Preprint. To appear in Algorithmica.
DOI 10.1007/s00453-012-9695-6 – Online available.

On Making a Distinguished Vertex of Minimum Degree by Vertex Deletion

Nadja Betzler[1] · Hans L. Bodlaender · Robert Bredereck[1] · Rolf Niedermeier · Johannes Uhlmann[2]

Received: 15 July 2011 / Accepted: 24 September 2012

Abstract For directed and undirected graphs, we study how to make a distinguished vertex the unique minimum-(in)degree vertex through deletion of a minimum number of vertices. The corresponding NP-hard optimization problems are motivated by applications concerning control in elections and social network analysis. Continuing previous work for the directed case, we show that the problem is W[2]-hard when parameterized by the graph's feedback arc set number, whereas it becomes fixed-parameter tractable when combining the parameters "feedback vertex set number" and "number of vertices to delete". For the so far unstudied undirected case, we show that the problem is NP-hard and W[1]-hard when parameterized by the "number of vertices to delete". On the positive side, we show fixed-parameter tractability for several parameterizations measuring tree-likeness. In particular, we provide a dynamic programming algorithm for graphs of bounded treewidth and a vertex-linear problem kernel with respect to the parameter "feedback edge set number".

An extended abstract of this paper appeared in Proceedings of the 37th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM-2011), January 22-28, 2011, Nový Smokovec, Slovakia, volume 6543 in Lecture Notes in Computer Science, pages 123–134, Springer, 2011. Compared to the conference version, the most important change (besides providing missing details) here is that we provide a concrete tree decomposition-based dynamic programming algorithm for Min-Degree Deletion parameterized by treewidth while the conference version just claimed a classification result based on monadic second-order logic.

[1] Supported by the DFG, research project PAWS, NI 369/10.

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

Nadja Betzler, Robert Bredereck, Rolf Niedermeier, Johannes Uhlmann
Institut für Softwaretechnik und Theoretische Informatik, TU Berlin Germany
E-mail: {nadja.betzler, johannes.uhlmann}@campus.tu-berlin.de,
{robert.bredereck, rolf.niedermeier,}@tu-berlin.de

Hans L. Bodlaender
Department of Information and Computing Sciences Utrecht University, The Netherlands
E-mail: H.L.Bodlaender@uu.nl

On the contrary, we show a non-existence result concerning polynomial-size problem kernels for the combined parameter "vertex cover number and number of vertices to delete", implying corresponding non-existence results when replacing vertex cover number by treewidth or feedback vertex set number.

1 Introduction

Making a distinguished vertex of minimum degree by vertex deletion leads to natural and simple though widely unexplored graph problems. We contribute new insights into the algorithmic complexity of the corresponding computational problems, providing intractability as well as fixed-parameter tractability results.

Formally, we study the following two decision problems.

Min-Indegree Deletion (MID)
Given: A directed graph $D = (W, A)$, a distinguished vertex $w_c \in W$, and an integer $k \geq 1$.
Question: Is there a subset $W' \subseteq W \setminus \{w_c\}$ of size at most k such that w_c becomes the uniquely determined vertex that has minimum indegree in $D[W \setminus W']$?

While MID has been studied in previous work [6], its undirected counterpart seems completely unexplored.

Min-Degree Deletion (MDD)
Given: An undirected graph $G = (V, E)$, a distinguished vertex $w_c \in V$, and an integer $k \geq 1$.
Question: Is there a subset $V' \subseteq V \setminus \{w_c\}$ of size at most k such that w_c becomes the uniquely determined vertex that has minimum degree in $G[V \setminus V']$?

MID directly emerges from a problem concerning electoral control (by removing candidates) with respect to so-called "Llull voting" [6,16], one of the well-known voting systems based on pairwise comparison of candidates. As to motivate MDD, note that in undirected social networks the degree of a vertex relates to its popularity or influence [35, pages 178–180]. Then, making a distinguished vertex of minimum degree (equivalently, making it of maximum degree in the complement graph) would correspond to activities or campaigns where a single agent shall be transformed to the least or most important agent in its community. Minimum vertex deletion, hence, can be interpreted as making "competing agents" disappear at minimum cost.

