contains a cluster K' with*

1. *$K \subseteq K'$ and*
2. *$K' \subseteq K \cup Y$.*

Proof. Let $K_1, \ldots, K_\ell$ ($\ell \geq 1$) denote the clusters in $G \,\Delta\, S$ with $K_i \cap K \neq \emptyset$. Let $B_i := K \cap K_i$ and note that $K = \bigcup B_i$. Without loss of generality, assume that B_1 has maximum size of all B_i's. Since $N_G(K) \subseteq Y$ and by Lemma 1, $|B_1| \geq c+1$. First, we show that $K_1 \setminus B_1 \subseteq Y$. Let $X := K_1 \setminus (B_1 \cup Y)$ and assume towards a contradiction that $X \neq \emptyset$. Since $|B_1| \geq c+1$ one obtains a cluster graph that is closer to G than $G \,\Delta\, S$ by making X an isolated clique, which requires at most $|X| \cdot c$ additional edge deletions, however, allows one to undo the edge insertions between B_1 and X which amount to at least $|X| \cdot (c+1)$.

Next, we prove that $\ell = 1$, directly implying the lemma. Assume towards a contradiction that $\ell > 1$. Since $|B_1| > c$ one obtains a cluster graph that is closer to G than $G \,\Delta\, S$ by deleting all edges between B_2 and $K_2 \setminus B_2$ (at most $|K_2 \cap Y| \cdot |B_2|$ additional edge deletions), inserting all edges between B_2 and $K_1 \setminus B_1$ (at most $|B_2| \cdot |K_1 \cap Y|$ edge insertions since $K_1 \setminus B_1 \subseteq Y$), and undoing the edge deletions between B_2 and B_1 (at least $|B_2| \cdot |B_1|$). Since $|B_2| \cdot |K_2 \cap Y| + |B_2| \cdot |K_1 \cap Y| \leq |B_2| \cdot c < |B_2| \cdot (c+1) \leq |B_2| \cdot |B_1|$, this is a contradiction to the fact that S is optimal. □

According to Lemma 2, a cluster in $G-Y$ of size at least $3c+1$ will not be split by any optimal edge modification set and also not "merged" with any other clusters of $G-Y$. We refer to these clusters of $G-Y$ as *large clusters*. The basic idea to establish fixed-parameter tractability is as follows (see Alg. 1 for an outline). Given a cluster vertex deletion set Y, in a first step, we guess the partition of Y "induced" by the clusters generated by an optimal edge modification set (Line 5 of Alg. 1). We refer to the sets of such a partition as *fixed subclusters* in the following. Then, in a second step, we guess which of these fixed subclusters will end up in a cluster together with a large cluster (Line 10 of Alg. 1). From Lemma 2, we know that the large clusters cannot be split, and since the subclusters in Y are fixed, the large clusters end up in a final cluster with at most one fixed subcluster. Hence, the "mapping" of the large clusters to the respective fixed subclusters can efficiently be done by computing a maximum weight matching. To this end, Alg. 1 employs the subroutine `SolveLarge` (see Line 13). The remaining instance is solved by the subroutine `SolveSmall`. This subroutine uses data reduction to bound the number of small clusters in $G-Y$ by a function

```
Function CEbyCVD($G$)
Input: A graph $G = (V, E)$
Output: Size of an optimal solution $S$ for Cluster Editing
$Y$ = SolveCVD (G) ;
Let $A_1, \ldots, A_p$ denote the clusters in $G - Y$ of size at most $3d$;
Let $B_1, \ldots, B_q$ denote the clusters in $G - Y$ of size at least $3d + 1$;
$m_1 = +\infty$;
forall partitions $Q_1, \ldots, Q_t$ of $Y$ ($1 \leq t \leq |Y|$) do
    Add all edges between vertices in $Q_i$, $1 \leq i \leq t$;
    Delete all edges between $Q_i$ and $Q_j$, $1 \leq i < j \leq t$;
    Let $c_1$ denote the number of these edge modifications;
    $m_2 = +\infty$;
    forall subsets $I \subseteq \{1, \ldots, t\}$ do
        Delete all edges between $Q_i$ and $A_j$, $i \in I$ and $1 \leq j \leq p$;
        Let $c_2$ denote the number of these edge deletions;
        $c_l$ =SolveLarge ($\{Q_i \mid i \in I\}$, $B_1, \ldots, B_p$) ;
        $c_s$ =SolveSmall ($\{Q_i \mid i \in \{1, \ldots, t\} \setminus I\}$, $A_1, \ldots, A_q$ );
        $m_2 = \min(m_2, c_2 + c_l + c_s)$;
    end
    $m_1 = \min(m_1, c_1 + m_2)$;
end
return $m_1$;
```

