Polynomial Kernels for Deletion to Classes of Acyclic Digraphs*

Matthias Mnich[1] **and Erik Jan van Leeuwen**[2]

1 **Universität Bonn, Bonn, Germany**
mmnich@uni-bonn.de

2 **Max-Planck-Institut für Informatik, Saarbrücken, Germany**
erikjan@mpi-inf.mpg.de

Abstract

We consider the problem to find a set X of vertices (or arcs) with $|X| \le k$ in a given digraph G such that $D = G - X$ is an acyclic digraph. In its generality, this is Directed Feedback Vertex Set or Directed Feedback Arc Set respectively. The existence of a polynomial kernel for these problems is a notorious open problem in the field of kernelization, and little progress has been made.

In this paper, we consider both deletion problems with an additional restriction on D, namely that D must be an out-forest, an out-tree, or a (directed) pumpkin. Our main results show that for each of these three restrictions the vertex deletion problem remains NP-hard, but we can obtain a kernel with $k^{O(1)}$ vertices on general digraphs G. We also show that, in contrast to the vertex deletion problem, the arc deletion problem with each of the above restrictions can be solved in polynomial time.

1998 ACM Subject Classification F2.2 Nonnumerical Algorithms and Problems

Keywords and phrases directed feedback vertex/arc set, parameterized algorithms, kernels

Digital Object Identifier 10.4230/LIPIcs.STACS.2016.55

1 Introduction

In this paper, we study the problem of removing a (small) subset of vertices X from a graph G such that the resulting graph $G - X$ is acyclic. On undirected graphs, this translates immediately to the property that $G - X$ is a forest or (if we insist that $G - X$ is connected) a tree. The problem to decide whether a given undirected graph G has a set $X \subseteq V(G)$ of size at most a given integer k such that $G - X$ is a forest or a tree is known as Feedback Vertex Set and Tree Deletion Set respectively.

Over the past years, we have gotten to understand the complexity of Feedback Vertex Set and Tree Deletion Set quite well. Both problems are NP-hard [14, 20]. It is long known that the minimization version of Feedback Vertex Set admits a polynomial-time 2-approximation algorithm [2] and that Feedback Vertex Set admits a polynomial kernel parameterized by k [19] (see e.g. [8, 11] for background on kernelization). The minimization version of Tree Deletion Set, in contrast, cannot be polynomial-time approximated within a factor $O(n^{1-\epsilon})$ for any $\epsilon > 0$ [20], unless P = NP. However, Tree Deletion Set was recently shown to admit a polynomial kernel (when parameterized by k) [13].

The usual way to generalize Feedback Vertex Set and Tree Deletion Set to digraphs is to insist that the resulting digraph has no directed cycle. Indeed, the problem to

* Research of M.M. was supported by ERC Starting Grant 306465 (BeyondWorstCase).

decide whether a given digraph G has a set $X \subseteq V(G)$ of size at most a given integer k such that $G - X$ is a (connected) acyclic digraph is known as Directed Feedback Vertex Set (Connected DAG Vertex Deletion Set). In contrast to their undirected counterparts, the complexity situations for Directed Feedback Vertex Set and Connected DAG Vertex Deletion Set are very much unclear.

It is known that Directed Feedback Vertex Set is NP-hard [14], even on tournaments [18]. Connected DAG Vertex Deletion Set is NP-hard and cannot be polynomial-time approximated within a factor $O(n^{1-\epsilon})$ for any $\epsilon > 0$ [20], but we are not aware of any results on the parameterized complexity of this problem. Directed Feedback Vertex Set is polynomial-time approximable within a factor of $O(\log|V(G)|\log\log|V(G)|)$ on general digraphs [9, 17], but it is open whether this is best possible. Directed Feedback Vertex Set has a kernel of exponential size $k^{O(k)}$, as the problem was shown fixed-parameter tractable by Chen et al. [4], but it is unknown whether a polynomial kernel exists. In fact, this question remains open despite being posed several times [4, 10, 8, 6, 5].

There is limited insight into whether Directed Feedback Vertex Set could admit a polynomial kernel. Abu-Khzam [1] (see also Dom et al. [7]) showed that Directed Feedback Vertex Set admits a polynomial kernel if the given digraph is a (bipartite) tournament and Bang-Jensen et al. [3] recently extended this to generalizations of tournaments. We are not aware of polynomial kernels for Directed Feedback Vertex Set on other restricted classes of digraphs. This suggests to explore other roads towards an answer to the open question of a polynomial kernel for Directed Feedback Vertex Set.

Our Contributions

We study a different translation of Feedback Vertex Set and Tree Deletion Set to digraphs. Instead of transferring the property that the resulting graph should be acyclic to digraphs, we transfer the property that the resulting graph should be a forest or tree. To this end, we consider the notion of an *out-tree*, which is a digraph where each vertex has in-degree at most 1 and the underlying (undirected) graph is a tree. An *out-forest* is a disjoint union of out-trees. This leads to the following parameterized problems:

Out-Forest/Out-Tree Vertex Deletion Set
Input: A digraph G and an integer k.
Question: Is there a set $X \subseteq V(G)$ with $|X| \leq k$ s.t. $G - X$ is an out-forest/out-tree?

Note that these problems can also be viewed as a restricted version of Directed Feedback Vertex Set and Connected DAG Vertex Deletion Set. Here, instead of restricting the input graph G as Abu-Khzam [1] and Dom et al. [7] did when they considered tournaments, we consider general digraphs G as input but restrict what kind of acyclic digraph the resulting digraph $G - X$ should be.

Thinking further in this direction, we consider another restriction on the resulting digraph, namely that it should be a pumpkin. A digraph is a *pumpkin* if it consists of a source vertex s and a sink vertex t ($s \neq t$), together with a collection of internally vertex-disjoint induced directed paths from s to t. Note that the underlying graph of a pumpkin is not acyclic, in contrast to out-forests and out-trees. This leads to the following parameterized problem:

Pumpkin Vertex Deletion Set
Input: A digraph G and an integer k.
Question: Is there a set $X \subseteq V(G)$ with $|X| \leq k$ s.t. $G - X$ is a pumpkin?

We consider all three problems on general digraphs, and observe that each is NP-hard, even on acyclic digraphs. More importantly, we show that all three problems admit a polynomial kernel.