A problem related to MDD is Bounded Degree Deletion (BDD) and its dual problem (considering the complement graph) Maximum k-Plex. For BDD the goal is to bound the maximum vertex degree by a prespecified value d (the case $d = 0$ is equivalent to the well-known Vertex Cover problem) using a minimum number of vertex deletions. Other than MDD, BDD and its dual Maximum k-Plex have been studied quite intensively in recent years [2,5, 18,27,33], partially motivated by their applications in social and biological network analysis.

Although both MID and MDD are simple and natural graph problems, we are aware of only one previous publication concerning these problems. MID has been shown W[2]-complete for parameter solution size k even when restricted to tournament graphs and it is polynomial-time solvable on directed acyclic graphs [6].

MID and MDD turn out to be computationally intractable in general. Both become polynomial-time solvable on acyclic graphs. Hence, it is natural to investigate in what quantitative sense their computational complexities depend on the "tree-likeness" of the input graphs. To this end, we study several distance functions (in form of parameters) measuring how close a graph is to being acyclic. Thus, we initiate a thorough theoretical analysis of MID and MDD mainly focussing on "tree-likeness parameterizations", employing several basic structural parameters measuring the tree-likeness of graphs.

Parameters and their computation. The most famous tree-likeness parameter is the *treewidth* tw of the input graph, which comes along with the concept of tree decompositions of graphs (see Subsection 3.2.1 for the definition) [9, 24]. The *feedback vertex set number* s_v of a graph is the minimum number of vertices to delete from a graph to make it acyclic. The *feedback edge set number* s_e and the *feedback arc set number* s_a, respectively, denote the minimum number of edges or arcs to delete from an undirected or directed graph to make it acyclic. For undirected graphs, it holds that $(\text{tw} - 1) \leq s_v \leq s_e$. Analogously, for directed graphs $s_v \leq s_a$. While the computation of tw, s_v and s_a leads to NP-hard problems, s_e can be quickly determined by a spanning tree computation.

Note that a small value of s_e means that the studied graph is very sparse—however, there are several sparse social networks [23,30,32], motivating parameterized complexity studies with respect to the parameter s_e.

Our contributions. Table 1 summarizes our results. We extend previous results for MID [6] by showing that MID is W[2]-hard even when parameterized by s_a whereas it turns fixed-parameter tractable for the combined parameter (k, s_v). Note that this also implies fixed-parameter tractability with respect to the combined parameter (k, s_a) since s_a is a *weaker* parameter than s_v in the sense that $s_v \leq s_a$ (refer to a recent survey [25] for a more extensive discussion on stronger and weaker parameters). As to MDD, we show that it is NP-complete as well as W[1]-hard with respect to the parameter k by devising a parameterized many-one reduction from the Independent Set problem. In addition, we show that MDD is fixed-parameter tractable for each of the tree-likeness parameters treewidth tw, size s_v^* of a feedback vertex set not containing the distinguished vertex, and feedback edge set number s_e. Herein, our fixed-parameter tractability result for tw comes with the largest combinatorial explosion. Since one can easily compute a tree decomposition of width $s_v + 1$, the algorithm can also be used for the parameter s_v. We

Table 1 Overview on the parameterized complexity of MID and MDD. The considered parameters are "treewidth tw of the input graph" (treewidth of the underlying undirected graph, respectively), "size s_v of a feedback vertex set", "size s_a of a feedback arc set", "size s_v^* of a feedback vertex set not containing w_c", "size s_e of a feedback edge set", "number k of vertices to delete", and "maximum degree" d. The number of vertices of the input graph is denoted by n. Entries marked with "†" present results from previous work [6]. The first and the last entry which use both, the MID and the MDD column, present results that hold for both problems.

