Foundations of Software Technology and Theoretical Computer Science (Bangalore) 2008.
Editors: R. Hariharan, M. Mukund, V. Vinay; pp 280-291

A Cubic-Vertex Kernel for Flip Consensus Tree

Christian Komusiewicz* **and Johannes Uhlmann†**
Institut für Informatik,
Friedrich-Schiller-Universität Jena,
Ernst-Abbe-Platz 2,
D-07743 Jena, Germany.
{ckomus,uhlmann}@minet.uni-jena.de

ABSTRACT. Given a bipartite graph $G = (V_c, V_t, E)$ and a non-negative integer k, the NP-complete MINIMUM-FLIP CONSENSUS TREE problem asks whether G can be transformed, using up to k edge insertions and deletions, into a graph that does not contain an induced P_5 with its first vertex in V_t (a so-called M-graph or Σ-graph). This problem plays an important role in computational phylogenetics, V_c standing for the characters and V_t standing for taxa. Chen et al. [IEEE/ACM TCBB 2006] showed that MINIMUM-FLIP CONSENSUS TREE is NP-complete and presented a parameterized algorithm with running time $O(6^k \cdot |V_t| \cdot |V_c|)$. Recently, Böcker et al. [IWPEC '08] presented a refined search tree algorithm with running time $O(4.83^k(|V_t| + |V_c|) + |V_t| \cdot |V_c|)$. We complement these results by polynomial-time executable data reduction rules yielding a problem kernel with $O(k^3)$ vertices.

1 Introduction

The MINIMUM-FLIP CONSENSUS TREE problem arises in computational phylogenetics in the context of supertree construction. Given a binary matrix, the task is to "flip" a minimum number of entries of the matrix in order to obtain a binary matrix that admits what is called a *perfect phylogeny*. These are matrices from which a rooted phylogenetic tree can be inferred [15, 21].

In this work, we employ a graph-theoretic formulation of the problem, which was introduced by Chen et al. [4]: the binary input matrix A is represented by a bipartite graph $G = (V_c, V_t, E)$ where an edge between two vertices $i \in V_c$ and $j \in V_t$ is drawn iff $A_{i,j} = 1$. The matrix then admits a perfect phylogeny iff the graph does not contain an *M-graph* as an induced subgraph. An M-graph is a path of five vertices with the first vertex belonging to V_t. An example of such an M-graph is depicted in Fig. 1. Then, the flipping of a matrix entry $A_{i,j}$ from 0 to 1 corresponds to the insertion of the edge $\{i, j\}$, and from 1 to 0 corresponds to the deletion of the edge $\{i, j\}$. The MINIMUM-FLIP CONSENSUS TREE problem is then defined as follows.

Instance: A bipartite graph $G = (V_c, V_t, E)$ and an integer $k \geq 0$.
Question: Can G be changed by up to k edge modifications into an M-free graph, that is, a graph without an induced M-graph?

*Supported by a PhD fellowship of the Carl-Zeiss-Stiftung.
†Supported by the DFG, research project PABI, NI 369/7.

FSTTCS 2008
IARCS Annual Conference on
Foundations of Software Technology and Theoretical Computer Science
http://drops.dagstuhl.de/opus/volltexte/2008/1760

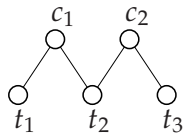

Figure 1: An M-subgraph with $t_1, t_2, t_3 \in V_t$ and $c_1, c_2 \in V_c$.

Chen et al. [4] showed that MINIMUM-FLIP CONSENSUS TREE is NP-complete, which motivates the study of the MINIMUM-FLIP CONSENSUS TREE problem in the context of parameterized algorithmics [19]. Other than previous work [1, 4] on parameterized algorithms for MINIMUM-FLIP CONSENSUS TREE, which mainly dealt with the development of depth-bounded search trees, here we deal with polynomial-time data reduction with provable performance guarantee, that is, kernelization. Kernelization is considered as one of the theoretically and practically most interesting algorithmic methods of parameterized algorithmics [6, 14, 17, 19]. Roughly speaking, the goal is to derive a problem kernel which is an instance "equivalent" to the original one but with (hopefully) much smaller size; in particular, the size of the problem kernel shall only be a function of the parameter k. Moreover, the problem kernel needs to be computable in polynomial-time—so this is closely related to polynomial-time preprocessing.

Known results and previous work. The MINIMUM-FLIP CONSENSUS TREE was introduced by Chen et al. [4] who also proved its NP-completeness and described a factor-$2d$ approximation algorithm for graphs with maximum degree d. Furthermore, they showed fixed-parameter tractability with respect to the number of flips k by describing a simple $O(6^k \cdot mn)$ search tree algorithm that is based on the forbidden induced subgraph characterization with M-graphs. Recently, Böcker et al. [1] improved the running time to $O(4.83^k(|V_c| + |V_t|) + |V_c| \cdot |V_t|)$ by employing a refined branching strategy that leads to a search tree of size $O(4.83^k)$. This theoretically proven running time acceleration was also practically confirmed by computational experiments [1].