▶ **Theorem 1.** Out-Tree Vertex Deletion Set *is* NP*-hard, even on acyclic digraphs, but admits a kernel with* $O(k^3)$ *vertices.*

▶ **Theorem 2.** Out-Forest Vertex Deletion Set *is* NP*-hard, even on acyclic digraphs, but admits a kernel with* $O(k^3)$ *vertices.*

▶ **Theorem 3.** Pumpkin Vertex Deletion Set *is* NP*-hard, even on acyclic digraphs, but admits a kernel with* $O(k^{18})$ *vertices.*

The polynomial kernel for Out-Tree Vertex Deletion Set, presented in Sec. 2, relies on a large set of reduction rules that heavily exploit that vertices of out-trees have in-degree at most 1. Although this 'forbidden structure' seems to lend itself naturally to a characterization by forbidden induced subgraphs that can be attacked through a standard approach using the Sunflower Lemma (cf. [12, Lemma 3.2]), the demand that the resulting digraph be *connected* means that substantially different methods must be employed to obtain meaningful structure.

After applying our set of reduction rules, the underlying graph of the resulting digraph G' contains a small feedback vertex set F. It then remains to show that the forest $G' - F$ has bounded size by analyzing the interaction of $G' - F$ with F. In particular, we argue that the leaves of $G' - F$ can be split into four types, and bound the number of leaves of each type. In the analysis of one of the types (the fourth), we adapt some of the rules of the polynomial kernel for Tree Deletion Set by Giannopoulou et al. [13] to the directed case; the analysis for the other cases is new and specific to Out-Tree Vertex Deletion Set.

The kernel for Out-Forest Vertex Deletion Set follows the same lines, but requires an additional reduction rule, presented in Sec. 3. We believe that the Sunflower Lemma could yield an alternative road to a polynomial kernel for Out-Forest Vertex Deletion Set (as connectedness is no longer an issue), but we chose to instead present our simple extension of the kernel for Out-Tree Vertex Deletion Set.

The polynomial kernel for Pumpkin Vertex Deletion Set, presented in Sect. 4, uses completely different methods. We first show that there are only poly(k) candidates for the source and sink of the pumpkin. Therefore, we can split the instance into poly(k) new instances with an annotated source and sink, each of which we subsequently kernelize. The resulting kernelized instances can be seen as a Turing kernel of the problem. Instead of being satisfied with this, we show that we can modify and combine these kernelized instances into a single instance of Pumpkin Vertex Deletion Set, which forms the final kernel.

The NP-hardness results and missing proofs are deferred to the full version of this paper.

Edge and Arc Deletion Problems

Instead of deleting vertices to get an acyclic graph, we also consider the problem of deleting edges or arcs from a given (di)graph to obtain an acyclic (di)graph. On undirected graphs, the problem to delete edges to obtain a forest or tree can easily be shown to be polynomial-time solvable by reducing to finding a spanning forest/tree. On digraphs, however, the complexity situation is quite different. In fact, it can be readily shown that the problem of deleting arcs from a digraph to remove all cycles must have the same complexity as Directed Feedback Vertex Set [9]. Therefore, one wonders how this affects the three problems of this paper.

The arc-deletion versions of our problems are defined as follows. Given a subset B of the arcs of a digraph G, the digraph *induced* by B is the graph with vertex set equal to the set of endpoints of the arcs in B and arc set equal to B.

OUT-FOREST/OUT-TREE/PUMPKIN ARC DELETION SET	
Input:	A digraph G and an integer k.
Question:	Is there a set $X \subseteq A(G)$ with $|X| \leq k$ s.t. the arcs of $A(G) - X$ induce an out-forest/out-tree/pumpkin?

▶ **Theorem 4.** OUT-FOREST ARC DELETION SET, OUT-TREE ARC DELETION SET, *and* PUMPKIN ARC DELETION SET *can be solved in polynomial time.*

A shortened proof of this theorem is in Sect. 5; full proofs are deferred to the full version.

Throughout this paper, we consider digraphs G with vertex set $V(G)$ and arc set $A(G)$. For a digraph G, we denote its underlying undirected graph by $\langle G \rangle$. The in-degree and the out-degree of a vertex $v \in V(G)$ is denoted $d^-(v)$ resp. $d^+(v)$.

For a digraph G and distinct vertices $u, v \in V(G)$, call $P = [w_0, \dots, w_\ell]$ an *induced directed u, v-path* (*of length ℓ*) if $u = w_0$, $v = w_\ell$, (w_i, w_{i+1}) are arcs of G for $i = 0, \dots, \ell - 1$ and $d^-(w_i) = d^+(w_i) = 1$ for $i = 1, \dots, \ell - 1$.

2 Polynomial Kernel for Out-Tree Vertex Deletion

For a digraph G, call a set $S \subseteq V(G)$ an *out-tree vertex deletion set* if $G - S$ is an out-tree. To obtain a polynomial kernel for OUT-TREE VERTEX DELETION SET parameterized by solution size k, we first generalize the problem to its vertex-weighted variant, defined as:

WEIGHTED OUT-TREE VERTEX DELETION SET	
Input:	A digraph G with weight function $w : V(G) \to \mathbb{N}$; an integer k.
Question:	Is there a set $S \subseteq V(G)$ of weight $w(S) = \sum_{v \in S} w(v) \leq k$ so that $G - S$ is an out-tree?

Allowing vertices to carry weights will allow for more flexible reduction rules. Using four relatively standard reduction rules (described in the full version), we can impose the following structure on instances of WEIGHTED OUT-TREE VERTEX DELETION SET:

▶ **Lemma 5.** *Given an instance (G, w, k) of* WEIGHTED OUT-TREE VERTEX DELETION SET, *in polynomial time we can construct an instance (G', w', k') such that:*

1. *$d^-(v) \leq k + 1$ for each $v \in V(G')$;*
2. *$w'(v) \leq k + 1$ for each $v \in V(G')$;*
3. *no vertex $v \in V(G')$ has $d^-(v) = 1$ and $d^+(v) = 0$;*
4. *every induced directed path of G' has length at most 4;*
5. *(G, w) has an out-tree vertex deletion set of weight at most k if and only if (G', w') has an out-tree vertex deletion set of weight at most k'*

We now define several novel reduction rules that are specific to WEIGHTED OUT-TREE VERTEX DELETION SET. A *collision* in G is an ordered triple (u, m, v) of distinct vertices u, m, v such that $(u, m), (v, m) \in A(G)$. A collision only demands that $(u, m), (v, m) \in A(G)$; it does not specify anything about other arcs between u, m, v.