parameter	MID	MDD
tw	$O((2t+4)^{2t+2} \cdot n)$, no poly kernel	
s_v	W[2]-hard	$O((2s_v+6)^{2s_v+4} \cdot n)$, no poly kernel
s_v^*	W[2]-hard	$O((2s_v^*+4)^{s_v^*} \cdot n^6)$, no poly kernel
s_a/s_e	W[2]-hard	$O(2^{s_e} \cdot n^3)$, vertex-linear kernel
k	W[2]-complete†	W[1]-hard
d	FPT†	FPT†
(k, s_v)	$O(s_v \cdot (k+1)^{s_v} \cdot n^2)$, no poly kernel	

provide a slightly improved running time bound for the parameter s_v^*. More specifically, the result relies on dynamic programming and bears a "combinatorial explosion" of $O((2s_v^*+4)^{s_v^*})$ while the dynamic programming for tw runs in $(2\,\mathrm{tw}+4)^{2\,\mathrm{tw}+2} \cdot \mathrm{poly}$. For the feedback edge set number we provide a $2s_e$-vertex problem kernel and a size-$O(2^{s_e})$ search tree. Finally, building on a recent framework for proving non-existence of polynomial-size problem kernels [7,20], we also show that there is presumably no polynomial-size problem kernel for MDD for the combined parameter (k, s_c^*), where s_c^* denotes the size of a vertex cover *not* containing the distinguished vertex. This directly implies the non-existence of polynomial-size problem kernels for the parameters feedback vertex set number and treewidth.

2 Preliminaries

Parameterized complexity is a two-dimensional framework for studying the computational complexity of problems [15,19,28]. 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* if it can be solved in $f(k) \cdot n^{O(1)}$ time, where f is a computable function only depending on k. The computational complexity class consisting of all fixed-parameter tractable problems is denoted by FPT.

A core tool in the development of fixed-parameter algorithms is polynomial-time preprocessing by *data reduction* [10,22,26]. 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' and the new parameter value k' are upper-bounded by some function only depending on k and the instance (x, k) is a yes-instance if and only if (x', k') is a yes-instance. The reduced instance, which must be computable in polynomial time, is called a *problem kernel*, and the whole process is called *reduction to a problem kernel* or *kernelization*. Usually, the kernelization is achieved by applying (several) polynomial-time data reduction rules. We call a data reduction rule *sound* if the new instance after an application of this rule is a yes-instance if and only if the original instance is a yes-instance.

Downey and Fellows [15] developed a formal framework for showing *fixed-parameter intractability* by means of *parameterized reductions*. A parameterized reduction from a parameterized problem P to another parameterized problem P' is a function that, given an instance (x, k), computes in $f(k) \cdot n^{O(1)}$ time an instance (x', k') (with k' only depending on k) such that (x, k) is a yes-instance of problem P if and only if (x', k') is a yes-instance of problem P'. The basic complexity class for fixed-parameter intractability is called $W[1]$. There is good reason to believe that $W[1]$-hard problems are not fixed-parameter tractable [15,19,28]. In this sense, $W[1]$-hardness is the parameterized complexity analog of NP-hardness. The next level of parameterized intractability is captured by the complexity class $W[2]$ with $W[1] \subseteq W[2]$.

We assume familiarity with basic graph-theoretic concepts. Let $G = (V, E)$ be an undirected graph. Unless stated otherwise, let $n := |V|$ and $m := |E|$. For $V' \subseteq V$ we denote the subgraph induced by V' as $G[V']$. Furthermore, we write $G - V'$ for $G[V \setminus V']$. Analogously, we write $G - E'$ for $(V, E \setminus E')$. The open neighborhood of a vertex v is denoted by $N_G(v)$ and the *degree* of v in G is $\deg_G(v) := |N_G(v)|$. We omit the subscript "G" if G is clear from the context. We use analogous terms for directed graphs and differentiate between in- and out-(degree, neighborhood, etc.) by a subscript in the notation (e.g., $\deg_{\text{in}}(v)$ denotes the indegree of v).

