Theoretical Computer Science 410 (2009) 718–726

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

A more effective linear kernelization for cluster editing ☆

Jiong Guo *

Institut für Informatik, Friedrich-Schiller-Universität Jena, Ernst-Abbe-Platz 2, D-07743 Jena, Germany

ARTICLE INFO

Article history:
Received 7 August 2007
Received in revised form 15 September 2008
Accepted 9 October 2008
Communicated by X. Deng

Keywords:
Combinatorial problems
Fixed-parameter tractability
Computational biology
Preprocessing
Data reduction

ABSTRACT

In the NP-hard Cluster Editing problem, we have as input an undirected graph G and an integer $k \geq 0$. The question is whether we can transform G, by inserting and deleting at most k edges, into a cluster graph, that is, a union of disjoint cliques. We first confirm a conjecture by Michael Fellows [IWPEC 2006] that there is a polynomial-time kernelization for Cluster Editing that leads to a problem kernel with at most $6k$ vertices. More precisely, we present a cubic-time algorithm that, given a graph G and an integer $k \geq 0$, finds a graph G' and an integer $k' \leq k$ such that G can be transformed into a cluster graph by at most k edge modifications iff G' can be transformed into a cluster graph by at most k' edge modifications, and the problem kernel G' has at most $6k$ vertices. So far, only a problem kernel of $24k$ vertices was known. Second, we show that this bound for the number of vertices of G' can be further improved to $4k$ vertices. Finally, we consider the variant of Cluster Editing where the number of cliques that the cluster graph can contain is stipulated to be a constant $d > 0$. We present a simple kernelization for this variant leaving a problem kernel of at most $(d+2)k+d$ vertices.

1. Introduction

Problem kernelization has been recognized as one of the most important contributions of parameterized algorithmics to practical computing [15,21,25]. A *kernelization* is a polynomial-time algorithm that transforms a given instance I with parameter k of a problem P into a new instance I' with parameter $k' \leq k$ of P such that the original instance I is a yes-instance with parameter k iff the new instance I' is a yes-instance with parameter k' and $|I'| \leq g(k)$ for a function g. The instance I' is called the *problem kernel*. For instance, the derivation of a problem kernel of linear size, that is, function g is a linear function, for the Dominating Set problem on planar graphs [2] is one of the breakthroughs in the kernelization area. The problem kernel derived there consists of at most $335k$ vertices, where k denotes the domination number of the given graph, and this was subsequently improved by further refined analysis and some additional reduction rules to a size bound of $67k$ [9]. In this work, we are going to improve a size bound of $24k$ vertices for a problem kernel for Cluster Editing [15] to a size bound of $4k$. Moreover, we present improvements concerning the time complexity of the kernelization algorithm.

The edge modification problem Cluster Editing is defined as follows:

> **Input:** An undirected graph $G = (V, E)$ and an integer $k \geq 0$.
> **Question:** Can we transform G, by deleting and adding at most k edges, into a graph that consists of a disjoint union of cliques?

☆ A preliminary version of this paper appears in the Proceedings of the 1st International Symposium on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies (ESCAPE'07), Springer, LNCS 4614, pages 36–47, held in Hangzhou, China, April 7–9 2007.

* Tel.: +49 3641946326; fax: +49 3641946002.
E-mail address: guo@minet.uni-jena.de.

0304-3975/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2008.10.021

We call a graph consisting of disjoint cliques a *cluster graph*. In other words, Cluster Editing asks for a set E' of unordered vertex pairs such that $|E'| \leq k$ and the graph $(V, (E \setminus E') \cup (E' \setminus E))$ is a cluster graph.

The study of Cluster Editing can be dated back to the 1980's. Křivánek and Morávek [23] showed that the so-called Hierarchical-Tree Clustering problem is NP-complete if the clustering tree has a height of at least 3. Cluster Editing can be easily reformulated as a Hierarchical-Tree Clustering problem where the clustering tree has height exactly 3. After that, motivated by some computational biology questions, Ben-Dor et al. [4] rediscovered this problem. Later, Shamir et al. [27] showed the NP-completeness of Cluster Editing. Bansal et al. [3] also introduced this problem as an important special case of the Correlation Clustering problem which is motivated by applications in machine learning and they also showed the NP-completeness of Cluster Editing. It is also worth mentioning the work of Chen et al. [10] in the context of phylogenetic trees; among other things, they also derived that Cluster Editing is NP-complete.

Concerning the polynomial-time approximability of the optimization version of Cluster Editing, Charikar et al. [8] proved that there exists some constant $\epsilon > 0$ such that it is NP-hard to approximate Cluster Editing within a factor of $1+\epsilon$. Moreover, they also provided a polynomial-time factor-4 approximation algorithm for this problem. A randomized expected factor-3 approximation algorithm has been given by Ailon et al. [1]. The first non-trivial fixed-parameter tractability results were given by Gramm et al. [19]. They presented a kernelization for this problem which runs in $O(n^3)$ time on an n-vertex graph and results in a problem kernel with $O(k^2)$ vertices. Moreover, they also gave an $O(2.27^k + n^3)$-time algorithm [19] for Cluster Editing. A practical implementation and an experimental evaluation of the algorithm given in [19] have been presented by Dehne et al. [11]. Very recently, the kernelization result of Gramm et al. has been improved by two research groups: Protti et al. [26] presented a kernelization running in $O(n+m)$ time on an n-vertex and m-edge graph that leaves also an $O(k^2)$-vertex graph. In his invited talk at IWPEC'06, Fellows [15] presented a polynomial-time kernelization algorithm for this problem which achieves a kernel with at most $24k$ vertices. This kernelization algorithm needs to solve an LP-formulation of Cluster Editing. Fellows conjectured that a $6k$-vertex problem kernel should exist.

In this paper, we also study the variant of Cluster Editing, denoted as Cluster Editing[d], where one seeks for a set of at most k edge modifications that transform a given graph into a disjoint union of exactly d cliques for a constant d. For each $d \geq 2$, Shamir et al. [27] showed that Cluster Editing[d] is NP-complete. A simple factor-3 approximation algorithm has been provided by Bansal et al. [3]. As their main technical contribution, Giotis and Guruswami [18] proved that there exists a PTAS for Cluster Editing[d] for every fixed $d \geq 2$. More precisely, they showed that Cluster Editing[d] can be approximated within a factor of $1 + \epsilon$ for arbitrary $\epsilon > 0$ in $n^{O(9^d/\epsilon^2)} \cdot \log n$ time. To our best knowledge, the parameterized complexity of Cluster Editing[d] was unexplored so far.