▶ **Lemma 6** (Collision Star Rule)**.** *Let $(u_1, m_1, v), \dots, (u_{k+1}, m_{k+1}, v)$ be collisions that pairwise intersect only in v. Let $G' = G - v$, let $k' = k - w(v)$ and $w' = w_{|V(G')}$. Then (G, w) has an out-tree vertex deletion set of weight at most k if and only if (G', w') has an out-tree vertex deletion set of weight at most k'.*

Proof. Let S be an out-tree vertex deletion set of (G, w) of weight at most k. We argue that $v \in S$; this then implies immediately that $S \setminus \{v\}$ is an out-tree vertex deletion set of (G', w') with weight at most $k' = k - w(v)$. To this end, suppose that $v \notin S$; then from each collision (u_i, m_i, v), at least one of u_i, m_i belongs to S, since S must intersect all collisions (out-trees are free of collisions). As each of u_i, m_i has weight at least 1 and the pairs $(u_1, m_1), \ldots, (u_{k+1}, m_{k+1})$ are pairwise vertex-disjoint, this contradicts that $w(S) \leq k$; hence, $v \in S$.

Conversely, let S' be an out-tree vertex deletion set of (G', w') of weight at most k'. As out-tree vertex deletion sets are closed under taking supersets, $S = S' \cup \{v\}$ is an out-tree vertex deletion set of (G, w). Further, the weight of S is equal to $w(S) = w'(S') + w(v) \leq k' + w(v) = k$.

We can implement the rule in polynomial time by considering each v in turn and then running a maximum matching algorithm. For a vertex v, let N_v^+ denote its set of out-neighbors. Let N_v^* denote the set of in-neighbors of the vertices in N_v^+, where we ensure that v is not placed in N_v^*. Now consider an auxiliary graph H_v on $N_v^+ \cup N_v^*$, were $u \in N_v^+$ and $w \in N_v^*$ are adjacent if and only if there is an arc from w to u in G. Then any set of collisions that pairwise intersect only in v (as in the lemma statement) induces a matching in H_v, and conversely, any matching in H_v induces a set of collisions that pairwise intersect only in v (as in the lemma statement). Hence, it suffices to find, for each $v \in V(G)$ a matching of size at least k in H_v, which can be done in polynomial time. ◀

Note again that the rule only specifies that the arcs (u_i, m_i) and (m_i, v) for $i = 1, \ldots, k+1$ should belong to $A(G)$; it does not specify anything about other arcs between these vertices.

▶ **Lemma 7** (Source Rule). *If G contains at least $k+2$ vertices of in-degree 0, then (G, w) does not contain an out-tree vertex deletion set of weight at most k.*

We apply the above rules exhaustively, before continuing to the next rule.

▶ **Lemma 8** (Feedback Vertex Set Rule). *If a 2-approximate minimum size undirected feedback vertex set of $\langle G \rangle$ has size more than $2k$, then (G, w) does not admit an out-tree vertex deletion set of weight at most k.*

This rule is correct, because $\langle G - S \rangle$ is acyclic, where S is an out-tree vertex deletion set of (G, w). We thus assume that $\langle G \rangle$ has a feedback vertex set F of size at most $2k$.

We next argue that if $G - F$ has many vertex-disjoint collisions, then G does not admit an out-tree vertex deletion set of low weight, because an out-tree does not contain any collisions. Observe that finding the maximum number of vertex-disjoint collisions in a digraph is in general an NP-hard problem, because of a straightforward reduction from the P_2-matching problem, which is known to be NP-hard [15]. However, here we only need to solve this task in $G - F$, which is a forest. We are then able to employ a greedy algorithm that 'cuts away' collisions in a bottom-up fashion and finds a largest set of vertex-disjoint collisions in $G - F$ in polynomial time.

▶ **Lemma 9** (Disjoint Collisions Rule). *If $G - F$ contains more than k vertex-disjoint collisions, then (G, w) does not admit an out-tree vertex deletion set of weight at most k.*

Assume now that Lemma 9 has been applied, and that F (as before) is a feedback vertex set of $\langle G \rangle$. Let $T_1, \ldots, T_c$ denote the set of connected components of $G - F$; then each underlying undirected component $\langle T_i \rangle$ is a tree. Let $\mathcal{L}$ denote the set of leaves of $\langle T_1 \rangle, \ldots, \langle T_c \rangle$. Our strategy to prove a polynomial kernel will be to first bound $|\mathcal{L}|$, and therefrom bound both c and $\sum_{i=1}^{c} |V(T_i)|$.

Bounding the Number of Leaves

We consider four types of leaves. Throughout, we assume that a leaf of type i is not of type $i-1, \dots, 1$.

Type 1: Leaves with an arc to a vertex of F.

By Lemma 5(1), $\sum_{v \in F} d^-(v) \leq |F|(k+1) \leq 2k(k+1)$. Hence, the number of leaves with an arc to a vertex of F is at most $2k(k+1)$.

Type 2: Leaves without any arc to or from a vertex of F.

By Lemma 5(3), such a leaf must be a source of G. We can bound the number of leaves of type 2 by $k+1$, since Lemma 7 does not apply.

Type 3: Leaves with an arc from a vertex of F whose unique incident arc in $G-F$ is an out-arc.

Let $v_1, \dots, v_\ell$ denote those leaves. An *F-disjoint path* is a directed path P in G between two vertices $u, v \in V(G)$ such that no vertex of P belongs to F; in particular, $u, v \notin F$. Call a vertex $m \notin F$ a *mixer for i, j* (where i, j are distinct) if there exist F-disjoint paths from v_i to m and from v_j to m; we also say that i, j get *mixed* at m. If $m \notin F$ is a mixer for some i, j, then we simply call m a *mixer*. Observe that for any $m \notin F$, there is an F-disjoint path in G to some v_i only if v_i and m are part of the same tree of $\langle G \rangle - F$; therefore, if such a path exists, then it is unique.