Algorithm 1: An algorithm to find an optimal solution for Cluster Editing.

only depending on c, thus yielding a problem kernel for this subproblem. This directly implies fixed-parameter tractability.

Next, we focus on the computation of an optimal solution for the subproblem that has to be solved by `SolveLarge`. Formally, we have to find a solution to the following problem Fixed Clique Cluster Editing: Let $G = (V, E)$ be a graph and let $B \uplus Q$ be a two partition of V such that $G[B]$ and $G[Q]$ are cluster graphs. Furthermore, let $\mathcal{B} = \{B_1, \ldots, B_q\}$ be the set of clusters in $G[B]$ and let $\mathcal{Q} = \{Q_1, \ldots, Q_s\}$ be the set of clusters in $G[Q]$. The task is to find a cluster graph G_c on V such that the following *cluster constraints* are fulfilled:

1. each set in $\mathcal{B} \cup \mathcal{Q}$ is a subset of a cluster of G_c,
2. for every $B_i \in \mathcal{B}$ ($Q_i \in \mathcal{Q}$) the cluster containing B_i (Q_i) does not contain any other set from $\mathcal{B}$ ($\mathcal{Q}$) and at most one set from $\mathcal{Q}$ ($\mathcal{B}$), and
3. among all such cluster graphs G_c has minimum edit distance to G.

This problem can be formulated as a bipartite maximum weight matching problem and, hence, can be solved in polynomial time.

Lemma 3. Fixed Clique Cluster Editing *can be solved in polynomial time.*

Next, we focus on the problem that has to be solved by `SolveSmall`, where all remaining clusters in $G - Y$ have size at most $3c$.

Lemma 4. *Let Y denote a cluster vertex deletion set for G of size c. If all clusters in $G - Y$ have size at most $3c$, then* CE *is fixed-parameter tractable.*

Proof. The basic idea to show fixed-parameter tractability is as follows. We group the clusters of $G - Y$ into different "types", where two clusters Q_i and Q_j of $G - Y$ have the same type if there is a graph-isomorphism ϕ between $G[Y \cup Q_i]$ and $G[Y \cup Q_j]$ such that $\forall v \in Y : \phi(v) = v$. We show that the number of types of the clusters in $G - Y$ is bounded by $2^{O(c^2)}$ and that if there is a type of which there are more than $O(c^2)$ clusters, then one of these clusters will be a cluster in the final cluster graph. Hence, we can delete all edges outgoing from this cluster and remove it from G. Afterwards, since each cluster contains at most $3c$ vertices the total number of vertices is bounded by a function only depending on c, directly implying fixed-parameter tractability.

To bound the number of cluster types, we first classify the vertices in $V \setminus Y$ in 2^c types; two vertices in $u, w \in V \setminus Y$ have the same type if $N_G(u) \cap Y = N_G(w) \cap Y$. Then, a cluster K can be described by a vector of length 2^c with at most $3c$ non-zero entries, where the ith entry contains the number of type-i vertices in K (note that the sum of entries does not exceed $3c$). Further, two clusters have the same type if these corresponding vectors are identical. Finally, observe that there are at most $\sum_{i=1}^{3c} \binom{3c \cdot 2^c}{i} \leq 3c \cdot (3c \cdot 2^c)^{3c} = 2^{O(c^2)}$ cluster types.