From a graph-theoretic point of view, MINIMUM-FLIP CONSENSUS TREE belongs to the class of so-called Π-EDGE MODIFICATION problems: Given a graph G, a graph property Π, and an integer $k \geq 0$, the question is whether G can be transformed by at most k edge modifications into a graph with property Π. A lot of work has been put into classifying Π-EDGE MODIFICATION problems with respect to their classical complexity [3, 18, 24]. Recently, parameterized algorithmics —in particular kernelizations—for Π-EDGE MODIFICATION problems have attracted special attention. For instance, there is a series of papers studying the kernelizability of CLUSTER EDITING and some of its variations [7, 9, 11, 13, 22]. Also vertex deletion problems such as UNDIRECTED FEEDBACK VERTEX SET with its cubic-size problem kernel [2]—very recently improved to a quadratic-vertex problem kernel [23]—have been studied, underpinning the importance of kernelization in the wide area of graph modification problems. Furthermore, even exponential-size kernels such as those for CLIQUE COVER [10] and BICLIQUE COVER [8] are of importance, since they often provide the only known way to show that a problem is fixed-parameter tractable. Damaschke [5] investigated kernelization in the context of enumerating all inclusion-minimal solutions of size

at most k. In this scenario, when designing reduction rules one has to guarantee that all inclusion-minimal solutions of size at most k are preserved. Kernels that fulfill these additional constraints are called *full kernels*. In this setting, Damaschke [5] presents a full kernel consisting of $O(6^k)$ matrix entries for the following problem closely related to MINIMUM-FLIP CONSENSUS TREE: Given a binary matrix and a non-negative integer k, enumerate all inclusion-minimal sets of at most k flips that transform the matrix into a matrix that admits an unrooted perfect phylogeny.

Our contributions. In this work, we provide several polynomial-time data reduction rules for MINIMUM-FLIP CONSENSUS TREE that lead to a problem kernel containing $O(k^3)$ vertices. This is the first non-trivial kernelization result for MINIMUM-FLIP CONSENSUS TREE. Combining our kernelization algorithm with the search tree by Böcker et al. [1], we achieve a running time of $O(4.83^k + \text{poly}(|V_c|, |V_t|))$ instead of the previous $O(4.83^k \cdot \text{poly}(|V_c|, |V_t|))$. Furthermore, we describe one of the data reduction rules in a fairly abstract and general way, making it applicable to a wide range of Π-EDGE MODIFICATION problems. Due to the lack of space, several details are deferred to a full version of the paper.

2 Preliminaries

The open *neighborhood* $N_G(v)$ of a vertex $v \in V$ is the set of vertices that are adjacent to v in $G = (V, E)$. For a set of vertices $V' \subseteq V$, the induced subgraph $G[V']$ is the graph over the vertex set V' with edge set $\{\{v, w\} \in E \mid v, w \in V'\}$. For $V' \subseteq V$ we use $G - V'$ as abbreviation for $G[V \setminus V']$ and for a vertex $v \in V$ let $G - v$ denote $G - \{v\}$. For two sets X and Y with $X \cap Y = \emptyset$, let $E_{X,Y}$ denote the set $\{\{x, y\} \mid x \in X \wedge y \in Y\}$. As an abbreviation for $E_{\{x\},Y}$ we write $E_{x,Y}$. For two sets E and F, define $E \Delta F := (E \setminus F) \cup (F \setminus E)$ (the symmetric difference). Further, for a bipartite graph $G = (V_c, V_t, E)$ and a set $F \subseteq E_{V_c,V_t}$ define $G \Delta F := (V_c, V_t, E \Delta F)$. Sometimes we refer to a vertex $c \in V_c$ as c-vertex, and to a vertex $t \in V_t$ as t-vertex. A graph property Π is called *hereditary* if it holds for all induced subgraphs of a graph G with Π. That is, the class of graphs with a hereditary graph property Π is closed under vertex deletion. Clearly, all graph properties that can be described by a (possibly non-finite) set of forbidden induced subgraphs (such as M-freeness for example) are hereditary. Two c-vertices c_1 and c_2 are said to be *in conflict* if there exists an induced M-graph containing both of them. It is not hard to see that two vertices $c_1, c_2 \in V_c$ are in conflict iff

$$(N_G(c_1) \setminus N_G(c_2) \neq \emptyset) \wedge (N_G(c_1) \cap N_G(c_2) \neq \emptyset) \wedge (N_G(c_2) \setminus N_G(c_1) \neq \emptyset).$$

For our data reduction we crucially use a structure called *critical independent set*.

DEFINITION 1. *Given an undirected graph $G = (V, E)$, a set $I \subseteq V$ is called a* critical independent set *if for any two vertices $v, w \in I$ it holds that v and w are non-adjacent, $N_G(v) = N_G(w)$, and I is maximal with respect to this property.*

All critical independent sets of a graph can be found in linear time [16]. Given a graph $G = (V, E)$ and the collection $\mathcal{I} = \{I_1, I_2, \ldots, I_q\}$ of its critical independent sets,

where $q \leq n$, the *critical independent set graph* of G is the undirected graph $(\mathcal{I}, \mathcal{E})$ with $\{I_i, I_j\} \in \mathcal{E}$ iff $\forall u \in I_i, v \in I_j : \{u, v\} \in E$.

A bipartite graph $G = (X, Y, E)$ is called a *chain graph* if the neighborhoods of the vertices in X form a chain [24]. That is, there is an ordering of the vertices in X, say $x_1, x_2, \ldots, x_{|X|}$, such that $N_G(x_1) \subseteq N_G(x_2) \subseteq \ldots \subseteq N_G(x_{|X|})$. It is easy to see that the neighborhoods of Y also form a chain if G is a chain graph. Moreover, a bipartite graph is a chain graph iff it is $2K_2$-free [24] (herein, a $2K_2$ is the graph that consists of two independent edges). Since every M-graph contains an induced $2K_2$, the set of chain graphs is contained in the class of M-free graphs. One of our data reduction rules is based on identifying and reducing the size of subgraphs of the input graphs that are chain graphs and additionally have a special neighborhood structure.