We now construct a set M of mixers, as follows. Initialize $M = \emptyset$ and $I = \{1, \dots, \ell\}$. Iteratively find a mixer m for some $\{i, j\} \subseteq I$ such that for any $\{i', j'\} \subseteq I$ that get mixed at m, the F-disjoint path from v_i to m or F-disjoint path from v_j to m is free of mixers for i, j', and i', j, and i', j' as internal vertex. Then add m to M, and remove all indices from I that get mixed at m. Denote by I' the set of indices in I at the end of this procedure.

We can show that the set of mixers induces a set of vertex-disjoint collisions of size $|M|$ by using that two unique indices i, j get mixed at each $m \in M$; the F-disjoint paths from v_i and v_j to m then ensure that m has two unique in-neighbors. Hence, the following lemma follows from Lemma 9:

▶ **Lemma 10.** *For the constructed set M of mixers, $|M| \leq k$.*

Consider any index $i \in \{1, \dots, \ell\} \setminus I'$, and let m denote the mixer that was added to M at the step that i was removed from I. Let P_i be an F-disjoint path from v_i to m that is guaranteed by m being a mixer. It follows from the construction of M that $\{P_i \mid i \in \{1, \dots, \ell\} \setminus I'\}$ forms a collection of F-disjoint paths that are pairwise disjoint except possibly for their end vertices. By Lemma 5(1), $d^-(m) \leq k+1$ for each $m \in M$. Hence, $\ell - |I'| \leq |M|(k+1) \leq k(k+1)$.

To bound $|I'|$, we find a set of inclusion-wise maximal directed paths starting in the vertices in $\{v_i \mid i \in I'\}$. The ends (not equal to v_i) of these paths yield a set of collisions, which are almost disjoint. Then Lemmas 6 and 9 combined imply:

▶ **Lemma 11.** $|I'| \leq 3k^2 + 2k$.

Adding up, we have that $\ell \leq 3k^2 + 2k + k(k+1) = 4k^2 + 3k$.

Type 4: Leaves with an arc from a vertex of F whose arc in $G-F$ is an in-arc.

To bound the number of leaves of Type 4, we need an extra reduction rule. This rule is similar to one applied by Giannopoulou et al. [13] in a different context. The general idea is to translate the constraints on leaves of Type 4 into a set of linear equations, then kernelize this set of equations, and use that kernel to reduce the number of leaves of Type 4. Since this works in the same way as in the paper by Giannopoulou et al. [13], we defer the details to the full version.

▶ **Lemma 12.** *The number of leaves of Type 4 is at most* $2k(k+1)^2 + k + 1$.

After analyzing all types of leaves, we can conclude that $|\mathcal{L}| \leq 2k^3 + 10k^2 + 9k + 2$. Using standard arguments about trees in conjunction with the properties ensured by Lemma 5(4), we can show the following:

▶ **Lemma 13.** *The number of vertices in* $T_1, \ldots, T_c$ *is* $O(k^3)$ *in total.*

Proof. Since we already bounded the number of leaves of $T_1, \ldots, T_c$, it suffices to bound the number of internal vertices. Let D denote the set of internal vertices of $T_1, \ldots, T_c$ that have degree at least 3 in $\langle G \rangle$. Since the number $|\mathcal{L}|$ of leaves is $O(k^3)$, we have $|D| = O(k^3)$ as well by standard arguments on trees.

It remains to bound the number of internal vertices of degree 2 in $\langle G \rangle$. The number of such vertices that neighbor a leaf is $O(k^3)$, so it remains to consider vertices that have distance at least 2 to every leaf in $\langle G \rangle$.

We start by bounding the number of such vertices involved in a collision. First, consider collisions (u, m, v) for which m has degree at least 3 in $\langle G \rangle$. Such collisions involve at most $\sum_{v \in D} d_{\langle G \rangle}(v)$ vertices of degree 2 in $\langle G \rangle$. Since this sum is bounded by $2|\mathcal{L}|$, we obtain a bound of $O(k^3)$ for such vertices.

Now consider collisions (u, m, v) for which m has degree 2 in $\langle G \rangle$ but at least one of u, v has degree at least 3 in $\langle G \rangle$. The number of vertices of degree 2 involved in such collisions is at most $2 \sum_{v \in D} d_{\langle G \rangle}(v) = O(k^3)$.

It remains to count vertices involved in collisions that do not touch an internal vertex of degree at least 3 in $\langle G \rangle$. Note that any such collision can overlap at most two others. Hence, by Lemma 9, at most $7k$ vertices can be involved, or we can reject (G, k) as a "no"-instance. Therefore, $O(k^3)$ internal vertices of degree 2 are involved in a collision.

Any internal vertex of degree 2 that is not involved in a collision must lie on a directed path between two vertices that are either of degree at least 3 or are involved in a collision. By another rule deferred to the full version of this paper, all directed paths have length at most 4; this leads to a bound of $4((\sum_{v \in D} d(v)) + O(k^3)) = O(k^3)$. ◀

Now, by Lemma 5(2), each vertex has weight at most $k + 1$, and thus this kernel can be encoded with $O((k^3) \log k)$ bits. This completes the proof of Theorem 1.

3 Polynomial Kernel for Out-Forest Vertex Deletion

Given a digraph G, we call a set $S \subseteq V(G)$ an *out-forest vertex deletion set* if $G - S$ is an out-forest. To obtain the polynomial kernel for Out-Forest Vertex Deletion Set, we proceed similarly as in the case of Out-Tree Vertex Deletion Set. Namely, we can argue that almost all reduction rules that apply to Out-Tree Vertex Deletion Set also work for Out-Forest Vertex Deletion Set. More precisely, one can argue that Lemmas 5, 6, 8, and 10, and 12 also apply. This way, we can bound the number of Type 1, Type 3 and Type 4 leaves by a polynomial in k in instances of Out-Forest Vertex Deletion Set that have been reduced by their respective rules.