Here, we confirm Fellows' conjecture by presenting an $O(n^3)$-time combinatorial algorithm which achieves a $6k$-vertex problem kernel for Cluster Editing. Independently, Fellows et al. [16] also achieved a $6k$-vertex kernelization for this problem. Our algorithm is inspired by the "crown reduction rule" used in [15,16]. However, by way of contrast, we introduce the *critical clique* concept into the study of Cluster Editing. This concept played a key role in the fixed-parameter algorithms solving the so-called Closest Leaf Power problem [12,13] and it goes back to the work of Lin et al. [24]. It also turns out that with this concept the correctness proof of the algorithm becomes significantly simpler than in [15,16]. Moreover, we present a new $O(nm^2)$-time kernelization algorithm which achieves a problem kernel with at most $4k$ vertices. Finally, based on the critical clique concept, we show that Cluster Editing[d] admits a problem kernel with at most $(d+2) \cdot k + d$ vertices. The corresponding kernelization algorithm runs in $O(m + n)$ time.

After the publication of the conference version of this paper [20], the practical relevance of our kernelization results has been evaluated with real-world data sets, conduced by a bioinformatic research group [5–7]: It turns out that, combined with some heuristic data reduction rules and cutting plane technique for solving integer linear programs, our kernelization algorithms allow for the first time to exactly solve complex and very large instances of Cluster Editing such as those arising in biological applications.

2. Preliminaries

In this work, we consider only undirected graphs without self-loops and multiple edges. The open (closed) neighborhood of a vertex v in graph $G = (V, E)$ is denoted by $N_G(v)$ ($N_G[v]$), while with $N_G^2(v)$ we denote the set of vertices in G which have a distance of exactly 2 to v. For a vertex subset $V' \subseteq V$, we use $G[V']$ to denote the subgraph of G induced by V', that is, $G[V'] = (V', \{e = \{u, v\} \mid (e \in E) \wedge (u \in V') \wedge (v \in V')\})$. We use $\triangle$ to denote the symmetric difference between two sets, that is, $A \triangle B = (A \setminus B) \cup (B \setminus A)$. A *clique* in graph G is a set C of vertices that induce a complete subgraph $G[C]$. Throughout this paper, let $n := |V|$ and $m := |E|$.

In the following, we introduce the concepts of *critical clique* and *critical clique graph* which have been used in dealing with leaf powers of graphs [24,13,12].

Definition 1. A *critical clique* of a graph G is a clique K where the vertices of K all have the same sets of neighbors in $V \setminus K$, and K is maximal under this property.

Definition 2. Given a graph $G = (V, E)$, let $\mathcal{K}$ be the collection of its critical cliques. Then the *critical clique graph* $\mathcal{C}$ is a graph $(\mathcal{K}, E_{\mathcal{C}})$ with

$$\{K_i, K_j\} \in E_{\mathcal{C}} \iff \forall u \in K_i, v \in K_j : \{u, v\} \in E.$$

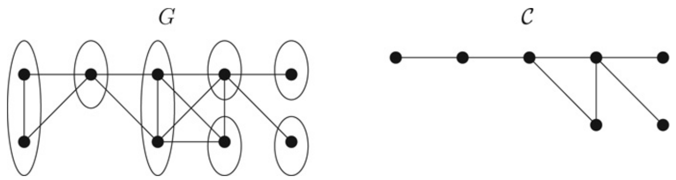

Fig. 1. A graph G and its critical clique graph $\mathcal{C}$. *Ovals* denote the critical cliques of G.

That is, the critical clique graph has the critical cliques as nodes, and two nodes are connected iff the corresponding critical cliques together form a larger clique.

See Fig. 1 for an example of a graph G and its critical clique graph. Note that we use the term *nodes* for the vertices in $\mathcal{C}$. Moreover, we use $K(v)$ to denote the critical clique containing vertex v and use $V(K)$ to denote the set of vertices contained in the critical clique $K \in \mathcal{K}$.

Parameterized complexity is a two-dimensional framework for studying the computational complexity of problems [14,17,25]. One dimension is the input size n (as in classical complexity theory), and the other one is the *parameter* k (usually a positive integer). A problem is called *fixed-parameter tractable* (FPT) if it can be solved in $f(k) \cdot n^{O(1)}$ time, where f is a computable function only depending on k. This means that when solving a combinatorial problem that is FPT, the combinatorial explosion can be confined to the parameter.

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

3. Data reduction leading to a 6*k*-vertex kernel

Based on the concept of critical cliques, we present a polynomial-time kernelization algorithm for Cluster Editing which leads to a problem kernel consisting of at most $6k$ vertices. In this way, we confirm the conjecture by Fellows that Cluster Editing admits a $6k$-vertex problem kernel [15,16]. Our data reduction rules are inspired by the "crown reduction rule" introduced in [15,16]. The main innovation from our side is the novel use of the critical clique concept.

The basic idea behind introducing critical cliques is the following: suppose that the input graph $G = (V, E)$ has a solution with at most k edge modifications. Then, at most $2k$ vertices are "affected" by these edge modifications, that is, they are endpoints of edges added or deleted. Thus, in order to give a size bound on V depending only on k, it remains to upper-bound the size of the "unaffected" vertices. The central observation is that, in the cluster graph obtained after making the at most k edge modifications, the unaffected vertices contained in one clique must form a critical clique in the original graph G. By this observation, it seems easier to derive data reduction rules working for the critical cliques and the critical clique graph than to derive rules directly working on the input graph.

The following two lemmas show the connection between critical cliques and optimal solution sets for the optimization version of Cluster Editing, where, given a graph $G = (V, E)$, one asks for a set E' of unordered vertex pairs such that the graph $(V, E \triangle E')$ is a cluster graph and E' is of minimal size. Such a set E' is called an optimal solution set.