3 Min-Degree Deletion

In this section, we investigate the parameterized complexity of Min-Degree Deletion with respect to several parameters. Besides the "standard parameter" solution size k (that is, the number of vertices to delete), we focus on structural graph parameters measuring the tree-likeness. This section is organized as follows. In Subsection 3.1, we show that Min-Degree Deletion is W[1]-hard parameterized by solution size. In Subsection 3.2, we provide fixed-parameter tractability results for the "tree-likeness" parameterizations of MDD. For example, we show that MDD is fixed-parameter tractable when parameterized by the treewidth and present a linear-size problem kernel for parameter feedback edge set number. Finally, based on a plausible complexity-theoretic assumption, in Subsection 3.3 we refute the existence of polynomial-size problem kernels for all considered "tree-likeness" parameterizations except for the feedback edge set number.

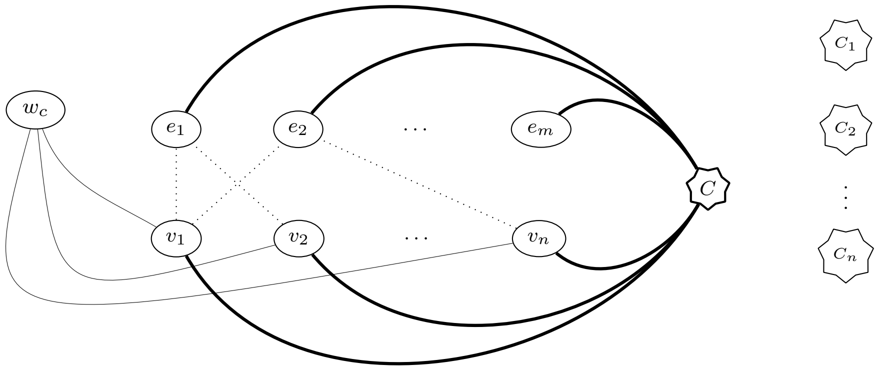

Fig. 1 Min-Degree Deletion-instance obtained from a parameterized reduction from an n-vertex Independent Set-instance. Each star represents a clique with $(n-k+2)$ vertices. Thin lines represent individual edges, bold lines represent $n-k$ edges, and dotted lines represent individual edges whose existence depends on the original instance.

3.1 Hardness of parameterization by solution size

We investigate the parameterized complexity of Min-Degree Deletion (MDD) with respect to the standard parameterization by the solution size. Note that for the directed counterpart of MDD, that is, Min-Indegree Deletion, it has been shown that the problem is W[2]-complete even when restricted to tournament graphs, providing a parameterized reduction from the W[2]-complete Dominating Set problem [6]. Here, we show that MDD is NP-complete and W[1]-hard for the parameter solution size. To this end, we devise a parameterized reduction from the W[1]-complete Independent Set problem. Given an undirected graph and an integer $k \geq 0$, Independent Set asks whether there is a size-k vertex subset $V' \subseteq V$ such that there is no edge between any two vertices from V'. The set V' is called an independent set.

Theorem 1 Min-Degree Deletion *is NP-complete and W[1]-hard for the parameter "number of vertices to delete".*

Proof We devise a parameterized reduction from the NP-complete [21] and W[1]-complete [15] Independent Set problem, yielding both the NP-hardness and W[1]-hardness of MDD (containment in NP is trivial).