We now show that for each cluster type we can delete all but c^2 clusters. First, note that there are at most c clusters in a closest cluster graph that contain vertices from Y. Second, we can assume that each cluster of a closest cluster graph intersecting with Y contains vertices from at most c clusters of $G - Y$; it is straightforward to verify that otherwise separating the vertices of one cluster results in a cluster graph with the same or smaller edit distance to G. As a consequence, each of the at most c clusters intersecting with Y can contain vertices from at most c clusters of each type. Finally, note that if there is a cluster K of $G - Y$ such that in a closest cluster graph no vertex of K is in a cluster intersecting with Y, then K is a cluster of this closest cluster graph. Hence, if there are $c^2 + i$, $1 \leq i$, clusters of the same type in $G - Y$, then at least i of these clusters are clusters in a closest cluster graph, and, hence, can be deleted (together with the edges between these clusters and Y). After deleting these clusters there are at most $2^{O(c^2)} c^2 + c = 2^{O(c^2)}$ vertices in G. □

Using these two results on the running times of `SolveLarge` and `SolveSmall`, we can show the fixed-parameter tractability with respect to c.

Theorem 1. Cluster Editing *is fixed-parameter tractable with respect to the cluster vertex deletion number c of G.*

Proof. We use Alg. 1 to compute an optimal solution for CE. First, we show that Alg. 1 is correct. To this end, let G' denote a cluster graph with closest edit distance to G. Since Alg. 1 enumerates all partitions of Y, the sets $Q_1, \ldots, Q_t$ (Line 5) once one-to-one correspond to the clusters in $G'[Y]$. By Lemma 2, all clusters of size at least $3c + 1$ ("large cluster") in $G - Y$ either form a cluster in G' or are contained in a cluster of G' together with vertices from Y. Since the algorithm has already guessed the partition of Y, every such large cluster is contained in a cluster of G' with the vertices of at most one Q_i. By trying all two-partitions of $\{1, \ldots, t\}$, Alg. 1 guesses the Q_i's that are clusters in G' or that are contained in a cluster together with one large cluster. Note that the corresponding subproblem exactly corresponds to Fixed Clique Cluster Editing since each clique Q_i is in a cluster with at most one large cluster B_j and vice versa. The remaining clusters in $G - Y$ all have size at most $3c$. Hence, by Lemma 4 the remaining instance can be solved in fpt-time.

For the running time note that there are $O(c^c)$ partitions of Y. For each such partition we try all two-partitions. Hence, Alg. 1 enters the body of the inner for loop $O(2^{c\log(c)+c})$ times. Since by Lemma 3 the subroutine `SolveLarge` can be applied in polynomial time and since subroutine `SolveSmall` runs in fpt-time (by Lemma 4), Alg. 1 runs in fpt-time. A naive estimation gives a bound of $2^{2^{O(c^2)}}$ for the combinatorial explosion in the running time. □

2.2 Cluster Deletion

For Cluster Deletion parameterized by the cluster vertex deletion number, we can achieve an algorithm with a better running time than the algorithm for Cluster Editing (that in the subprocedure `SolveSmall` basically resorts to brute-force). The main feature of Cluster Deletion that helps in achieving this better algorithm is that none of the clusters in $G - Y$, where Y is the cluster vertex deletion set, can be merged since only edge deletions are allowed.

Theorem 2. Cluster Deletion *can be solved in* $c^{7c}|V| \cdot poly(n)$ *time, where* c *is the cluster vertex deletion number of* G.

3 Maximum Degree

We show that Cluster Editing is NP-hard even when restricted to graphs with maximum degree six. To the best of our knowledge all previous NP-hardness proofs require an unbounded degree. As an immediate consequence, Cluster Editing is NP-hard even if each vertex is only incident to a constant number of modified edges.