Lemma 1. *There is no optimal solution set E_{opt} for the optimization version of* Cluster Editing *on G that "splits" a critical clique of G. That is, every critical clique is entirely contained in one clique in $G_{\mathrm{opt}} = (V, E \triangle E_{\mathrm{opt}})$ for every optimal solution set E_{opt}.*

Proof. We show this lemma by contradiction. Suppose that we have an optimal solution set E_{opt} for G that splits a critical clique K of G, that is, there are at least two cliques C_1 and C_2 in G_{opt} with $K_1 := C_1 \cap K \neq \emptyset$ and $K_2 := C_2 \cap K \neq \emptyset$. Furthermore, we partition $C_1 \setminus K_1$ (and $C_2 \setminus K_2$) into two subsets, namely, set C_1^1 (and C_2^1) containing the vertices from $C_1 \setminus K_1$ (and $C_2 \setminus K_2$) which are neighbors of the vertices in K in G and $C_1^2 := (C_1 \setminus K_1) \setminus C_1^1$ (and $C_2^2 := (C_2 \setminus K_2) \setminus C_2^1$). See part (a) in Fig. 2 for an illustration. Clearly, E_{opt} deletes the edges E_{K_1,K_2} between K_1 and K_2. In addition, E_{opt} has to delete the edges between K_1 and C_2^1 and the edges between K_2 and C_1^1, and, moreover, E_{opt} has to insert the edges between K_1 and C_1^2 and the edges between K_2 and C_2^2. In summary, E_{opt} needs at least

$$|E_{K_1,K_2}| + |K_1| \cdot |C_2^1| + |K_2| \cdot |C_1^1| + |K_1| \cdot |C_1^2| + |K_2| \cdot |C_2^2|$$

edge modifications.

In what follows, we construct solution sets that are smaller than E_{opt}, giving a contradiction. Consider the following two cases: $|C_1^1| + |C_2^2| \le |C_1^2| + |C_2^1|$ and $|C_1^1| + |C_2^2| > |C_1^2| + |C_2^1|$. In the first case, we remove K_1 from C_1 and merge it to C_2. Herein, we need the following edge modifications: deleting the edges between $K_1 \cup K_2$ and C_1^1 and inserting the edges between $K_1 \cup K_2$ and C_2^2. Here, we need $|K_1| \cdot |C_1^1| + |K_2| \cdot |C_1^1| + |K_1| \cdot |C_2^2| + |K_2| \cdot |C_2^2|$ edge modifications. See part (b) in Fig. 2 for an illustration. In the second case, we remove K_2 from C_2 and merge it to C_1. Herein, we need the following

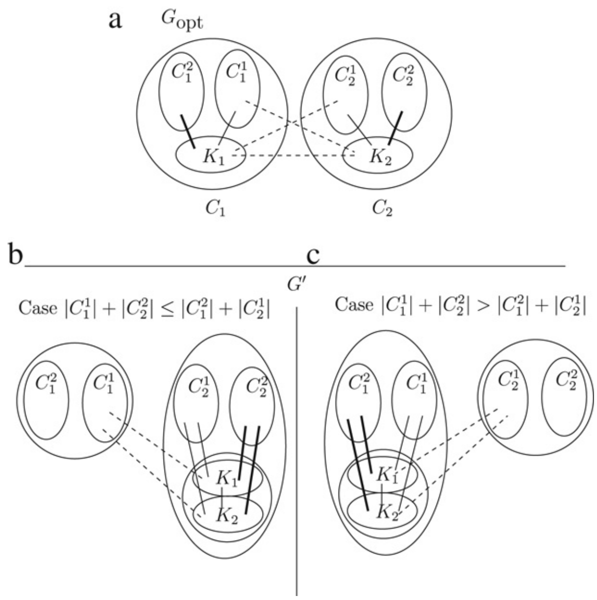

Fig. 2. An illustration of the proof of Lemma 1. The dashed lines indicate edge deletions, the thick lines indicate edge insertions, and the thin lines represent the edges unaffected.

edge modifications: deleting the edges between $K_1 \cup K_2$ and C_2^1 and inserting the edges between $K_1 \cup K_2$ and C_1^2. Here, we need $|K_1| \cdot |C_2^1| + |K_2| \cdot |C_2^1| + |K_1| \cdot |C_1^2| + |K_2| \cdot |C_1^2|$ edge modifications. See part (c) in Fig. 2 for an illustration. Comparing the edge modifications needed in these two cases with $E_{\rm opt}$, we can each time observe that $E_{\rm opt}$ contains some additional edges, namely E_{K_1,K_2}. This means that, in both cases $|C_1^1| + |C_2^2| \le |C_1^2| + |C_2^1|$ and $|C_1^1| + |C_2^2| > |C_1^2| + |C_2^1|$, we can find a solution set that causes less edge modifications than $E_{\rm opt}$, a contradiction to the optimality of $E_{\rm opt}$. □

Lemma 2. *Let K be a critical clique with $|V(K)| \ge |\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')|$. Then, there exists an optimal solution set $E_{\rm opt}$ such that, for the clique C in $G_{\rm opt} = (V, E \triangle E_{\rm opt})$ containing K, it holds $C \subseteq \bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')$.*

Proof. By Lemma 1, the critical clique K is contained entirely in a clique C in $G_{\rm opt} = (V, E \triangle E_{\rm opt})$ for any optimal solution set $E_{\rm opt}$. Suppose that, for an optimal solution set $E_{\rm opt}$, C contains some vertices that are neither from $V(K)$ nor adjacent to a vertex in $V(K)$, that is, $D := C \setminus (\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')) \neq \emptyset$. Then, $E_{\rm opt}$ has inserted at least $|D| \cdot |V(K)|$ many edges into G to obtain the clique C. Then, we can easily construct a new solution set E' which leaves a cluster graph G' having a clique C' with $C' = C \setminus D$. That is, instead of inserting edges between $V(K)$ and D, the solution set E' deletes the edges between $C \cap (\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K'))$ and D. Since $|V(K)| \ge |\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')|$, $E_{\rm opt}$ cannot be better than E'. Hence, E' is an optimal solution set that leads to a cluster graph G' containing a clique $C' \subseteq \bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')$. This completes the proof. □

The following data reduction rules work on both the input graph G and its critical clique graph $\mathcal{C}$. Note that the critical clique graph can be constructed in $O(m + n)$ time [22]: As in the computation of a lexicographic ordering that is built upon partitioning the vertices according to their open neighborhoods, to construct critical clique graphs, one partitions the vertices using their closed neighborhoods. The vertices that cause the same partition form a critical clique.