Let $(G^* = (V^*, E^*), k)$ be an Independent Set instance, with $V^* = \{v_1^*, v_2^*, \ldots, v_n^*\}$ and $E^* = \{e_1^*, e_2^*, \ldots, e_m^*\}$. We construct an undirected graph G with distinguished vertex w_c such that (G, w_c, k) is a yes-instance of MDD if and only if (G^*, k) is a yes-instance of Independent Set. The reduction is illustrated in Figure 1. The vertex set of G consists of w_c and the union of the following disjoint vertex sets: $V := \{v_i \mid i \in \{1, \ldots, n\}\}$, representing the set of vertices of G^*, and $E := \{e_i \mid i \in \{1, \ldots, m\}\}$, representing the set of edges of G^*. In addition, there is a clique C with $(n-k+2)$ vertices and for each vertex $v_i \in V$, there is an isolated clique C_i with $(n-k+2)$ vertices. Moreover, there is an edge between v_i and e_j if and only if v_i^* is incident to e_j^*. Furthermore, the distinguished vertex w_c is adjacent to each vertex in V. Finally, each vertex from $V \cup E$ is adjacent to $(n-k)$ arbitrary vertices from C. This finishes the description of the construction.

The basic idea of the reduction is as follows. Observe that there are at least $(n-k+2) \cdot n > k$ vertices with degree exactly $n-k+1$. Thus, the degree of w_c in the solution graph is at most $n-k$. Hence, since only k vertex deletions are allowed and $\deg_G(w_c) = n$, a solution of size at most k for the MDD-instance contains only neighbors of w_c. Moreover, since each vertex in E has degree $n-k+2$, for each $e \in E$ at most one of its two neighbors in V can be in a solution for the MDD-instance. Thus, a size-k independent set for G^* one-to-one corresponds to a size-k solution for G and vice versa.

More formally, for the correctness we show that (G^*, k) is a yes-instance of Independent Set if and only if (G, w_c, k) is a yes-instance of MDD.
"$\Rightarrow$": Let (G^*, k) be a yes-instance of Independent Set and let $V_d^* \subseteq V^*$ be a size-k independent set of G^*. Let $M_d := \{v_i \in V \mid v_i^* \in V_d^*\}$. Clearly, $|M_d| = k$ and w_c has degree $n-k$ in $G - M_d$. Each vertex e_j with $j \in \{1, \ldots, m\}$ has degree at least $n-k+1$, because V_d^* is an independent set which means that at most one of the two neighbors of e_j in V is deleted. Since the degree of all other vertices is at least $n-k+1$, the distinguished vertex w_c is the only vertex of minimum degree in $G - M_d$.

"$\Leftarrow$": Let $(G = (V, E), w_c, k)$ be a yes-instance of MDD and let M_d denote a solution set of size at most k.

First, we show that the distinguished vertex w_c has degree $n-k$ in $G - M_d$. Assume towards a contradiction that $\deg_{G-M_d}(w_c) \neq n-k$. Observe that $\deg_{G-M_d}(w_c) \geq n-k$ since $\deg_G(w_c) = n$ and $|M_d| \leq k$. Hence, by assumption we have $\deg_{G-M_d}(w_c) > n-k$. Since the vertices of the C_i's have degree $n-k+1$, the fact that w_c is the only vertex with minimum degree in $G - M_d$ implies that $\bigcup_{1 \leq i \leq n} C_i \subseteq M_d$. Thus, $|M_d| \geq |\bigcup_{1 \leq i \leq n} C_i| > k$; a contradiction. In summary, $\deg_{G-M_d}(w_c) = n-k$ directly implying that $|M_d| = k$ and $M_d \subseteq V$.

Consider the set $V_d^* := \{v_i^* \mid v_i \in M_d\}$. Since $|M_d| = k$, it holds that $|V_d^*| = k$. Next, we argue that V_d^* is an independent set for G^*. Assume towards a contradiction that there is an edge $e_i^* = \{v_a^*, v_b^*\}$ in $G^*[V_d^*]$. Thus, both v_a and v_b are in M_d. As a consequence, the vertex e_i of G corresponding to edge e_i^* in G^* has degree $n-k$ after deleting v_a and v_b and, hence, must be deleted, too; a contradiction to the fact that $M_d \subseteq V$. Altogether, it follows that (G^*, k) is a yes-instance of Independent Set. $\square$