For the NP-hardness proof we present a reduction from 3-SAT. The basic idea of the construction is as follows. For each variable x_i of a given 3-CNF formula we construct a so-called variable cycle of length $4m$. It is easy to verify that only deleting every second edge gives an optimal solution for an even-length cycle. Thus, the two corresponding possibilities are used to represent the two choices for the value of x_i. Moreover, for each clause C_j containing the variables x_p, x_q, and x_r, we connect the three corresponding variable cycles by a clause gadget. In doing so, the goal is to ensure that if the solutions for the variable gadgets correspond to an assignment that satisfies C_j, then all P_3s of the clause gadget can be destroyed by four edge modifications and otherwise by at least five edge modifications. This guarantees that there is a satisfying assignment for the 3-CNF formula if and only if the constructed graph can be transformed into a cluster graph by exactly $2mn + 4m$ edge modifications. The details follow.

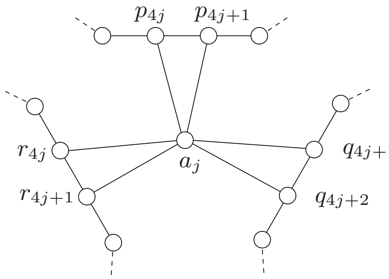

Fig. 1. Illustration of the clause gadget for a clause $C_j = (x_p \vee \overline{x_q} \vee x_r)$.

Given a 3-CNF formula ϕ consisting of the clauses $C_0, \ldots, C_{m-1}$ over the variables $\{x_0, \ldots, x_{n-1}\}$, construct a CE-instance consisting of a graph $G = (V, E)$ and an integer k as follows.

For each variable x_i, $0 \leq i < n$, G contains a *variable cycle* consisting of the vertices $V_i^v := \{i_0, \ldots, i_{4m-1}\}$ and the edges $E_i^v := \{\{i_k, i_{k+1}\} \mid 0 \leq k < m\}$ (for ease of presentation let $i_{4m} = i_0$). So far, the constructed graph consists of a disjoint union of cycles, each of length $4m$. We use the following notation. The edges $\{i_0, i_1\}, \{i_2, i_3\}, \ldots, \{i_{4m-2}, i_{4m-1}\}$ of the variable cycle of x_i are called *even* and all others are called *odd*.

Moreover, for each clause C_j containing the variables x_p, x_q, and x_r (either negated or non-negated), we construct a clause gadget connecting the variable gadgets of x_p, x_q, and x_r. More specifically, let a_j be a new vertex and let E_j^c contain for each $i \in \{p, q, r\}$ the edges $\{a_j, i_{4j}\}$ and $\{a_j, i_{4j+1}\}$ if x_i occurs non-negated in C_j or the edges $\{a_j, i_{4j+1}\}$ and $\{a_j, i_{4j+2}\}$, otherwise. See Fig. 3 for an illustration. Finally, let $V := \bigcup_{i=0}^{n-1} V_i^v \cup \bigcup_{j=0}^{m-1} \{a_j\}$ and $E := \bigcup_{i=0}^{n-1} E_i^v \cup \bigcup_{j=0}^{m-1} E_j^c$. This completes the construction of $G = (V, E)$.

Theorem 3. Cluster Editing *and* Cluster Deletion *are NP-complete even when restricted to graphs with maximum vertex degree six.*

4 Number of Clusters and Maximum Number of Edge Modifications per Vertex

We show that a constrained version of Cluster Editing is fixed-parameter tractable with respect to the combined parameter "number d of clusters in the target graph" and "maximum number t of modifications per vertex". The problem under consideration is a generalization of Cluster Editing:

(d, t)-Constrained-Cluster Editing ((d, t)-CCE)
Input: An undirected graph $G = (V, E)$, a function $\tau : V \to \{0, \ldots, t\}$, and nonnegative integers d and k.
Question: Can G be transformed into a cluster graph G' by applying at most k edge modifications such that G' has at most d clusters and each vertex $v \in V$ is incident to at most $\tau(v)$ modified edges?