Rule 1: Remove all isolated critical cliques K from $\mathcal{C}$ and remove $V(K)$ from G.

Lemma 3. *Rule* 1 *is correct and can be carried out in $O(m + n)$ time.*

Proof. This rule is correct, since isolated critical cliques are connected components in the input graph G that are cliques. We need no modification for these components. To find all isolated critical cliques is clearly doable in $O(n)$ time. Therefore, the running time of this rule is dominated by the time for constructing the critical clique graph. □

Rule 2: If, for a node K in $\mathcal{C}$, it holds $|V(K)| > |\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')| + |\bigcup_{K' \in N^2_{\mathcal{C}}(K)} V(K')|$, then remove nodes K and $N_{\mathcal{C}}(K)$ from $\mathcal{C}$ and remove the vertices in $\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')$ from G. Accordingly, decrease parameter k by the sum of the number of edges needed to transform subgraph $G[\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')]$ into a complete graph and the number of edges in G between the vertices in $\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')$ and the vertices in $\bigcup_{K' \in N^2_{\mathcal{C}}(K)} V(K')$. If $k < 0$, then the given instance has no solution.

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

Proof. Let K denote a critical clique in G that satisfies the precondition of Rule 2. Let $A := \{K' \in N_{\mathcal{C}}(K)\}$ and $B := \{K' \in N^2_{\mathcal{C}}(K)\}$. Let $V(A) := \bigcup_{K' \in A} V(K')$ and $V(B) := \bigcup_{K' \in B} V(K')$. From the precondition of Rule 2, we know that $|V(K)| > |V(A)| + |V(B)|$. We show the correctness of Rule 2 by proving the claim that there exists an optimal solution set leaving a cluster graph where there is a clique having exactly the vertex set $V(K) \cup V(A)$.

From Lemmas 1 and 2, we know that there is an optimal solution set E_{opt} such that K is contained entirely in a clique C in $G_{\text{opt}} = (V, E \triangle E_{\text{opt}})$ and clique C contains only vertices from $V(K) \cup V(A)$, that is, $V(K) \subseteq C \subseteq V(K) \cup V(A)$. We show the claim by contradiction. Suppose that $C \subsetneq V(K) \cup V(A)$. By Lemma 1, there is a non-empty subset A_1 of A whose critical cliques are not in C. Let $A_2 := A \setminus A_1$. Moreover, let $E_{A_2,B}$ denote the edges between $V(A_2)$ and $V(B)$ and E_{A_1,A_2} denote the edges between $V(A_1)$ and $V(A_2)$. Clearly, E_{opt} comprises $E_{A_2,B}$ and E_{A_1,A_2}. Moreover, E_{opt} causes the insertion of a set E_{A_2} of edges to transform $G[V(A_2)]$ into a complete graph and causes the deletion of a set E_{K,A_1} of edges between K and A_1. This means that E_{opt} needs at least

$$|E_{A_1,A_2}| + |E_{A_2,B}| + |E_{A_2}| + |E_{K,A_1}| = |E_{A_1,A_2}| + |E_{A_2,B}| + |E_{A_2}| + |V(K)| \cdot |V(A_1)|$$

edge modifications to obtain clique C.

Now, we construct a solution set that is smaller than E_{opt}, giving a contradiction. Consider the solution set E' that leaves a cluster graph G' where K and all critical cliques in A form a clique C' and the vertices in $V \setminus (V(K) \cup V(A))$ are in the same cliques as in G_{opt}. To obtain clique C', the solution set E' contains also the edges in E_{A_2} and the edges in $E_{A_2,B}$. In addition, E' causes the insertion of all possible edges between the vertices in $V(A_1)$, the insertion of all possible edges between $V(A_1)$ and $V(A_2)$, and the deletion of the edges between $V(A_1)$ and $V(B)$. However, these additional edge modifications together amount to at most $|V(A_1)| \cdot (|V(A)| + |V(B)|)$. To create other cliques which do not contain vertices from $V(K) \cup V(A)$, the set E' causes at most as many edge modifications as E_{opt}. From the precondition of Rule 2 that $|V(K)| > |V(A)| + |V(B)|$, we know that even if $E_{A_1,A_2} = \emptyset$, E_{opt} needs more edge modifications than E', which contradicts the optimality of E_{opt}. This completes the proof of the correctness of Rule 2.

The running time of Rule 2 is easy to prove: the construction of $\mathcal{C}$ is doable in $O(m + n)$ time [22]. To decide whether Rule 2 is applicable, we need to iterate over all critical cliques and, for each critical clique K, we need to compute the sizes of $\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')$ and $\bigcup_{K' \in N^2_{\mathcal{C}}(K)} V(K')$. By applying a breadth-first search, these two set sizes for a fixed critical clique can be computed in $O(n)$ time. Thus, we can decide the applicability of Rule 2 in $O(n^2)$ time. Moreover, since every application of Rule 2 removes some vertices from G, it can be applied at most n times. The overall running time follows. $\square$

An instance to which none of the above two reduction rules applies is called *reduced* with respect to these rules. The proof of the following theorem works in analogy to the one of Theorem 3 showing the $24k$-vertex problem kernel in [16].

Theorem 1. *If a reduced graph for* Cluster Editing *has more than* $6k$ *vertices, then it has no solution with at most* k *edge modifications.*

Proof. Suppose that the reduced instance has a solution set of at most k edge modifications. We partition the vertices V of the reduced graph into two sets V_1 and V_2, set V_1 containing the vertices affected by the edge modifications, and set V_2 containing the unaffected vertices. With at most k edge modifications, $|V_1|$ is clearly upper-bounded by $2k$. It remains to upper-bound $|V_2|$.

Consider the cluster graph resulting by the at most k edge modifications. Each of the disjoint cliques of the cluster graph may contain some unaffected vertices. Moreover, the unaffected vertices in each of these cliques form exactly one critical clique in the input graph: let K denote the set of all unaffected vertices in one of these cliques. Since the vertices in K have the same neighborhoods in the resulting cluster graph as in the input graph, they also form a clique in the input graph. By the same reason, their closed neighborhoods in the input graph are identical. Finally, K is maximal, because an optimal solution set will never affect exactly one of two vertices with identical neighborhood, as shown in the proof of Lemma 1. Thus, K is a critical clique.