Parameterized algorithmics [19] aims at a multivariate complexity analysis of problems. This is done by studying relevant problem parameters and their influence on the computational complexity. The decisive question is whether a given parameterized problem is *fixed-parameter tractable (FPT)* with respect to the parameter k. In other words, here we ask for the existence of a solving algorithm with running time $f(k) \cdot \text{poly}(n)$ for some computable function f. A core tool in the development of parameterized algorithms that has been recognized as one of the most important contribution of parameterized algorithmics to practical computing [6, 14, 17, 19] is polynomial-time preprocessing by *data reduction rules*, often yielding a *problem kernel*. Herein, the goal is, given any problem instance G with parameter k, to transform it in polynomial time into a new instance G' with parameter k' such that the size of G' is bounded from above by some function only depending on k, $k' \leq k$, and (G, k) is a yes-instance iff (G', k') is a yes-instance. We call a data reduction rule *correct* if the new instance after an application of this rule is a yes-instance iff the original instance is a yes-instance. An instance is called *reduced* with respect to some data reduction rule if the data reduction rule has been exhaustively applied.

3 A Universal Rule for Critical Independent Sets

In this section, we describe a polynomial-time data reduction rule for parameterized graph modification problems that applies to a certain kind of hereditary graph property and is a generalization of a rule that was developed for BICLUSTER EDITING [22]. Here, we prove the new result that this reduction rule can be applied to a wide range of Π-EDGE MODIFICATION problems, including MINIMUM-FLIP CONSENSUS TREE.

The basic idea of the data reduction is to show that, for some graph properties, vertices that belong to the same critical independent set are subject to the "same" edge modifications. Therefore, large critical independent sets can be reduced. First, we give a description of these graph properties. Let Π be a hereditary graph property. We call Π *critical independent set preserving* (*cisp*) whenever for all forbidden induced subgraphs F of Π, there are no two vertices $u, v \in V(F)$ that form a critical independent set in F (that is, all critical independent sets of F have size one). Note that M-freeness is a cisp graph property: all vertices in an induced M-graph have different neighborhoods. Therefore, the following lemmas and reduction rule apply directly to MINIMUM-FLIP CONSENSUS TREE. First, we can show that cisp graph properties are closed under a certain vertex-addition operation.

LEMMA 2. *Let $G = (V, E)$ be a graph fulfilling a cisp graph property Π. Let G' be the graph that results by adding to G a new vertex $x \notin V$ and making it adjacent to $N_G(v)$ for an arbitrary vertex $v \in V$. Then, G' also fulfills Π.*

Using Lemma 2, we can show that for graph modification problems for cisp properties there is an optimal solution that treats the vertices of a critical independent set equally.

LEMMA 3. *Let $I \subseteq V$ be a critical independent set in $G = (V, E)$, and let Π be a cisp graph property. Then there exists a minimum-cardinality edge modification set S such that $G' := G\Delta S$ fulfills Π and I is part of a critical independent set in G'.*

With Lemma 3 at hand, the following data reduction rule is not hard to see.

REDUCTION RULE 1. *Let $I \subseteq V$ be a critical independent set. If $|I| > k+1$, then delete $|I| - (k+1)$ arbitrary vertices from I.*

LEMMA 4. *Reduction Rule 1 is correct and can be exhaustively applied in $O(|V| + |E|)$ time.*

This general data reduction rule also applies to the COMPLETION and DELETION version of a Π-EDGE MODIFICATION problem for a cisp graph property Π. Examples for graph modification problems to which this rule can be applied are CHAIN DELETION and COTRIVIALLY PERFECT DELETION.[‡]

4 Specific Data Reduction Rules for Minimum-Flip Consensus Tree

In this section, we present three further polynomial-time data reduction rules that together with Reduction Rule 1 produce an $O(k^3)$-vertex kernel. The first reduction rule is obvious.

REDUCTION RULE 2. *Remove M-free connected components from the input graph.*

The next reduction rule removes c-vertices from G that do not appear in an M-graph.

REDUCTION RULE 3. *Let $G = (V_c, V_t, E)$ be a bipartite graph. If there exists a vertex $c \in V_c$ that is not in conflict with any other vertex in V_c, then remove c.*

LEMMA 5. *Reduction Rule 3 is correct and can be exhaustively applied in $O(|V_c|^2 \cdot |V_t|)$ time.*

PROOF. Let G be the original graph and let $G' := G - c$, where $c \in V_c$ is not in conflict with any other c-vertex. First, we prove the correctness of Reduction Rule 3. To this end, we show the following.

Claim: (G, k) is a yes-instance iff (G', k) is a yes-instance.

"$\Rightarrow$:" Follows directly because M-freeness is a hereditary graph property.

"$\Leftarrow$:" This direction is based on the observation that graph G' can be decomposed into two edge disjoint subgraphs G_1 and G_2 that can be solved independently from each other,

[‡]Definitions and kernelization results for these problems have been obtained by Guo [12].