However, Lemma 7 does not apply. Therefore, we cannot bound the number of Type 2 leaves by a polynomial in k in instances of Out-Forest Vertex Deletion Set that have been reduced by all rules except the one of Lemma 7. Instead, we propose the following additional rule:

▶ **Lemma 14** (Out-Forest Source Rule). *Let (G, w, k) be an instance of* Out-Forest Vertex Deletion Set. *If G contains a vertex v of in-degree 0 with unique out-neighbor u that itself has v as its unique in-neighbor, then let $G' = G - v$ and let $w' = w_{|V(G')}$. Then (G, w) has an out-forest vertex deletion set of weight at most k if and only if (G', w') has an out-forest vertex deletion set of weight at most k.*

Crucially, the rule is invalid for Out-Tree Vertex Deletion Set, because if u is part of the deletion set for the resulting instance, then we might also need to add v to the deletion set in the original instance.

Once an instance of Out-Forest Vertex Deletion Set has been reduced by all previous rules except the one of Lemma 7 and additionally by the one of Lemma 14, we can also bound the number of Type 2 leaves by a polynomial in k.

▶ **Lemma 15.** *There are at most $3k^2(k + 1)$ leaves of Type 2.*

Proof. We perform a marking scheme. Initially, no vertices are marked. Consider any unmarked leaf u of Type 2 for which its unique out-neighbor m is also unmarked. Since the rule of Lemma 14 cannot be performed, m has at least one other in-neighbor besides u. Let v be an arbitrary in-neighbor of m that is not u. Store the collision (u, m, v), and mark u, m, and v. Perform this procedure until no longer possible. At the end, each leaf of Type 2 has a marked vertex as its out-neighbor. Since each marked vertex has in-degree at most $k + 1$ by Lemma 5(1), it suffices to bound the number of marked vertices.

Consider all collisions $Z = \{(u_i, m_i, v_i)\}$ that were stored during the marking scheme. Whenever we added (u, m, v), the vertices u and m were unmarked. Therefore, for each $i \neq j$, $\{u_i, m_i\} \cap \{u_j, m_j\} = \emptyset$. However, the v-vertices might coincide between the different stored collisions. Let $Y = \{v_i \mid (u_i, m_i, v_i) \in Z\}$. Since Lemma 9 does not apply, $|Y| \leq k$. Now consider the set Y_v of collisions that were stored for a fixed vertex $v \in Y$, i.e., $Y_v := \{(u_i, m_i, v_i) \in Z \mid v_i = v\}$. Since Lemma 6 cannot be applied, $|Y_v| \leq k$. Therefore, $|Z| \leq |Y| \cdot \max_{v \in Y}\{|Y_v|\} \leq k^2$, and thus the number of marked vertices is at most $3k^2$. ◀

4 Polynomial Kernel for Pumpkin Vertex Deletion Set

We first give a simple property of the problem, and use it to give our first reduction rule. Let (G, k) be an instance of Pumpkin Vertex Deletion Set. Let $\mathrm{HI} = \{v \in V(G) \mid d^-(v) \geq k + 2\}$ and $\mathrm{HO} = \{v \in V(G) \mid d^+(v) \geq k + 2\}$.

▶ **Lemma 16** (High Degree Rule). *If $|\mathrm{HI}| > k + 1$ or $|\mathrm{HO}| > k + 1$, then (G, k) is a "no"-instance.*

We now assume that (G, k) is an instance of Pumpkin Vertex Deletion Set to which Lemma 16 does not apply; that is, $|\mathrm{HI}|, |\mathrm{HO}| \leq k+1$. We call such an instance *primary-reduced.* We now split (G, k) into $(k + 2)^2$ instances of the following problem.

Annotated Pumpkin Vertex Deletion Set	
Input:	A digraph G, an integer k, a set $S \subseteq V(G)$ with $|S| \leq 1$ and a set $T \subseteq V(G)$ with $|T| \leq 1$ and $S \cap T = \emptyset$, such that each vertex $v \in V(G) \setminus (S \cup T)$ satisfies $\max\{d^-(v), d^+(v)\} \leq k + 1$.
Question:	Is there a set $X \subseteq V(G)$ with $|X| \leq k$ for which $G - X$ is a pumpkin with source s and sink t s.t. $S \subseteq \{s\}$ and $T \subseteq \{t\}$?

The *annotation* of the problem is a source and sink (when $S \neq \emptyset$ or $T \neq \emptyset$). We call it Fully Annotated Pumpkin Vertex Deletion Set if both S and T are non-empty.

Now, for each $S \subseteq \mathrm{HO}$ with $|\mathrm{HO} \cap S| \leq 1$ and for each $T \subseteq \mathrm{HI} \setminus S$ with $|\mathrm{HI} \cap T| \leq 1$, let $W_{S,T} = (\mathrm{HI} \cup \mathrm{HO}) \setminus (S \cup T)$ and construct the instance $I_{S,T} = (G - W_{S,T}, k - |W_{S,T}|, S, T)$ of Annotated Pumpkin Vertex Deletion Set. Let $\mathcal{I}$ denote the set of all such instances; we call this the *primary split* of (G, k). Observe that $|\mathcal{I}| \leq (k+2)^2$ since $|\mathrm{HI}|, |\mathrm{HO}| \leq k+1$.

▶ **Lemma 17.** *(G, k) is a "yes"-instance if and only if at least one of the instances of* Annotated Pumpkin Vertex Deletion Set *of the primary split of (G, k) is a "yes"-instance.*

4.1 Polynomial Kernel for Annotated Pumpkin Vertex Deletion Set

We start with several reduction rules. Let (G', k', S', T') be an instance of Annotated Pumpkin Vertex Deletion Set.

▶ **Lemma 18** (Source-Sink Rule). *If G' has more than $k'+1$ vertices v with $d^-(v) = 0$ or more than $k'+1$ vertices v with $d^+(v) = 0$, then (G', k', S', T') is a "no"-instance.*

▶ **Lemma 19** (Long Path Rule). *For distinct vertices $u, v \in V(G')$, let $P = \{w_0, \ldots, w_\ell\}$ be an induced directed u,v-path of G' of length $\ell > k'+2$ so that $\{w_1, \ldots, w_{\ell-1}\} \cap (S \cup T) = \emptyset$. Obtain G'' from G' by removing w_1 and adding the arc (w_0, w_2). Then (G', k', S', T') is a "yes"-instance if and only if (G'', k', S', T') is.*

▶ **Lemma 20** (Parallel Paths Rule). *Let u, v be distinct vertices. If there are ℓ induced directed u,v-paths that do not contain a vertex of S' and $\ell > k'+2$, then let G'' be obtained from G' by removing all vertices of $\ell - (k'+2)$ arbitrary such paths. Then (G'', k', S', T') is a "yes"-instance if and only if (G', k', S', T') is.*

We now assume that (G', k', S', T') is an instance of Annotated Pumpkin Vertex Deletion Set on which Lemmas 18, 19, and 20 have no effect. We may thus assume that G has at most $k'+1$ vertices v with $d^-(v) = 0$ and that G has at most $k'+1$ vertices v with $d^+(v) = 0$. Moreover, each induced directed path has length at most $k'+2$. Finally, there are at most $k'+2$ induced directed paths between any (ordered) pair of vertices. We call such an instance *reduced*.