Consider one of these critical cliques K formed by the unaffected vertices. Since the graph is reduced with respect to Rule 1, the neighborhood of K in $\mathcal{C}$ is not empty, that is, $N_{\mathcal{C}}(K) \neq \emptyset$. Since the vertices in $V(K)$ are unaffected, all vertices in $\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')$ are in the same clique of the cluster graph as the vertices in $V(K)$ and the vertices in $\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')$ must be affected vertices. Moreover, the vertices in $\bigcup_{K' \in N^2_{\mathcal{C}}(K)} V(K')$ cannot be in the same clique as the vertices in $V(K)$ and are affected vertices as well. Thus, $(\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')) \cup (\bigcup_{K' \in N^2_{\mathcal{C}}(K)} V(K')) \subseteq V_1$. Furthermore, since the input graph is reduced with respect to Rule 2, the size of the critical clique K is bounded from above by $|\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')| + |\bigcup_{K' \in N^2_{\mathcal{C}}(K)} V(K')|$.

Now, we add up the sizes of all unaffected critical cliques, giving an upper bound on $|V_2|$. Suppose that there are ℓ cliques in the cluster graph, namely $C_1, C_2, \ldots, C_\ell$. Let K_i denote the unaffected critical clique contained in C_i. We have

$$\begin{aligned} |V_2| &= \sum_{i=1}^{\ell} |K_i| \\ &\leq \sum_{i=1}^{\ell} \left[\left| \bigcup_{K'_i \in N_{\mathcal{C}}(K_i)} V(K'_i) \right| + \left| \bigcup_{K'_i \in N^2_{\mathcal{C}}(K_i)} V(K'_i) \right| \right]. \end{aligned} \tag{1}$$

Note that if two cliques C_r and C_s have at least one edge between them in the reduced graph, some affected vertices in C_r and C_s could be counted more than once in (1); for example, an affected vertex in C_r which in G is adjacent to a vertex of C_s could be counted once in $\bigcup_{K'_r \in N_{\mathcal{C}}(K_r)} V(K'_r)$ and once in $\bigcup_{K'_s \in N^2_{\mathcal{C}}(K_s)} V(K'_s)$. However, if an affected vertex is counted more than twice, then it is incident to at least two modified edges. This means $|V_1| \leq 2k - 1$. In the worst case, we have that every affected vertex is counted exactly twice and $|V_1| = 2k$. Therefore, we have

$$|V_2| = \sum_{i=1}^{\ell} |K_i| \leq 2 \cdot |V_1| = 4k.$$

This gives us the number of vertices in the kernel to be upper-bounded by $2k + 4k = 6k$. $\square$

4. Data reduction leading to a 4*k*-vertex kernel

In this section, we show that the size bound for the number of vertices of the problem kernel for Cluster Editing can be improved from $6k$ to $4k$. In the proof of Theorem 1, the size of the set V_2 of the unaffected vertices is bounded by a function of the size of the set V_1 of the affected vertices. Since $|V_1| \leq 2k$ and each affected vertices could be counted twice, we have then the size bound $4k$ for V_2. In the following, we present two new data reduction rules, Rules 3 and 4, which, combined with Rule 1 in Section 3, enable us to show that $|V_2| \leq 2k$. Note that we achieve this smaller number of kernel vertices with a somewhat increased running time. More specifically, we need an additional factor of $O(m^2/n^2) = O(m)$.

> **Rule 3**: Let K denote a critical clique in the critical clique graph $\mathcal{C}$ with $|V(K)| \geq |\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')|$. If, for a critical clique K' in $N_{\mathcal{C}}(K)$, it holds $E_{K', N^2_{\mathcal{C}}(K)} \neq \emptyset$ and $|V(K)| \cdot |V(K')| \geq |E_{K', N_{\mathcal{C}}(K)}| + |E_{K', N^2_{\mathcal{C}}(K)}|$, where $E_{K', N_{\mathcal{C}}(K)}$ denotes the set of edges needed to connect the vertices in $V(K')$ to the vertices in all other critical cliques in $N_{\mathcal{C}}(K)$ and $E_{K', N^2_{\mathcal{C}}(K)}$ denotes the set of edges between $V(K')$ and the vertices in the critical cliques in $N^2_{\mathcal{C}}(K)$, then we remove all edges in $E_{K', N^2_{\mathcal{C}}(K)}$ and decrease the parameter k accordingly. If $k < 0$, then the given instance has no solution.

Lemma 5. *Rule 3 is correct and can be carried out in $O(nm^2)$ time.*

Proof. Let K be a critical clique with $|V(K)| \geq |\bigcup_{K' \in N_{\mathcal{C}}(K)} |V(K')|$. Suppose that there is a critical clique K' in $N_{\mathcal{C}}(K)$ for which the precondition of Rule 3 holds. By Lemma 1, an optimal solution set splits neither K nor K', that is, every optimal solution set either creates a clique containing both K and K' or separates them. To prove the correctness of Rule 3, we show that separating K and K' is never better than keeping them together. Suppose that we have an optimal solution set E_{opt} such that in $G_{\text{opt}} = (V, E \triangle E_{\text{opt}})$, K and K' are contained in two different cliques C and C', respectively. Now we construct a new cluster graph G' that, compared with G_{opt}, differs only in C and C', namely, we remove K' from C' and add it to C. Let E' denote the set of the necessary edge modifications from the input graph G to G'. It remains to show that E' is optimal. Compared to E_{opt}, E' saves, on one hand, the edge deletions between $V(K)$ and $V(K')$. On the other hand, by Lemma 2, there is an optimal solution set leaving a clique that contains K and is a subset of $\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')$. Hence, we can assume that E_{opt} is such a solution and $C \subseteq \bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')$. Then, to add K' to C, E' has to delete all edges in $E_{K', N^2_{\mathcal{C}}(K)}$. In addition, E' has to insert the edges between $V(K')$ and the vertices in $(C \cap (\bigcup_{K'' \in N_{\mathcal{C}}(K)} V(K''))) \setminus V(K')$. Obviously, these additional edge insertions amount to at most $|E_{K', N_{\mathcal{C}}(K)}|$. By the precondition of Rule 3, that is, $|V(K)| \cdot |V(K')| \geq |E_{K', N_{\mathcal{C}}(K)}| + |E_{K', N^2_{\mathcal{C}}(K)}|$, E' has no more edge modifications than E_{opt}. Thus, E' is also an optimal solution set and separating K and K' is never better than keeping them together. Therefore, we can safely remove the edges in $E_{K', N^2_{\mathcal{C}}(K)}$ and Rule 3 is correct.