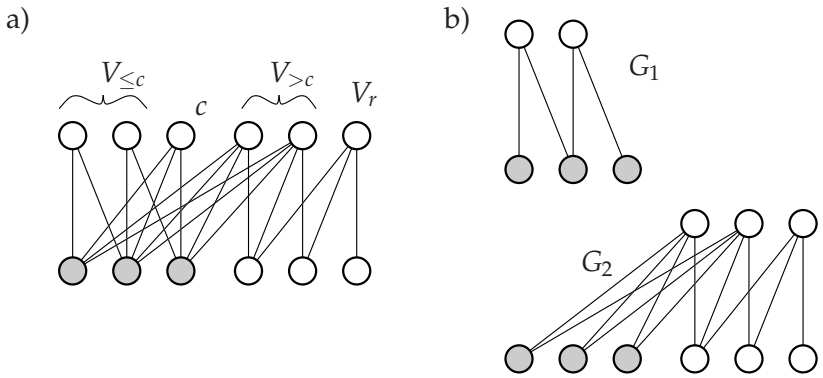

Figure 2: Correctness of Reduction Rule 3. a) Partition of the vertices in V_c depending on their relation to c. The neighbors of c are colored gray. b) The graphs G_1 (induced by $V_{\leq c}$ and $N(c)$) and G_2 (induced by $V_{>c}$,V_r, and V_t) .

without creating a new conflict containing c. We need the following notation.

$$V_{>c} := \{c' \in V_c \mid N(c) \subsetneq N(c')\},$$
$$V_{\leq c} := \{c' \in (V_c \setminus \{c\}) \mid N(c') \subseteq N(c)\}, \text{ and}$$
$$V_r := V_c \setminus (V_{\leq c} \cup V_{>c}).$$

See Fig. 2 a) for an example. Note that, since c is not in conflict with any other vertex $c' \in V_c - c$, either $N_G(c) \cap N_G(c') = \emptyset$ or $c' \in (V_{\leq c} \cup V_{>c})$. In particular, this implies that for every vertex $c' \in V_r$ it holds that $N_G(c) \cap N_G(c') = \emptyset$.

Let F' be a solution for (G', k). We show that from F' we can compute a solution F for (G, k). Let $V_2 := V_r \cup V_{>c}$. Consider the two graphs $G_1 := G[V_{\leq c} \cup N_G(c)]$ and $G_2 := G[V_2 \cup V_t]$. See Fig. 2 b) for an example. Observe that $F_1 := F' \cap E_{V_{\leq c}, N_G(c)}$ is a solution for G_1 and $F_2 := F' \cap E_{V_2, V_t}$ is a solution for G_2, since G_1 and G_2 are induced subgraphs of G'. Furthermore, note that $F_1 \cap F_2 = \emptyset$ since $V_{\leq c} \cap V_2 = \emptyset$. As a consequence, $|F_1| + |F_2| \leq |F'| \leq k$.

Consider the graph G_2. It is easy to observe that $N_G(c)$ is contained in a critical independent set in G_2. This can be seen as follows: since $N_G(c) \subset N_G(c')$ for every vertex $c' \in V_{>c}$ and $N_G(c) \cap N_G(c'') = \emptyset$ for every vertex $c'' \in V_r$, every vertex $t \in N_G(c)$ is adjacent in G_2 to exactly the vertices in $V_{>c}$. Since $N_G(c)$ is a critical independent set in G_2, according to Lemma 3 there exists a minimum-cardinality solution F_2' for G_2 such that $N_G(c)$ is contained in a critical independent set in $G_2 \Delta F_2'$. Clearly, $|F_2'| \leq |F_2|$.

Based on these facts, we show that $F := F_1 \cup F_2'$ is a solution for (G, k). First of all, note that by the discussion above $|F| = |F_1| + |F_2'| \leq |F_1| + |F_2| \leq k$. Second, no two vertices in V_c are in conflict, and hence, $G\Delta F$ is M-free. This can be seen as follows. Since F_1 is a solution for G_1, any two vertices $c_1, c_2 \in V_{\leq c}$ are not in conflict in $G\Delta F$. The same holds true for any two vertices in V_2, since $G_2 \Delta F_2'$ is M-free. Moreover, since for every vertex $c' \in V_{\leq c}$ it holds that $N_{G\Delta F}(c') = N_{G_1 \Delta F_1}(c') \subseteq N_G(c) = N_{G\Delta F}(c)$, c is not in conflict with any vertex in $V_{\leq c}$. Finally, since $N_G(c)$ is a critical independent set in $G_2 \Delta F_2'$, we know that for every $c' \in V_2$ either $N_{G\Delta F}(c') \cap N_G(c) = \emptyset$ or $N_G(c) \subseteq N_{G\Delta F}(c')$ and hence c' is not in conflict with any

vertex $c'' \in V_{\leq c} \cup \{c\}$. Therefore, $G\Delta F$ is M-free.