We use τ during our algorithm to keep track of the number of modifications that each vertex has been incident to. We can initially set $\tau(v) := t$ for each $v \in V$ and model directly the constraints described above. If only edge deletions are allowed, the corresponding problem is called (d,t)-Constrained-Cluster Deletion. Clearly, CE is exactly (n,n)-CCE. We investigate the parameterized complexity of (d,t)-CCE with respect to the combined parameter (d,t). Before presenting an algorithm for this problem, we discuss several aspects of both the problem formulation and parameterization.

Concerning the problem formulation, in many application scenarios a reasonable upper bound for the number of clusters d is given in advance. Furthermore, constraining the maximum number t of modifications per vertex yields another measure of closeness of the cluster graph to the input graph. In comparison to Cluster Editing, (d,t)-Constrained-Cluster Editing allows to further constrain the solution by adjusting the values of d and t. In certain application scenarios this may help to obtain more reasonable clusterings.

Concerning the parameterization, we consider the combined parameter (d,t) since Cluster Editing is NP-hard for $t = 6$ (which follows from Theorem 3). Moreover, when comparing the parameterizations k and (d,t) one can observe that for some instances, k is not bounded by a function in d and t. Consider for example a graph $G = (V,E)$ that consists of two cliques K_1 and K_2, each of order $|V|/2$. Furthermore, let each $v \in K_1$ have exactly one neighbor in K_2 and vice versa. An optimal solution for this graph is to delete all $|V|/2$ edges between K_1 and K_2. Hence, the parameter k is very large for this instance, whereas $d = 2$ and $t = 1$. In general, we can always assume $t \leq k$. The general relation between d and k is a bit more tricky. For example, in case G is connected, we can assume $d \leq k+1$ since we can create at most $k+1$ connected components by applying k edge modifications to G. Furthermore, in case G does not contain isolated cliques, we can assume $d \leq 2k$, since to each clique in the final cluster graph at least one edge modification is incident. In summary, the parameters d and t can be arbitrarily small compared to k and are bounded from above by a linear function of k when G does not contain isolated cliques.

We now show that (d,t)-Constrained-Cluster Editing is fixed-parameter tractable with respect to (d,t). More precisely, we present four data reduction rules for (d,t)-Constrained-Cluster Editing that produce a kernel consisting of at most $4dt$ vertices. The first two rules identify edge modifications that have to be performed by every solution, since otherwise there would be vertices to which more than t edge modifications are incident.

Reduction Rule 1 *If G contains two adjacent vertices $u, v \in V$ such that $|N(u) \setminus N[v]| > 2t$, then remove $\{u,v\}$ from E and set $\tau(v) := \tau(v) - 1$, $\tau(u) := \tau(u) - 1$, and $k := k - 1$.*

Reduction Rule 2 *If G contains two non-adjacent vertices $u, v \in V$ such that $|N(u) \cap N(v)| > 2t$, then add $\{u,v\}$ to E and set $\tau(v) := \tau(v) - 1$, $\tau(u) := \tau(u) - 1$, $k := k - 1$.*

Lemma 5. *Rules 1 and 2 are correct and can be exhaustively performed in $O(n^3)$ time.*

Proof. Let $(G = (V, E), d, t, k)$ be an input instance of (d, t)-Constrained-Cluster Editing. We show the correctness of each rule and then bound the running time of exhaustively applying both rules.

Let u and v be as described in Rule 1. We show that every solution deletes the edge $\{u, v\}$. Suppose that there is some solution S that does not delete $\{u, v\}$, let $G' := G \Delta S$ be the resulting cluster graph, and let K be the cluster of G' such that $u, v \in K$. Clearly, $|K \cap N(u) \setminus N[v]| \leq t$ since at most t inserted edges are incident to v. Then, however, more than t deleted edges are incident to u. This contradicts that S is a solution.