Given a critical clique graph $\mathcal{C}$ and a fixed critical clique K, we can compute, for all critical cliques $K' \in N_{\mathcal{C}}(K)$, the sizes of the two edge sets $E_{K', N_{\mathcal{C}}(K)}$ and $E_{K', N^2_{\mathcal{C}}(K)}$ as defined in Rule 3 in $O(m)$ time. To decide whether Rule 3 can be applied, one iterates over all critical cliques K and computes $E_{K', N_{\mathcal{C}}(K)}$ and $E_{K', N^2_{\mathcal{C}}(K)}$ for all critical cliques $K' \in N_{\mathcal{C}}(K)$. Thus, the applicability of Rule 3 can be decided in $O(nm)$ time. Clearly, Rule 3 can be applied at most m times; this gives us an overall running time of $O(nm^2)$. $\square$

Note that the running time of Rule 3 can be improved by applying first the kernelization from Section 3 to the input graph and, then, Rule 3 to the remaining graph which has at most $6k$ vertices and $O(k^2)$ edges. In total, we arrive at the upper bound $O(n^3 + k^5)$ for the running time of Rule 3.

> **Rule 4**: Let K denote a critical clique with $|V(K)| \geq |\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')|$ and $N^2_{\mathcal{C}}(K) = \emptyset$. Then, we remove the critical cliques in $N_{\mathcal{C}}[K]$ from $\mathcal{C}$ and their corresponding vertices from G. We decrease the parameter k by the number of the missing edges between the vertices in $\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')$. If $k < 0$, then the given instance has no solution.

Lemma 6. *Rule 4 is correct and can be carried out in $O(n^3)$ time.*

Proof. If a critical clique K in $\mathcal{C}$ satisfies the precondition of Rule 4, then the critical cliques in $N_{\mathcal{C}}[K]$ and the vertices in $\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')$ form a connected component of $\mathcal{C}$ and G, respectively. We claim that any optimal solution will create a clique consisting of $V(K)$ and all vertices of the critical cliques in $N_{\mathcal{C}}(K)$. Suppose that this is not the case for an optimal solution set E_{opt}, that is, in $G_{\text{opt}} = (V, E \triangle E_{\text{opt}})$, critical clique K is together with a subset $\mathcal{K}_1 \subseteq N_{\mathcal{C}}(K)$ in a clique and the critical cliques in $\mathcal{K}_2 := N_{\mathcal{C}}(K) \setminus \mathcal{K}_1$ are in other cliques. Then, E_{opt} has to delete the edges between $V(K)$ and the vertices in $\bigcup_{K' \in \mathcal{K}_2} V(K')$, that is, we have $|V(K)| \cdot |\bigcup_{K' \in \mathcal{K}_2} V(K')|$ edge deletions. Additionally, E_{opt} causes the insertions of all missing edges $E_{\mathcal{K}_1}$ between the vertices in $\bigcup_{K' \in \mathcal{K}_1} V(K')$. However, putting all vertices in the critical cliques in $N_{\mathcal{C}}[K]$ into a clique requires, in addition to the insertion of the edges in $E_{\mathcal{K}_1}$, the insertions of all possible edges between the vertices in $\bigcup_{K' \in \mathcal{K}_2} V(K')$ and all possible edges between the vertices in $\bigcup_{K' \in \mathcal{K}_1} V(K')$ and the vertices in $\bigcup_{K' \in \mathcal{K}_2} V(K')$. Altogether, these edge modifications amount to at most $|\bigcup_{K' \in \mathcal{K}_2} V(K')| \cdot |\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')|$. The fact that $|V(K)| \geq |\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')|$ implies that putting all vertices in the critical cliques in $N_{\mathcal{C}}[K]$ into a clique needs less edge modifications than E_{opt}, a contradiction to the optimality of E_{opt}. The correctness of Rule 4 follows from this claim and the fact that the edge modifications made by Rule 4 create a clique consisting of all vertices in the critical cliques in $N_{\mathcal{C}}[K]$.

The running time of Rule 4 is obvious. We iterate over all connected components of $\mathcal{C}$ and, in each component, we iterate again over all critical cliques. To decide whether the precondition of Rule 4 holds for a fixed critical clique is clearly doable in $O(n)$ time. Since each application of Rule 4 removes some vertices, it can be applied at most $O(n)$ times. The overall running time then follows. $\square$

Based on these two data reduction rules, we achieve a problem kernel of $4k$ vertices for Cluster Editing.

Theorem 2. *If a graph G that is reduced with respect to Rules* 1, 3, *and* 4 *has more than* $4k$ *vertices, then there is no solution for* Cluster Editing *with at most k edge modifications.*

Proof. Suppose that there is a solution set E_{opt} of the reduced instance with at most k edge modifications that leads to a cluster graph with ℓ cliques, $C_1, C_2, \ldots, C_\ell$. We partition V into two sets, namely set V_1 of the affected vertices and set V_2 of the unaffected vertices. Obviously, $|V_1| \leq 2k$. We know that in each of the ℓ cliques the unaffected vertices must form exactly one critical clique in G. Let $K_1, K_2, \ldots, K_\ell$ denote the critical cliques formed by these unaffected vertices. These critical cliques can be divided into two sets, $\mathcal{K}_1$ containing the critical cliques K for which $|V(K)| < |\bigcup_{K' \in N_{\mathcal{C}}(K)} V(K')|$ holds, and $\mathcal{K}_2 := \{K_1, K_2, \ldots, K_\ell\} \setminus \mathcal{K}_1$.

First, we consider a critical clique K_i from $\mathcal{K}_1$. Since G is reduced with respect to Rule 1, $\bigcup_{K' \in N_{\mathcal{C}}(K_i)} V(K') \neq \emptyset$ and all vertices in $\bigcup_{K' \in N_{\mathcal{C}}(K_i)} V(K')$ must be affected vertices. Clearly, the size of $\bigcup_{K' \in N_{\mathcal{C}}(K_i)} V(K')$ can be bounded from above by $2|E_i^+| + |E_i^-|$, where E_i^+ is the set of the edges inserted by E_{opt} with both their endpoints being in C_i, and E_i^- is the set of the edges deleted by E_{opt} with exactly one of their endpoints being in C_i. Hence, $|V(K_i)| < 2|E_i^+| + |E_i^-|$.