For the running time consider the following. For each pair of vertices $c_1, c_2 \in V_c$, we can determine in $O(V_t)$ time whether they are in conflict by checking for each vertex $t \in V_t$, whether it is adjacent to c_1, c_2, or both. Each c-vertex that is in conflict with some other vertex is marked. Finally, unmarked vertices are removed from the graph. This can be performed in $O(|E|)$ time. The overall running time is thus $O(|V_c|^2 \cdot |V_t|)$. ∎

The structurally "deepest" reduction rule shrinks subgraphs of the input graph that resemble "local" chain graphs. We call such a subgraph *P-structure*:

DEFINITION 6. *Let $G = (V_c, V_t, E)$ be a bipartite graph. A tuple (C_P, T_P) of two subsets $C_P \subseteq V_c$ and $T_P \subseteq V_t$ forms a* P-structure *if the following three properties are fulfilled:*

1. *$G[C_P \cup T_P]$ is a chain graph,*
2. *for all $c', c'' \in C_P$ it holds that $N(c') \setminus T_P = N(c'') \setminus T_P$, and*
3. *for all $t', t'' \in T_P$ it holds that $N(t') \setminus C_P = N(t'') \setminus C_P$.*

It is easy to see that for a P-structure (C_P, T_P) of a bipartite graph G the neighborhoods in G of the vertices in C_P (and T_P) also form a chain (since "outside" of the P-structure they have the same neighbors). Moreover, note that the vertices of a P-structure form a subgraph that is M-free.

REDUCTION RULE 4. *Let (C_P, T_P) be a P-structure in a bipartite graph $G = (V_c, V_t, E)$. Let $T_P = \{t_1, t_2, \ldots, t_l\}$ such that $N(t_1) \subseteq N(t_2) \subseteq \ldots \subseteq N(t_l)$. If $l > 2(k+1)$, then remove $t_{k+2}, t_{k+3}, \ldots, t_{l-(k+1)}$ from G.*

LEMMA 7. *Reduction Rule 4 is correct and can be exhaustively applied in polynomial time.*

We can find P-structures in polynomial time by trying all possibilities for choosing the four "endpoints" t_1, t_l, c_1, c_q of the chain, where $N(t_1) \subseteq N(t_l)$ and $N(c_q) \subseteq N(c_1)$. It is not hard to see that in the case that t_1, t_l, c_1, c_q are indeed endpoints of a P-structure, we can reconstruct the corresponding P-structure as follows:

$$C_P = (N(t_l) \setminus N(t_1)) \cup \{c_1\} \cup \{c' \in V_v \mid N(c') = N(c_1)\}$$

and analogously

$$T_P = (N(c_1) \setminus N(c_q)) \cup \{t_l\} \cup \{t' \in V_t \mid N(t') = N(t_l)\}.$$

To recognize the cases that t_1, t_l, c_1, c_q are not the endpoints of a chain, we have to check whether the found vertex sets indeed form a P-structure. This approach works clearly in polynomial time, although there seems to be room for improving the efficiency, a task for future research.

5 Mathematical Analysis of the Problem Kernel Size

In this section, we bound the maximum number of vertices in a reduced instance. We need the following notation concerning rooted trees. We use *node* to refer to a vertex of a tree. For a rooted tree T let $L(T)$ denote the *leaves* of T (that is, the nodes of degree one). The nodes

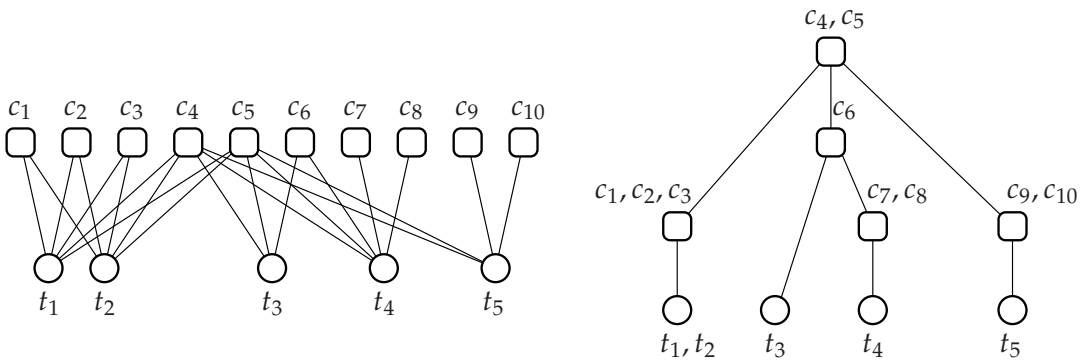

Figure 3: An M-free graph G and the corresponding tree $T_{cis}(G)$.

in $V(T) \setminus L(T)$ are denoted as *inner nodes*. The root of T is denoted by $r(T)$. Moreover, for a node $v \in V(T)$, the subtree rooted at v is denoted by T_v. We classify the children of a node as follows. We refer to a child of a node as its *leaf child* if it is a leaf , otherwise it is called its *non-leaf child*. We speak of the leaves (inner nodes) of a forest to refer to the union of the leaves (inner nodes) of the trees of the forest.

Given a connected and M-free graph $G = (V_c, V_t, E)$, one can construct a rooted tree T with node set $V_t \cup V_c$ and with $L(T) = V_t$ such that $t_i \in V_t$ is a descendant of $c_j \in V_c$ iff $t_i \in N_G(c_j)$, see [4, 15, 21] for details. Note that the critical independent set graph of an M-free graph is M-free. Hence, we can find a tree with the property that every leaf one-to-one corresponds to a critical independent set of the t-vertices and every inner vertex one-to-one corresponds to a critical independent set of the c-vertices. For an M-free graph G, this tree is denoted by $T_{cis}(G)$. Figure 3 shows an M-free graph G together with $T_{cis}(G)$.

The following easy observations are helpful in the analysis of the kernel size.