3.2 Fixed-parameter tractability results

In the following, all structural graph parameters are related to measuring the tree-likeness of the underlying graph. More specifically, we provide results for the treewidth tw, the size s_v^* of a feedback vertex set not containing the distinguished vertex, and the feedback edge set number s_e. By definition, $\text{tw} - 1 \leq s_v^* \leq s_e$. Hence, our fixed-parameter tractability result for MDD for the parameter tw implies fixed-parameter tractability for the parameters s_v^* and s_e. However, for each of these two parameterizations, we subsequently present specific fixed-parameter algorithms that come with improved running times.

3.2.1 Parameter treewidth

In this section, we give a linear-time algorithm for MDD when restricted to graphs of bounded treewidth showing fixed-parameter tractability with respect to the parameter treewidth. We employ a common technique for the design of algorithms on such graphs, expressing the algorithm as a form of dynamic programming on a special type of tree decompositions, called *nice tree decompositions*.

A *nice tree decomposition* of a graph $G = (V, E)$ is a pair (T, X), with $T = (I, F)$ being a rooted binary tree, and $X = \{X_i \mid i \in I\}$ being a family of subsets of V, called *bags*, such that the following holds.

- $\bigcup_{i \in I} X_i = V$.
- For each edge $\{v, w\} \in E$, there exists an $i \in I$ with $v, w \in X_i$.
- For each vertex $v \in V$, the tree nodes associated with bags that contain v, that is, $I_v = \{i \in I \mid v \in X_i\}$, form a connected subtree of T.
- If $i \in I$ is a node with two children j_1, $j_2 \in I$ in T, then $X_i = X_{j_1} = X_{j_2}$; i is called a *join node*.
- If $i \in I$ is a node with one child $j \in I$ in T, then there exists a vertex $v \in V$ with either $X_i = X_j \cup \{v\}$ (then i is called an *introduce node*) or $X_j = X_i \cup \{v\}$ (then i is called a *forget node*).
- If $i \in I$ is a leaf in T, then $|X_i| = 1$; i is called a *leaf node*.

The *width* of a nice tree decomposition is $\max_{i \in I}\{|X_i| - 1\}$. The treewidth of a graph equals the minimum width of a nice tree decomposition.

Treewidth is usually defined in terms of tree decompositions that do not need to be nice, but there always is a nice tree decomposition of optimal width. For each fixed t, there is a linear-time algorithm that decides if the treewidth of a given graph is at most t, and if so, finds a nice tree decomposition with $O(n)$ bags of width at most t: first decide if the treewidth is at most t and if so, find an arbitrary tree decomposition of width at most t with the algorithm of Bodlaender [8] and then transform this tree decomposition into a nice one with the same width, see, e.g., Kloks [24].

Suppose that we are given a graph $G = (V, E)$ and a nice tree decomposition (T, X) of G. For $i \in I$, let V_i be the union of all bags of nodes that are descendants of i, including the bag corresponding to i, and let $G_i = G[V_i]$ be the subgraph of G induced by V_i.

Given a graph $G = (V, E)$, a distinguished vertex $w_c \in V$, and a nonnegative integer $\ell \in \mathbb{N}$, we say that a set of vertices $W \subseteq V \setminus \{w_c\}$ is an ℓ-MDD set if w_c has degree at most ℓ in $G[V \setminus W]$ and all vertices in $V \setminus (W \cup \{w_c\})$ have degree at least $\ell + 1$ in $G[V \setminus W]$.