Second, we consider a critical clique K_i from $\mathcal{K}_2$. Since G is reduced with respect to Rules 1 and 4, we know that $N_{\mathcal{C}}(K_i) \neq \emptyset$ and $N_{\mathcal{C}}^2(K_i) \neq \emptyset$. Moreover, since G is reduced with respect to Rule 3, there exists a critical cliques K' in $N_{\mathcal{C}}(K_i)$ for which it holds that $E_{K',N_{\mathcal{C}}^2(K_i)} \neq \emptyset$ and $|V(K_i)| \cdot |V(K')| < |E_{K',N_{\mathcal{C}}(K_i)}| + |E_{K',N_{\mathcal{C}}^2(K_i)}|$, where $E_{K',N_{\mathcal{C}}(K_i)}$ denotes the set of edges needed to connect $V(K')$ to the vertices in the critical cliques in $N_{\mathcal{C}}(K_i) \setminus \{K'\}$ and $E_{K',N_{\mathcal{C}}^2(K_i)}$ denotes the set of edges between $V(K')$ and the vertices in the critical cliques in $N_{\mathcal{C}}^2(K_i)$. Then we have

$$|V(K_i)| < (|E_{K',N_{\mathcal{C}}(K_i)}| + |E_{K',N_{\mathcal{C}}^2(K_i)}|)/|V(K')| \leq |E_i^+| + |E_i^-|$$

where E_i^+ and E_i^- are defined as above.

To give an upper bound of $|V_2|$, we use E^+ to denote the set of edges inserted by E_{opt} and E^- to denote the set of edges deleted by E_{opt}. We have

$$\begin{aligned} |V_2| \; &= \; \sum_{i=1}^{\ell} |V(K_i)| \overset{(*)}{\leq} \sum_{i=1}^{\ell} (2|E_i^+| + |E_i^-|) \overset{(**)}{=} 2|E^+| + \sum_{i=1}^{\ell} |E_i^-| \\ &\overset{(***)}{=} 2|E^+| + 2|E^-| = 2k. \end{aligned}$$

The inequality $(*)$ follows from the analysis in the above two cases. The fact that E_i^+ and E_j^+ are disjoint for $i \neq j$ gives the equality $(**)$. Since an edge between two cliques C_i and C_j that is deleted by E_{opt} has to be counted twice, once for E_i^- and once for E_j^-, we have the equality $(***)$. Together with $|V_1| \leq 2k$, we thus arrive at the claimed size bound. $\square$

5. Cluster editing with a fixed number of cliques

In this section, we consider the Cluster Editing[d] problem. The first observation here is that the data reduction rules from Sections 3 and 4 do not work for Cluster Editing[d]. The reason is that Lemma 1 is not true if the number of cliques is fixed: in order to get a prescribed number of cliques, one critical clique might be split into several cliques by an optimal solution. However, based on the critical clique concept, we can show that Clique Editing[d] admits a problem kernel with at most $(d + 2)k + d$ vertices.

The kernelization is based on a simple data reduction rule.

> **Rule**: If a critical clique K contains at least $k + 2$ vertices, then remove the critical cliques in $N_{\mathcal{C}}[K]$ from the critical clique graph $\mathcal{C}$ and remove the vertices in $\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')$ from the input graph G. Accordingly, decrease d by one. Moreover, decrease the parameter k by the sum of the number of the edges needed to transform the subgraph $G[\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')]$ into a complete graph and the number of edges between the vertices in $\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K')$ and the vertices in $V \setminus (\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K'))$. If $k < 0$, then the given instance has no solution.

Lemma 7. *The above data reduction rule is correct and can be executed in $O(m + n)$ time.*

Proof. If we have a critical clique K with $|K| \geq k + 2$, then we cannot split K by at most k edge modifications allowed, since we need at least $k + 1$ edge deletions to separate one vertex of K from other vertices of K. Moreover, we cannot separate any neighboring critical cliques from K, for which we need at least $k+2$ edge deletions. To put K and some vertices in $V \setminus (\bigcup_{K' \in N_{\mathcal{C}}[K]} V(K'))$ together in a clique needs also at least $k+2$ edge insertions. Thus, any solution set with at most k edge modifications will put the critical cliques in $N_{\mathcal{C}}[K]$ into one clique. This shows the correctness of the above data reduction rule.

To examine the applicability of this data reduction rule, we can simply iterate over all critical cliques and check their sizes. Thus, the running time of this rule follows from the fact that a critical clique graph can be constructed in $O(m + n)$ time [22]. $\square$

Next, we show a problem kernel for Cluster Editing[d].

Theorem 3. *If a graph G that is reduced with respect to the above data reduction rule has more than $(d + 2) \cdot k + d$ vertices, then it has no solution for* Cluster Editing[d] *with at most k edge modifications allowed.*

Proof. As in the proofs of Theorems 1 and 2, we partition the vertices into two sets. The set V_1 of affected vertices has a size bounded from above by $2k$. It remains to upper-bound the size of the set V_2 of unaffected vertices. Since in Cluster Editing[d] the goal graph has exactly d cliques, these unaffected vertices can be partitioned into at most d subsets. In the input graph, each of these sets forms a clique and the vertices in such a set have the same neighborhood. Thus, each of these sets is a subset of a critical clique. Since the graph G is reduced, the maximal size of a critical clique is upper-bounded by $k + 1$. Thus, $|V_2| \leq d \cdot (k + 1)$ and $|V| \leq (d + 2) \cdot k + d$. $\square$

Based on Theorem 3 and the fact that a problem is fixed-parameter tractable iff it admits a problem kernel [14,25], we get the following corollary.

Corollary 1. *For a constant d,* Cluster Editing[d] *is fixed-parameter tractable with the number k of allowed edge modifications as parameter.*

Note that the above data reduction rule and the problem kernel analysis in the proof of Theorem 3 work also for the problem variant where one has the same input as for Cluster Editing[d] and where the goal graph is a disjoint union of *at most d* cliques. Therefore, the kernelization and fixed-parameter tractability results hold also for this variant.