1. Every inner vertex of T_{cis} has at most one leaf child, and
2. every inner vertex with at most one non-leaf child has exactly one leaf child.

Now, we arrive at our main result.

THEOREM 8. MINIMUM-FLIP CONSENSUS TREE *admits an $O(k^3)$-vertex problem kernel .*

PROOF. Consider a reduced instance $(G = (V_c, V_t, E), k)$. We show that if (G, k) is a yes-instance, then the number of vertices in $V_c \cup V_t$ is bounded by $O(k^3)$.

If (G, k) is a yes-instance, then there exists an optimal solution S of size at most k. That is, the graph $G_S := G\Delta S$ is M-free. Vertices that are involved in an edge modification are called *affected* in the following. Let X_c denote the c-vertices that are affected by an edge modification in S and let Y_c denote the c-vertices that are not affected by any edge modification. Analogously, we define X_t and Y_t. Note that since every edge modification involves a c-vertex and a t-vertex, we have that $|X_c| \leq k$ and $|X_t| \leq k$.

Let $G_{S,1}, G_{S,2}, \ldots, G_{S,p}$ denote the connected components of G_S. Recall that for every connected component $T_i := T_{cis}(G_{S,i})$ denotes the rooted tree corresponding to the critical independent set graph of $G_{S,i}$. Moreover, let T denote the forest containing all T_i. Recall that the leaves of T one-to-one correspond to the critical independent sets of V_t in G_S and that

the inner nodes of T one-to-one correspond to the critical independent sets of V_c in G_S. For a node $z \in V(T)$, let $\mathcal{C}(z)$ denote the set of vertices contained in the critical independent set corresponding to z. Moreover, for $Z \subseteq V(T)$, we define $\mathcal{C}(Z) := \bigcup_{z \in Z} \mathcal{C}(z)$.

We partition the set of inner nodes into three sets A, B, and Q as follows. The set A contains all inner nodes z for which it holds that either $\mathcal{C}(z) \cap X_c \neq \emptyset$ or z has a leaf child w with $\mathcal{C}(w) \cap X_t \neq \emptyset$. Note that A has cardinality at most $2k$ since there are at most $2k$ affected vertices. Moreover, let B contain the inner nodes that are not contained in A and that have at least two non-leaf children. Finally, Q contains all inner nodes not contained in $A \cup B$.

Next, we bound the number of the vertices contained in the critical independent sets corresponding to the nodes in $A \cup B$ and their leaf children. To this end, we show the following.

1. For every inner node x not contained in A, there exists at least one node $y \in V(T_x)$ with $y \in A$.
2. The cardinality of B is at most $2k$.
3. Let $L_{A,B}$ denote the leaves adjacent to the nodes in $A \cup B$. The number of vertices contained in the critical independent sets corresponding to the nodes in $A \cup B \cup L_{A,B}$ is $O(k^2)$.

1.) Assume that there exists an inner node $x \in V(T) \setminus (L(T) \cup A)$ such that $V(T_x) \cap A = \emptyset$. That is, no vertex in $\mathcal{C}(V(T_x))$ is affected. Consider a vertex $c \in \mathcal{C}(x)$. We show that c is not contained in any conflict in G, contradicting the fact that G is reduced with respect to Reduction Rule 3. First, for every vertex $y \in \mathcal{C}(V(T_x))$, it holds that $N_G(y) \subseteq N_G(c)$ since $N_{G_S}(y) \subseteq N_{G_S}(c)$ and S does not affect c or y. Second, for every vertex $y \in \mathcal{C}(V(T) \setminus V(T_x))$, it holds that $N_{G_S}(c) \cap N_{G_S}(y) = \emptyset$ or $N_{G_S}(c) \subseteq N_{G_S}(y)$. But since neither c nor any vertex in $N_{G_S}(c)$ is modified, this implies that $N_G(c) \cap N_G(y) = \emptyset$ or $N_G(c) \subseteq N_G(y)$. This means that c is not contained in any conflict in G.

2.) Consider the forest T' that results from deleting all leaves of T. Note that B is a subset of the nodes from T' with at least two children. From 1) it follows directly that the leaves of T' are contained in A and, hence, their number is bounded by $2k$. Since the number of inner nodes with at least two children is bounded by the number of leaves, we get that $|B| \leq 2k$.

3.) First, note that $|A \cup B| \leq 4k$ since A and B each have cardinality at most $2k$. Moreover, $|L_{A,B}| \leq 4k$ since every inner node has at most one leaf child. For every node $y \in A \cup B \cup L_{A,B}$, define $\mathcal{C}'(y) := \mathcal{C}(y) \setminus (X_c \cup X_t)$. For every $y \in A \cup B \cup L_{A,B}$, since no vertex in $\mathcal{C}'(y)$ is affected, $\mathcal{C}'(y)$ forms a critical independent set in G and—since G is reduced with respect to Reduction Rule 1—we thus get that $|\mathcal{C}'(y)| \leq k + 1$. Putting all together, we obtain

$$|\mathcal{C}(A \cup B \cup L_{A,B})| \leq |X_c| + |X_t| + \sum_{y \in A \cup B \cup L_{A,B}} |\mathcal{C}'(y)| \leq 2k + 4k(k+1).$$

It remains to bound the number of the vertices contained in $\mathcal{C}(Q \cup L_Q)$, where L_Q denotes the leaves adjacent to the nodes in Q. Observe that each inner node contained in Q (and hence not contained in $A \cup B$) has exactly one leaf and one non-leaf child. That is, in the the forest $T' := T - L(T)$ these vertices have degree two. Recall that all leaves of T' (see 2.) above) are contained in A and hence $|L(T')| \leq 2k$. Consider a path $P =$