Let u and v be as described in Rule 2. We show that every solution adds the edge $\{u, v\}$. Suppose that there is some solution S that does not add $\{u, v\}$, let $G' := G \Delta S$ be the resulting cluster graph, and let K be the cluster of G' such that $u \in K$ and $v \notin K$. Since at most t deleted edges are incident to u, we have $|N(u) \cap N(v) \cap K| > t$. Then, however more than t deleted edges are incident to v. This contradicts that S is a solution.

To achieve a running time of $O(n^3)$ we proceed as follows. First, we initialize for each pair of vertices $u, v \in V$ three counters, one counter that counts $|N(u) \cap N(v)|$, one counter that counts $|N(u) \setminus N[v]|$, and one counter that counts $|N(v) \setminus N[u]|$. For each such pair, this is doable in $O(n)$ time when an adjacency matrix has been constructed in advance. Hence, the overall time for initializing the counters for all possible vertex pairs is $O(n^3)$. All counters that warrant an application of either Rule 1 or Rule 2 are stored in a list. We call these counters *active*. Next, we apply the reduction rules. Overall, since $k \leq n^2$ the rules can be applied at most n^2 times. As long as the list of active counters is non-empty, we perform the appropriate rule for the first active counter of the list. It remains to update all counters according to the edge modification applied by the rule. Suppose Rule 2 applies to u and v, that is $\{u, v\}$ is added. Then, we have to update the counters for each pair containing v or u. For v, this can be done in $O(n)$ time, by checking for each $w \neq v$, whether u must be added to $N(v) \cap N(w)$ or added to $N(v) \setminus N[w]$ or removed from $N(w) \setminus N[v]$ (for each counter this can be done in $O(1)$ time by using the constructed adjacency matrix). For each updated counter, we also check in $O(1)$ time whether it needs to be added to/removed from the list of active counters. The case that Rule 1 applies to u and v can be shown analogously. Overall, we need $O(n^3)$ time initialize the counters and $O(n^3)$ time for the exhaustive application of the rules. □

The following reduction rule simply checks whether the instance contains vertices to which already more than t modifications have been applied. Clearly, in this case the instance is a “no”-instance.

Reduction Rule 3 *If there is a vertex $v \in V$ with $\tau(v) < 0$, then output “no”.*

The final rule identifies isolated cliques whose removal does not destroy solutions of (d, t)-Constrained-Cluster Editing.

Reduction Rule 4 *If there is an isolated clique K in G such that $|K| > 2t$, then remove K from G and set $d := d - 1$.*

Lemma 6. *Rule 4 is correct and can be exhaustively performed in $O(m)$ time.*

We now show that the reduction rules above yield a problem kernel.

Theorem 4. (d, t)-Constrained-Cluster Editing *admits a $4dt$-vertex problem kernel which can be found in $O(n^3)$ time.*

Proof. We first show the kernel size and then bound the running time of the kernelization.

Let $(G = (V, E), d, t, k)$ be an input instance of (d, t)-Constrained-Cluster Editing and let G be reduced with respect to Rules 1–4. We show the following:

(G, d, t, k) is a yes-instance $\Rightarrow$ G has at most $4dt$ vertices.

Let S be a solution of the input instance and let G' be the cluster graph that results from applying S to G. We show that every cluster K_i of G' has size at most $4t$. Assume towards a contradiction that there is some K_i in G' with $|K_i| > 4t$. Since G is reduced with respect to Rule 4, there must be either an edge $\{u, v\}$ in G such that $u \in K_i$ and $v \in V \setminus K_i$ or a pair of vertices $u, v \in K_i$ such that $\{u, v\}$ is not an edge in G.