6. Open problems and future research

In this paper, we have presented several polynomial-time kernelization algorithms for Cluster Editing and Cluster Editing[d]. We propose the following directions for future research.

- Can the size bounds of the problem kernels of Cluster Editing and Cluster Editing[d] be further improved?
- Can the running time of the data reduction rules be improved to be linear?
- Can we apply the critical clique concept to derive a problem kernel for the more general Correlation Clustering problem [3]?
- Can the technique from [9] be applied to show a lower bound on the problem kernel size for Cluster Editing?

Acknowledgments

I thank Rolf Niedermeier (Universität Jena) for inspiring discussions and helpful comments improving the presentation.
Author was supported by the Deutsche Forschungsgemeinschaft (DFG), Emmy Noether research group PIAF (fixed-parameter algorithms), NI 369/4.

References

[1] N. Ailon, M. Charikar, A. Newman, Aggregating inconsistent information: Ranking and clustering, in: Proc. 37th ACM STOC, ACM Press, 2005, pp. 684–693.
[2] J. Alber, M.R. Fellows, R. Niedermeier, Polynomial time data reduction for Dominating Set, Journal of the ACM 51 (3) (2004) 363–384.
[3] Nikhil Bansal, Avrim Blum, Shuchi Chawla, Correlation clustering, Machine Learning 56 (1) (2004) 89–113.
[4] A. Ben-Dor, R. Shamir, Z. Yakhini, Clustering gene expression patterns, Journal of Computational Biology 6 (3/4) (1999) 281–297.
[5] S. Böcker, S. Briesemeister, Q.B.A. Bui, A. Truß, A fixed-parameter approach for weighted cluster editing, in: Proc. 6th Asia–Pacific Bioinformatics Conference, APBC 2008, in: Series on Advances in Bioinformatics and Computational Biology, vol. 5, Imperial College Press, 2008, pp. 211–220.
[6] S. Böcker, S. Briesemeister, Q.B.A. Bui, A. Truß, Going weighted: Parameterized algorithms for cluster editing, in: Proc. 2nd COCOA, in: LNCS, vol. 5165, Springer, 2008, pp. 1–12.
[7] S. Böcker, S. Briesemeister, G.W. Klau, Exact algorithms for cluster editing: Evaluation and experiments, in: Proc. 7th WEA, in: LNCS, vol. 5038, Springer, 2008, pp. 289–302.
[8] Moses Charikar, Venkatesan Guruswami, Anthony Wirth, Clustering with qualitative information, Journal of Computer and System Sciences 71 (3) (2005) 360–383.
[9] J. Chen, Henning Fernau, Iyad A. Kanj, Ge Xia, Parametric duality and kernelization: Lower bounds and upper bounds on kernel size, SIAM Journal on Computing 37 (4) (2007) 1077–1106.

[10] Zhi-Zhong Chen, Tao Jiang, Guohui Lin, Computing phylogenetic roots with bounded degrees and errors, SIAM Journal on Computing 32 (4) (2003) 864–879.
[11] F. Dehne, M.A. Langston, X. Luo, S. Pitre, P. Shaw, Y. Zhang, The Cluster Editing problem: Implementations and Experiments, in: Proc. 2nd IWPEC, in: LNCS, vol. 4196, Springer, 2006, pp. 13–24.
[12] M. Dom, J. Guo, F. Hüffner, R. Niedermeier, Extending the tractability border for closest leaf powers, in: Proc. 31st WG, in: LNCS, vol. 3787, Springer, 2005, pp. 397–408. Long version unter the title "Closest 4-leaf power is fixed-parameter tractable" to appear in Discrete Applied Mathematics.
[13] M. Dom, J. Guo, F. Hüffner, R. Niedermeier, Error compensation in leaf power problems, Algorithmica 44 (4) (2006) 363–381.
[14] Rodney G. Downey, Michael R. Fellows, Parameterized Complexity, Springer, 1999.
[15] Michael R. Fellows, The lost continent of polynomial time: Preprocessing and kernelization, in: Proc. 2nd IWPEC, in: LNCS, vol. 4639, Springer, 2006, pp. 312–321.
[16] M.R. Fellows, Michael A. Langston, Frances Rosamond, Peter Shaw, Polynomial-time linear kernelization for Cluster Editing, in: Proc. 16th FCT, in: LNCS, vol. 4169, Springer, 2007, pp. 276–277.
[17] Jörg Flum, Martin Grohe, Parameterized Complexity Theory, Springer, 2006.
[18] I. Giotis, V. Guruswami, Correlation clustering with a fixed number of clusters, in: Proc. 17th ACM-SIAM SODA, ACM Press, 2006, pp. 1167–1176.
[19] J. Gramm, J. Guo, F. Hüffner, R. Niedermeier, Graph-modeled data clustering: Exact algorithms for clique generation, Theory of Computing Systems 38 (4) (2005) 373–392.
[20] J. Guo, A more effective linear kernelization for Cluster Editing, in: Proc. 1st ESCAPE, in: LNCS, vol. 4614, Springer, 2007, pp. 36–47.
[21] J. Guo, R. Niedermeier, Invitation to data reduction and problem kernelization, ACM SIGACT News 38 (1) (2007) 31–45.
[22] W. Hsu, T. Ma, Substitution decomposition on chordal graphs and applications, in: Proc. 2nd International Symposium on Algorithms, in: LNCS, vol. 557, Springer, 1991, pp. 52–60.
[23] Mirko Křivánek, Jaroslav Morávek, NP-hard problems in hierarchical-tree clustering, Acta Informatica 23 (3) (1986) 311–323.
[24] Guohui Lin, Paul E. Kearney, Tao Jiang, Phylogenetic k-root and Steiner k-root, in: Proc. 11th ISAAC, in: LNCS, vol. 1969, Springer, 2000, pp. 539–551.
[25] Rolf Niedermeier, Invitation to Fixed-parameter Algorithms, Oxford University Press, 2006.
[26] F. Protti, M.D. da Silva, J.L. Szwarcfiter, Applying modular decomposition to parameterized bicluster editing, in: Proc. 2nd IWPEC, in: LNCS, vol. 4169, Springer, 2006, pp. 1–12. Long version to appear in Theory of Computing Systems.
[27] R. Shamir, R. Sharan, D. Tsur, Cluster graph modification problems, Discrete Applied Mathematics 144 (2004) 173–182.