Proposition 1 *Let $G = (V, E)$ be a graph of treewidth at most t, let w_c be a distinguished vertex, and let k be a nonnegative integer. A subset $W \subseteq V \setminus \{w_c\}$ with $|W| \leq k$ fulfills the property that w_c is the unique vertex of minimum degree in $G[V \setminus W]$ if and only if W is an ℓ-MDD set for some $\ell \in \{0, 1, \ldots, t\}$.*

Proof It is a well-known fact that a graph of treewidth at most t has a vertex of degree at most t [11,31]. The proposition now directly follows. □

Theorem 2 *Given a graph and a corresponding nice tree decomposition of width at most t,* Min-Degree Deletion *can be decided in $O(n \cdot t \cdot ((t+2)^2 + 1)^{t+1})$ time.*

Proof Suppose that we are given as input to the MDD problem a graph $G = (V, E)$, a distinguished vertex $w_c \in V$, and an integer $k \geq 1$. Furthermore, suppose that we are given a nice tree decomposition $(T = (I, F), X)$ of width at most t for G. Moreover, let r be the root of the corresponding nice tree decomposition. For the root bag X_r, we assume that $X_r = \emptyset$. If this is not the case, then we add an appropriate number of forget nodes above the root obtaining a new root with empty bag.

Instead of solving MDD directly, we compute for each $\ell \in \{0, 1, \ldots, t\}$ the minimum size of an ℓ-MDD set. By Proposition 1, we then only need to check whether at least one of these sizes is at most k. Below, we show that computing the minimum size of an ℓ-MDD set for fixed ℓ can be solved in $O(n \cdot ((\ell+2)^2 + 1)^{t+1})$ time. The theorem then directly follows. From now on, we assume ℓ to be a given fixed integer between 0 and t.

Next, we introduce the used notation and define the dynamic programming tables. Then, we present the dynamic programming procedure. Consider a node $i \in I$ and a subset $V' \subseteq V_i \setminus \{w_c\}$. We say that V' is an *i-ℓ-MDD* set if both of the following conditions hold:

– $V_i \setminus V'$ contains at most ℓ neighbors of w_c and
– each vertex in $V_i \setminus (X_i \cup V' \cup \{w_c\})$ has at least $\ell + 1$ neighbors in $V_i \setminus V'$.

Note that a set is ℓ-MDD if and only if it is r-ℓ-MDD with r being the root of the tree decomposition: as $X_r = \emptyset$, we have $V_r \setminus (X_r \cup V') = V \setminus V'$.

Informally speaking, an i-ℓ-MDD set can be seen as a subset of V_i that can possibly be extended to an ℓ-MDD set for G; the distinguished vertex has degree at most ℓ and every vertex from $V_i \setminus (X_i \cup V' \cup \{w_c\})$ (all whose neighbors are contained in V_i by the definition of tree decompositions) has already its "final" degree which is at least $\ell + 1$. Note that the definition of i-ℓ-MDD sets does not restrict the degree of the vertices in X_i since the vertices in X_i can have neighbors in $V \setminus V_i$. For our dynamic programming it is decisive to know the degree of the vertices of X_i in $G[V_i \setminus V']$. This is captured by the notion of fingerprints.

The *fingerprint* of a set $V' \subseteq V_i$ with respect to i, denoted $f_i(V')$, is the pair $(f, X_i \cap V')$, with $f : X_i \setminus V' \to \{0, 1, \ldots, \ell + 1\}$ being the function such that for all $v \in X_i \setminus V'$, $f(v)$ equals the minimum of $\ell + 1$ and the degree of v in $G[V_i \setminus V']$.

We now define the value $A_i(f, Z)$ for a node $i \in I$, a subset $Z \subseteq X_i \setminus \{w_c\}$, and a function $f : X_i \setminus Z \to \{0, 1, \ldots, \ell + 1\}$: $A_i(f, Z)$ equals the minimum size of an i-ℓ-MDD set $V' \subseteq V_i \setminus \{w_c\}$ such that (f, Z) is the fingerprint with respect to i of V'. The intuition behind $A_i(f, Z)$ is the following. Without loss of generality every solution deletes an i-ℓ-MDD set. Now, $A_i(f, Z)$ gives the minimum number of vertices we must delete from V_i such that we obtain the following:

- exactly the vertices in Z are deleted from X_i and
- f gives for all vertices in X_i that are not deleted (including w_c if $w_c \in X_i$) the minimum of $(\ell + 1)$ and the number of remaining neighbors in $G[V_i]$.