Case 1: $u \in K_i$, $v \in V \setminus K_i$ and $\{u, v\} \in E$. Since at most $t - 1$ edge additions are incident to u, it has in G at least $3t + 1$ neighbors in K_i. Furthermore, since at most t edge deletions are incident to v, it has in G at most t neighbors in K_i. Hence, there are at least $2t + 1$ vertices in K_i that are neighbors of u but not neighbors of v. Therefore, Rule 1 applies in G, a contradiction to the fact that G is reduced with respect to this rule.

Case 2: $u, v \in K_i$ and $\{u, v\} \notin E$. Both u and v are in G adjacent to at least $|K_i| - (t - 1)$ vertices of $K_i \setminus \{u, v\}$. Since $|K_i| > 4t$ they thus have in G at least $2t + 1$ common neighbors. Therefore, Rule 2 applies in G, a contradiction to the fact that G is reduced with respect to this rule.

We have shown that $|K_i| \leq 4t$ for each cluster K_i of G'. Since G' has at most d clusters, the overall bound on the number of vertices follows.

It remains to bound the running time of exhaustively applying Rules 1–4. By Lemma 5, the exhaustive application of Rules 1 and 2 runs in $O(n^3)$ time. After these two rules have been exhaustively applied, Rules3 and 4 can be exhaustively applied in $O(m)$ time. □

Corollary 1. (d, t)-Constrained-Cluster Editing *is fixed-parameter tractable with respect to the parameter (d, t).*

The data reduction rules can be adapted to the case that only edge deletions are allowed. Indeed, we can show a $2dt$-vertex kernel for (d, t)-Constrained-Cluster Deletion by just replacing $2t$ by t in Rules 1 and 4 (no further data reduction rule is needed). Then, with a slightly modified analysis, we arrive at the following (details omitted).

Theorem 5. (d,t)-Constrained-Cluster Deletion *admits a* $2dt$*-vertex kernel which can be found in* $O(n^3)$ *time. It is thus fixed-parameter tractable with respect to the parameter* (d,t).

References

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

[2] S. Böcker, S. Briesemeister, Q. B. A. Bui, and A. Truß. Going weighted: Parameterized algorithms for cluster editing. *Theor. Comput. Sci.*, 410(52):5467–5480, 2009.

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

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

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

[6] F. K. H. A. 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.

[7] E. D. Demaine, M. Hajiaghayi, and D. Marx. Open problems – parameterized complexity and approximation algorithms. In *Parameterized Complexity and Approximation Algorithms*, volume 09511 of *Dagstuhl Seminar Proceedings*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2010.

[8] M. R. Fellows, J. Guo, C. Komusiewicz, R. Niedermeier, and J. Uhlmann. Graph-based data clustering with overlaps. In *Proc. 15th COCOON*, volume 5609 of *LNCS*, pages 516–526. Springer, 2009.

[9] M. R. Fellows, M. A. Langston, F. A. Rosamond, and P. Shaw. Efficient parameterized preprocessing for Cluster Editing. In *Proc. 16th FCT*, volume 4639 of *LNCS*, pages 312–321. Springer, 2007.

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

[11] J. Guo. A more effective linear kernelization for cluster editing. *Theor. Comput. Sci.*, 410(8-10):718–726, 2009.

[12] J. Guo, C. Komusiewicz, R. Niedermeier, and J. Uhlmann. A more relaxed model for graph-based data clustering: s-plex cluster editing. *SIAM Journal on Discrete Mathematics*, 2010. To appear.

[13] F. Hüffner, C. Komusiewicz, H. Moser, and R. Niedermeier. Fixed-parameter algorithms for cluster vertex deletion. *Theory Comput. Syst.*, 47(1):196–217, 2010.

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

[15] R. Niedermeier. Reflections on multivariate algorithmics and problem parameterization. In *Proc. 27th STACS*, volume 5 of *LIPIcs*, pages 17–32. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2010.

[16] F. Protti, M. D. da Silva, and J. L. Szwarcfiter. Applying modular decomposition to parameterized cluster editing problems. *Theory Comput. Syst.*, 44(1):91–104, 2009.

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