▶ **Lemma 21.** *Let (G', k', S', T') be a reduced instance of* Annotated Pumpkin Vertex Deletion Set*. If $|V(G')| > 2(k'+3)^2(2k'+3)^3$, then (G', k', S', T') is a "no"-instance.*

Proof. The rule can clearly be executed in linear time, so it remains to prove correctness. Suppose that (G', k', S', T') is a "yes"-instance, and let $X' \subseteq V(G')$ be any set such that $G' - X'$ is a pumpkin with source s' (where $S' \subseteq \{s'\}$) and sink t' (where $T' \subseteq \{t'\}$), and that $|X'| \leq k'$. We will say that a vertex $v \in V(G')$ has *low degree* if $\max\{d^-(v), d^+(v)\} \leq k'+1$.

We perform an iterative marking scheme. Initially, no vertices are marked. As long as this is possible, find an unmarked low-degree vertex $v \in V(G')$ such that v has at least two unmarked in-neighbors or at least two unmarked out-neighbors; then, mark v and its low-degree in- and out-neighbors.

We claim that at most $(k'+2)(2k'+3)$ vertices are marked. Since in each iteration we mark a low-degree vertex and (some of) its in- and out-neighbors, the number of vertices that are marked in an iteration is at most $2k'+3$. Hence, it suffices to bound the number of iterations. Consider any iteration and let v denote the vertex for this iteration. We count two iterations for the case that $v = s'$ or $v = t'$, and thus may assume that $v \neq s'$ and $v \neq t'$.

Without loss of generality assume that v has at least two unmarked in-neighbors. Then either $v \in X'$ or at least one of the unmarked in-neighbors of v must be in X'. Since the marking scheme proceeds iteratively, this means that after $k'+3$ iterations, we know that X' contains at least $k'+1$ vertices, a contradiction. Hence, there are at most $k'+2$ iterations, and the claim follows.

We now bound the total number of vertices v with $\max\{d^-(v), d^+(v)\} > 1$. Let v be any vertex such that $\max\{d^-(v), d^+(v)\} > 1$. Since the marking scheme was exhaustive, v has at least one marked neighbor. Since each marked vertex has low degree and there are at most $(k'+2)(2k'+3)$ marked vertices by the above claim, there can be at most $(k'+2)(2k'+3)^2$ vertices v with $\max\{d^-(v), d^+(v)\} > 1$.

To complete the proof, we notice that since the instance is reduced and thus Lemma 18 cannot be applied, there are at most $2k'+2$ vertices v with $\min\{d^-(v), d^+(v)\} = 0$. Let C denote the set of vertices v ($v \neq s', t'$) satisfying $\min\{d^-(v), d^+(v)\} = 0$ or $\max\{d^-(v), d^+(v)\} > 1$. We just proved that $|C| \leq (k'+3)(2k'+3)^2$. As any vertex $v \in V(G') \setminus (C \cup \{s', t'\})$ satisfies $d^-(v) = d^+(v) = 1$, all vertices of $V(G') \setminus (C \cup \{s', t'\})$ are on induced directed paths between vertices of $C \cup \{s', t'\}$. Since the instance is reduced and thus Lemma 19 cannot be applied, any such induced directed path has length at most $k'+2$. Moreover, because each vertex of C has low degree, a vertex of C is incident upon at most $2k'+2$ induced directed paths. Hence, there at most $|C|(2k'+2)(k'+1)$ vertices $v \in V(G') \setminus (C \cup \{s', t'\})$ in induced directed paths that have at least one vertex of C as an endpoint. Finally, there might be induced directed paths that have s' and t' as endpoints. However, since the instance is reduced and thus Lemmas 19 and 20 cannot be applied, there are at most $k'+2$ induced directed s', t'-paths, each of at most $k'+1$ internal vertices. Therefore, $|V(G')| \leq 2 + (k'+1)(k'+2) + |C| + |C|(2k'+2)(k'+1) \leq 2(k'+3)^2(2k'+3)^3$ and the lemma follows. ◀

▶ **Theorem 22.** Annotated Pumpkin Vertex Deletion Set *has a polynomial kernel with at most* $2(k'+3)^2(2k'+3)^3 = O(k'^5)$ *vertices.*

4.2 Polynomial Kernel for Pumpkin Vertex Deletion Set

Let (G, k) be an instance of Pumpkin Vertex Deletion Set. Find the primary split $\mathcal{I}$ of (G, k), and kernelize each of the resulting instances of Annotated Pumpkin Vertex Deletion Set using Theorem 22. Let $\mathcal{I}'$ denote the set of resulting instances of Annotated Pumpkin Vertex Deletion Set. Let $p(k) = 2(k+3)^2(2k+3)^3$; that is, $p(k)$ is the bound of Theorem 22.

To obtain the kernel, we need to know the source and sink of the pumpkin. Therefore, for each instance $I'_{S,T} \in \mathcal{I}'$ where $|S| = 0$ or $|T| = 0$, we create at most $(p(k))^{2-|S|-|T|}$ new instances of Fully Annotated Pumpkin Vertex Deletion Set as follows. Let $I'_{S,T} = (G', k', S, T)$. If $|S| = 0$ and $|T| = 1$, then for each $v \in V(G') \setminus T$, create a new instance $J_{v,T} = (G', k', \{v\}, T)$. We create similar instances if $|S| = 1$ and $|T| = 0$. If $|S| = 0$ and $|T| = 0$, then for each ordered pair $u, v \in V(G')$, create a new instance $J_{u,v} = (G', k', \{u\}, \{v\})$. Let $\mathcal{J}$ denote the set of these new instances and of all instances $I'_{S,T} \in \mathcal{I}'$ where $|S|, |T| > 0$. We call $\mathcal{J}$ the *secondary split* of (G, k). Observe that $|\mathcal{J}| \leq (k+2)^2(p(k))^2$, since each instance in $\mathcal{I}'$ has at most $p(k)$ vertices by Theorem 22. Moreover, each instance of $\mathcal{J}$ is indeed an instance of Fully Annotated Pumpkin Vertex Deletion Set. Similar to Lemma 17, we can now prove the following lemma.