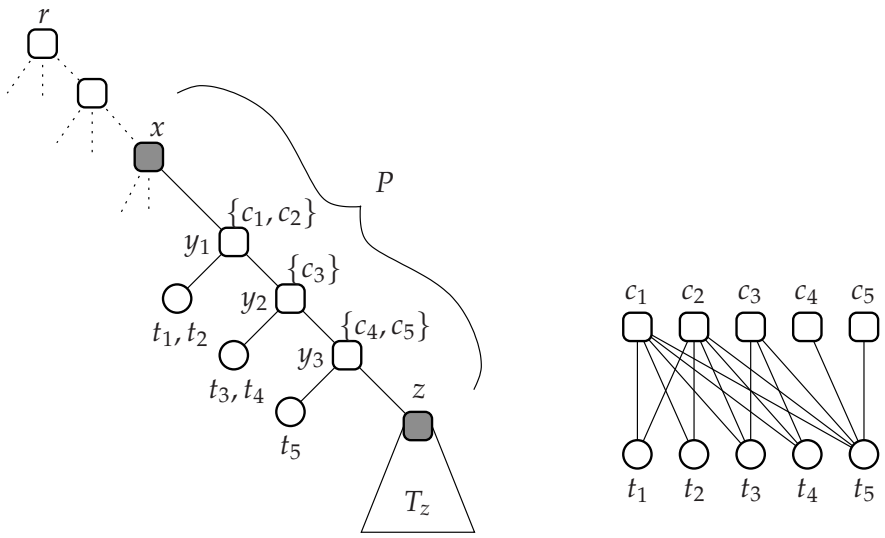

Figure 4: A degree-two-path P and the corresponding chain graph. Herein, $\mathcal{C}(y_1) = \{c_1, c_2\}$, $\mathcal{C}(y_2) = \{c_3\}$, $\mathcal{C}(y_3) = \{c_4, c_5\}$, and $\mathcal{C}(y_4) = \{c_6\}$.

$(\{x, y_1\}, \{y_1, y_2\}, \ldots, \{y_{l-1}, y_l\}, \{y_l, z\})$ in T' with $y_i \in Q$ for all $1 \leq i \leq l$ and $x, z \in A \cup B$. Such a path is called a *degree-two-path* in the following since by the above discussion $\deg_{T'}(y_i) = 2$ for all $1 \leq j \leq l$. Further, for every y_i, let w_i denote the leaf child of y_i in T. Note that in the forest T', there are at most $8k$ degree-two-paths since $L(T') \subseteq A$ and $|A \cup B| \leq 4k$. In the following, we bound the length of each degree-two-path by $2(k+1)$. Hence, for each such path we have

$$\sum_{i=1}^{l} (|\mathcal{C}(y_i)| + |\mathcal{C}(w_i)|) \leq l \cdot (2(k+1)) \leq (2(k+2)) \cdot 2(k+1)$$

vertices in G. Adding up over the at most $8k$ degree-two-paths, this amounts to $8k \cdot 2(k+1)(2(k+2)) \leq 32k(k+1)(k+2)$ vertices, yielding the bound of $O(k^3)$ vertices in total.

Next, we bound the length of each degree-two-path. To this end, consider such a degree-two-path $P = (\{x, y_1\}, \{y_1, y_2\}, \ldots, \{y_{l-1}, y_l\}, \{y_l, z\})$ in T', that is, $x, z \in A \cup B$ and $y_i \in Q$ for all $1 \leq i \leq l$. Without loss of generality, we assume that y_l is a descendent of y_1. See Fig. 4 for an example. Let $C_P := \bigcup_{i=1}^{l} \mathcal{C}(y_i)$ and $T_P := \bigcup_{i=1}^{l} \mathcal{C}(w_i)$.

We show that (C_P, T_P) forms a P-structure in G. First, note that $C_P \subseteq V_c$ and $T_P \subseteq V_t$. Next, note that $G[C_P \cup T_P]$ forms a chain graph. This can seen as follows. In G_S a vertex in $\mathcal{C}(y_1)$ is clearly adjacent to all vertices in T_P, a vertex in $\mathcal{C}(y_2)$ is adjacent to all vertices in $T_P \setminus \mathcal{C}(w_1)$, a vertex in $\mathcal{C}(y_3)$ is adjacent to all vertices in $T_P \setminus \mathcal{C}(\{w_1, w_2\})$, and so on. Hence, $G_S[C_P \cup T_P]$ is a chain graph and, since no vertex in C_P is involved in an edge modification, we have that $G[C_P \cup T_P]$ forms a chain graph, too (see Fig. 4). Next, we show that C_P and T_P fulfill the second and third property of a P-structure. On the one hand, every vertex in C_P is adjacent in G_S to all vertices contained in the critical independent sets corresponding to the leaves in T_z and, hence, for all $c, c' \in C_P$, we have $N_{G_S}(c) \setminus T_P = N_{G_S}(c') \setminus T_P$. Since no vertex in C_P is affected, this implies that $N_G(c) \setminus T_P = N_G(c') \setminus T_P$ for

all $c, c' \in C_P$. On the other hand, every vertex $t \in T_P$ is adjacent in G_S (and hence in G) to all c-vertices contained in a critical independent set on the path from the root r to z. Hence, for any two vertices $t, t' \in T_P$ it holds that $N_G(t) \setminus T_P = N_G(t') \setminus T_P$. In summary, (C_P, T_P) forms a P-structure.

Finally, we show that $l \leq 2(k+1)$. Assume towards a contradiction that $l > 2(k+1)$. This implies that $|T_P| > 2(k+1)$, too, since every y_i has exactly one leaf child that corresponds to a (non-empty) critical independent set of V_t. Hence, $|T_P| > 2(k+1)$ and thus all conditions to apply Reduction Rule 4 are fulfilled: a contradiction to the fact that G is reduced. ∎

Applying the technique of interleaving [20] to our kernelization and the search tree algorithm by Böcker et al. [1], we obtain an "additive FPT" algorithm for MINIMUM-FLIP CONSENSUS TREE.

COROLLARY 9. MINIMUM-FLIP CONSENSUS TREE *can be solved in running time* $O(4.83^k + \text{poly}(|V_c|, |V_t|))$.

6 Conclusion

As to future research, first of all, we want to implement and test the efficiency of our data reduction rules. Second, improving the polynomial running time of our data reduction rules is desirable. Obviously, obtaining data reduction rules that lead to a quadratic-vertex or linear-vertex kernel remains as an open question. Moreover, studying edge-weighted problem variants would be theoretically interesting. Finally, it would be interesting to adapt our data reduction to yield a full kernel (see [5]) for MINIMUM-FLIP CONSENSUS TREE.

References

[1] S. Böcker, Q. B. Bui, and A. Truss. An improved fixed-parameter algorithm for minimum-flip consensus trees. In *Proc. 3rd IWPEC*, volume 5018 of *LNCS*, pages 43–54. Springer, 2008.

[2] H. L. Bodlaender. A cubic kernel for feedback vertex set. In *Proc. 24nd STACS*, volume 4393 of *LNCS*, pages 320–331. Springer, 2007.

[3] P. Burzyn, F. Bonomo, and G. Durán. NP-completeness results for edge modification problems. *Discrete Applied Mathematics*, 154(13):1824–1844, 2006.

[4] D. Chen, O. Eulenstein, D. Fernández-Baca, and M. Sanderson. Minimum-flip supertrees: Complexity and algorithms. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 3(2):165–173, 2006. Preliminary version presented at *COCOON '02*.

[5] P. Damaschke. Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. *Theoretical Computer Science*, 351(3):337–350, 2006.

[6] M. R. Fellows. The lost continent of polynomial time: Preprocessing and kernelization. In *Proc. 2nd IWPEC*, volume 4169 of *LNCS*, pages 276–277. Springer, 2006.

[7] 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.

[8] H. Fleischner, E. Mujuni, D. Paulusma, and S. Szeider. Covering graphs with few complete bipartite subgraphs. In *Proc. 27th FSTTCS*, volume 4855 of *Lecture Notes in Computer Science*, pages 340–351. Springer, 2007.

[9] 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.

[10] J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier. Data reduction, exact, and heuristic algorithms for clique cover. In *Proc. 8th ALENEX*, pages 86–94. SIAM, 2006. Long version to appear in *ACM Journal of Experimental Algorithmics*.

[11] J. Guo. A more effective linear kernelization for Cluster Editing. In *Proc. 1st ESCAPE*, volume 4614 of *LNCS*, pages 36–47. Springer, 2007. Long version to appear in *Theoretical Computer Science*.

[12] J. Guo. Problem kernels for NP-complete edge deletion problems: split and related graphs. In *Proc. 18th ISAAC*, volume 4835 of *LNCS*, pages 915–926. Springer, 2007.

[13] J. Guo, F. Hüffner, C. Komusiewicz, and Y. Zhang. Improved algorithms for bicluster editing. In *Proc. 5th TAMC*, volume 4978 of *LNCS*. Springer, 2008.

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

[15] D. Gusfield. Efficient algorithms for inferring evolutionary trees. *Networks*, 21:19–28, 1991.

[16] 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.

[17] F. Hüffner, R. Niedermeier, and S. Wernicke. Techniques for practical fixed-parameter algorithms. *The Computer Journal*, 51(1):7–25, 2008.

[18] A. Natanzon, R. Shamir, and R. Sharan. Complexity classification of some edge modification problems. *Discrete Applied Mathematics*, 113(1):109–128, 2001.

[19] R. Niedermeier. *Invitation to Fixed-Parameter Algorithms*. Number 31 in Oxford Lecture Series in Mathematics and Its Applications. Oxford University Press, 2006.

[20] R. Niedermeier and P. Rossmanith. A general method to speed up fixed-parameter-tractable algorithms. *Information Processing Letters*, 73:125–129, 2000.

[21] I. Pe'er, T. Pupko, R. Shamir, and R. Sharan. Incomplete directed perfect phylogeny. *SIAM Journal on Computing*, 33(3):590–607, 2004.

[22] F. Protti, M. D. da Silva, and J. L. Szwarcfiter. Applying modular decomposition to parameterized cluster editing problems. *Theory of Computing Systems*. To appear.

[23] S. Thomasse. A quadratic kernel for feedback vertex set. In *Proc. 20th SODA*. SIAM, 2009. To appear.

[24] M. Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM Journal on Algebraic Discrete Methods*, 2(1):77–79, 1981.