▶ **Lemma 23.** (G, k) *is a "yes"-instance if and only if at least one of the instances of* Fully Annotated Pumpkin Vertex Deletion Set *of the secondary split of* (G, k) *is a "yes"-instance.*

Now consider the instances of the secondary split $\mathcal{J}$ of (G, k). Let (G', k', S', T') be such an instance. As argued before, $|V(G')| \leq p(k)$. Moreover, $|S'| = |T'| = 1$. We now add vertices such that the resulting graph has exactly $p(k) + 3k + 4$ vertices and that the source and sink are forced, even if we remove the annotation.

Let $(G', k', S', T') \in \mathcal{J}$. Since the instance is part of the secondary split, we know that $S' = \{s'\}$ for some $s' \in V(G')$ and that $T' = \{t'\}$ for some $t' \in V(G')$. Let A denote an arbitrary set of $\min\{k'+1, d^+(s')\}$ out-arcs of s'. For each arc $a \in A$, replace the arc by an induced directed path of length $p(k) - |V(G')| + k + k' + 3$. Then, add $k+2$ induced directed paths of length 2 from s' to t', and add $k - k'$ new vertices with an incoming arc from s'. Let G'_a denote the resulting graph. Observe that G'_a has exactly $p(k) + 3k + 4$ vertices, by construction. The resulting instance then is the instance (G'_a, k) of Pumpkin Vertex Deletion Set. Let $\mathcal{K}$ denote the set of these created instances for all instances of $\mathcal{J}$ combined. We call this the *tertiary split* of (G, k). Observe that, by construction, $|\mathcal{K}| \leq (k+1)(k+2)^2(p(k))^2$, since $k' \leq k$ by construction.

▶ **Lemma 24.** *(G, k) is a "yes"-instance if and only if at least one of the instances of* Pumpkin Vertex Deletion Set *of the tertiary split of (G, k) is a "yes"-instance.*

We can now complete the proof of Theorem 3 by simply taking a disjoint union of the instances of the tertiary split of (G, k) and setting the parameter to $k' = (|\mathcal{K}| - 1)(p(k) + 3k + 4) + k$; the full proof can be found in the full version.

5 Arc Deletion Problems

We give part of the proof of Theorem 4 by giving the polynomial-time algorithm for Out-Forest Arc Deletion Set and a sketch of the polynomial-time algorithm for Pumpkin Arc Deletion Set. The polynomial-time algorithm for Out-Tree Arc Deletion Set boils down to running breadth-first search from each vertex of the graph and is deferred to the full version.

Out-Forest Arc Deletion Set. Notice that for any out-forest T of G, the graph $(V(G), T)$ has exactly $|V(G)| - |T|$ vertices of in-degree 0, which we refer to as the *roots* of $(V(G), T)$. Therefore, if $|T| = |V(G)| - 1$, then $(V(G), T)$ is an out-tree.

Let $\mathcal{M}_1 = (E(G), \mathcal{I}_1)$ be the graphic matroid of $\langle G \rangle$, and let $\mathcal{M}_2 = (E(G), \mathcal{I}_2)$ be the partition matroid of G in which a set of arcs $I \subseteq E(G)$ is independent if and only if each vertex $v \in V(G)$ has at most one incoming arc in I. It follows that the set of out-forests of G is exactly the matroid intersection $\mathcal{M}_1 \cap \mathcal{M}_2$, that is, $\mathcal{M}_1 \cap \mathcal{M}_2 = (E(G), \mathcal{I}_1 \cap \mathcal{I}_2)$. (Notice that $\mathcal{M}_1 \cap \mathcal{M}_2$ is not generally a matroid itself.)

Using Edmond's Theorem, the matroid intersection polytope has an efficient separation oracle which consists of sequentially checking both $\mathcal{M}_1, \mathcal{M}_2$ separation oracles. Using the ellipsoid method to convert a separation oracle into an optimization algorithm allows us to construct a polynomial-time algorithm for optimization over the intersection polytope $P(\mathcal{M}_1 \cap \mathcal{M}_2)$. Linear programming duality combined with the matroid intersection theorem implies that we can find a maximum independent set of $\mathcal{M}_1 \cap \mathcal{M}_2$ in polynomial time. We refer to Schrijver's book [16, Theorem 41.1] for further explanation.

Therefore, we can find a maximum size out-forest of G in polynomial time. This proves that Out-Forest Arc Deletion Set can be solved in polynomial time.

Pumpkin Arc Deletion Set. We need the following auxiliary lemma.

▶ **Lemma 25.** *Given a digraph G and a pair of distinct vertices $s, t \in V(G)$, in polynomial time we can either find a set $Z \subseteq A(G)$ of smallest size such that the arcs of $G - Z$ induce a pumpkin with source s and sink t, or correctly answer that no $Z \subseteq A(G)$ exists such that the arcs of $G - Z$ induce a pumpkin with source s and sink t.*

Proof (Sketch). We perform the following algorithm to construct the set $Z \subseteq A(G)$; the proof of its correctness is deferred to the full version. First, add all incoming arcs of s and all outgoing arcs of t to Z, and remove these arcs from G. Now replace each vertex $v \in V(G) \setminus \{s, t\}$ by two vertices v^- and v^+, add an arc a_v from v^- to v^+, direct all incoming arcs of v towards v^-, and direct all outgoing arcs of v to start at v^+. Let G' denote the resulting graph. Let w be a cost function that assigns cost 0 to a_v for each $v \in V(G) \setminus \{s, t\}$ and cost (-1) to each other arc of G', and let c be a capacity function that assigns capacity 1 to each arc. Now find a minimum-cost s, t-flow f in G'; this takes polynomial time and yields a flow f that assigns flow 0 or 1 to each arc of G' [16, p. 177–181]. Add all arcs of $A(G)$ that are assigned flow 0 by f to Z. If the value of the flow f is 0, then answer "no". ◀

Now let (G, k) be an instance of Pumpkin Arc Deletion Set. For each pair of distinct vertices $s, t \in V(G)$, apply the algorithm of Lemma 25. Return "yes" if the size of the smallest set X found over all choices of s, t is at most k, and "no" otherwise. The correctness of the algorithm is immediate from Lemma 25. This proves that Pumpkin Arc Deletion Set can be solved in polynomial time.

6 Conclusions

In this paper, we took a different approach to generalizing Feedback Vertex Set and Tree Deletion Set to digraphs: instead of generalizing the property that the resulting graph should be 'acyclic', we generalized the property that the resulting graph should be a 'forest' and 'tree' respectively. The corresponding problems, Out-Forest Vertex Deletion Set and Out-Tree Vertex Deletion Set, were both shown to admit a polynomial kernel. We also considered Pumpkin Vertex Deletion Set, which in contrast to the previous two problems asks for the deletion to a digraph for which the underlying graph is not acyclic. We showed that Pumpkin Vertex Deletion Set admits a polynomial kernel as well.

In the past, efforts to find a polynomial kernel for Directed Feedback Vertex Set were aimed at considering restricted classes of digraphs [1, 7]. We believe that our work establishes a different line of attack that could help to resolve this longstanding open problem. In particular, all three studied problems are of the form "delete k vertices to an acyclic digraph that is a Π", where in our case Π is 'out-forest', 'out-tree', or 'pumpkin'. Therefore, we ask for which other properties Π does this problem have a polynomial kernel (parameterized by k) on general digraphs?

An interesting next step in the suggested research program would be to consider the problem to delete k vertices to obtain a *planar* acyclic digraph with a single source and a single sink. On the one hand, as evidenced by our polynomial kernel for Pumpkin Vertex Deletion Set, the restriction to a single source and a single sink can be quite helpful. On the other hand, there is no restriction on the (in-)degrees of the vertices, which neutralizes most of the reduction rules presented in this paper. Therefore, we believe that resolving Single-Source&Sink Planar Acyclic Digraph Vertex Deletion Set might yield crucial insights. (Note that without planarity condition, this problem is equivalent to Directed Feedback Vertex Set.)

Of course, this conclusion is not complete without asking the question whether Directed Feedback Vertex Set has a polynomial kernel.

References

1 F. N. Abu-Khzam. A kernelization algorithm for d-hitting set. *J. Comput. System Sci.*, 76(7):524–531, 2010. doi:10.1016/j.jcss.2009.09.002.

2 V. Bafna, P. Berman, and T. Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM J. Discrete Math.*, 12(3):289–297, 1999. doi:10.1137/S0895480196305124.

3 J. Bang-Jensen, A. Maddaloni, and S. Saurabh. Algorithms and kernels for feedback set problems in generalizations of tournaments. *Algorithmica*, to appear.

4 J. Chen, Y. Liu, S. Lu, B. O'Sullivan, and I. Razgon. A fixed-parameter algorithm for the directed feedback vertex set problem. *J. ACM*, 55(5):21:1–21:19, November 2008. doi:10.1145/1411509.1411511.

5 M. Cygan, F. Fomin, B.M.P. Jansen, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. Open problems for FPT school. Technical report, Warsaw University, 2014. URL: http://fptschool.mimuw.edu.pl/opl.pdf.

6 M. Cygan, L. Kowalik, and M. Pilipczuk. Open problems from workshop on kernels. Technical report, Warsaw University, 2013. URL: http://worker2013.mimuw.edu.pl/slides/worker-opl.pdf.

7 M. Dom, J. Guo, F. Hüffner, R. Niedermeier, and A. Truß. Fixed-parameter tractability results for feedback set problems in tournaments. *J. Discrete Algorithms*, 8(1):76–86, 2010. doi:10.1016/j.jda.2009.08.001.

8 R.G. Downey and M.R. Fellows. *Fundamentals of Parameterized Complexity Theory*. Springer, 2013.

9 G. Even, J. Naor, B. Schieber, and M. Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20(2):151–174, 1998. doi:10.1007/PL00009191.

10 M.R. Fellows, J. Guo, D. Marx, and S. Saurabh. Data reduction and problem kernels (Dagstuhl seminar 12241). Technical Report 2(6), Dagstuhl Reports, 2012.

11 J. Flum and M. Grohe. *Parameterized Complexity Theory*. Springer, 2006.

12 F.V. Fomin, S. Saurabh, and Y. Villanger. A polynomial kernel for proper interval vertex deletion. *SIAM J. Discrete Math.*, 27(4):1964–1976, 2013. doi:10.1137/12089051X.

13 A.C. Giannopoulou, D. Lokshtanov, S. Saurabh, and O. Suchy. Tree deletion set has a polynomial kernel (but no $OPT^{O(1)}$ approximation). In *Proc. FSTTCS 2014*, volume 29 of *LIPIcs*, pages 85–96, 2014.

14 R.M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972)*, pages 85–103, New York, 1972. Plenum.

15 D.G. Kirkpatrick and P. Hell. On the completeness of a generalized matching problem. In *Proc. STOC 1978*, pages 240–245, New York, 1978. ACM.

16 A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, 2003.

17 P.D. Seymour. Packing directed circuits fractionally. *Combinatorica*, 15(2):281–288, 1995. doi:10.1007/BF01200760.

18 E. Speckenmeyer. On feedback problems in digraphs. In Manfred Nagl, editor, *Proc. WG 1990*, volume 411 of *Lecture Notes Comput. Sci.*, pages 218–231. Springer, 1990. doi:10.1007/3-540-52292-1_16.

19 S. Thomassé. A $4k^2$ kernel for feedback vertex set. *ACM Trans. Algorithms*, 6(2), 2010. doi:10.1145/1721837.1721848.

20 M. Yannakakis. The effect of a connectivity requirement on the complexity of maximum subgraph problems. *J. ACM*, 26(4):618–630, 1979. doi:10.1145/322154.322157.