For the ease of presentation, we define $A_i(f, Z) = \infty$ if there is no i-ℓ-MDD set with fingerprint (f, Z).

The main step of our algorithm is to compute for each node $i \in I$ a table with all values $A_i(f, Z)$, for all subsets $Z \subseteq X_i \setminus \{w_c\}$, and functions $f : X_i \setminus Z \to \{0, 1, \ldots, \ell + 1\}$. This will be done in the decomposition tree in bottom-up order, that is, we compute the table for a node i after the tables of the children of i have been computed. We now describe for each of the four types of nodes (leaf, introduce, forget, join) how the table is computed.

Leaf nodes. Computing the values $A_i(f, Z)$ for a leaf node i is trivial since there are at most two subsets of $V_i \setminus \{w_c\}$ as $|V_i| = 1$ (by the definition of nice tree decompositions).

Introduce nodes. Suppose that i is an introduce node with child j, where $X_i = X_j \cup \{v\}$. Notice that $V_i = V_j \cup \{v\}$.

We first initialize all values $A_i(f, Z)$ to ∞. Now, for each $Z \subseteq X_j \setminus \{w_c\}$ and for each $f : X_j \setminus Z \to \{0, 1, \ldots, \ell + 1\}$, we update some table entries for A_i, using the value of $A_j(f, Z)$; we consider what fingerprints we can get by taking a j-ℓ-MDD set with fingerprint (f, Z). We consider two cases: either v is deleted (first case) or v is not deleted (second case).

In the first case, we set $A_i(f, Z \cup \{v\})$ to $A_j(f, Z) + 1$. Namely, each set $V' \subseteq V_i$ with fingerprint $(f, Z \cup \{v\})$ is an i-ℓ-MDD set if and only if $V' \setminus \{v\}$ is a j-ℓ-MDD set; the fingerprint of $V' \setminus \{v\}$ with respect to j equals $(f, Z \cup \{v\})$.

In the second case, we generate a new function g, and possibly update a table entry $A_i(g, Z)$. For $w \in X_j \setminus Z$, let $g(w) := f(w)$ if $\{v, w\} \notin E$, and let $g(w) := \min\{\ell + 1, f(w) + 1\}$ if $\{v, w\} \in E$. We now set $A_i(g, Z)$

to the minimum of its current value and $A_j(f, Z)$, unless $w_c \in X_i \setminus Z$ and $g(w_c) = \ell + 1$, where we do nothing.

The correctness of the second case follows from the following observation. Let V' be a j-ℓ-MDD set with fingerprint (f, Z). Observe that V' is also an i-ℓ-MDD set, unless the degree of w_c is too high. The latter can only occur when $\{v, w_c\} \in E$ and $w_c \in V_i$; thus $w_c \in X_i$ (and we always have $w_c \notin Z$). Hence, in this case we check whether the degree of w_c, which equals $g(w_c)$, is not too high. Finally, one can verify that $g(w)$ indeed gives the minimum of $\ell+1$ and the degree of the vertex w in $G[V_i \setminus V']$. Thus, (g, Z) is the fingerprint of V' with respect to i.

Forget nodes. Suppose that i is a forget node with child j with $X_i = X_j \setminus \{v\}$. Notice that $V_i = V_j$.

Again, we first initialize all values $A_i(f, Z)$ to ∞. Now, for each $Z \subseteq X_i \setminus \{w_c\}$ and $f : X_j \setminus Z \to \{0, 1, \dots, \ell + 1\}$, we possibly update one table entry in A_i.

If $w_c \neq v$, $v \notin Z$, and $f(v) \neq \ell + 1$, then we just discard the entry. For each j-ℓ-MDD set V' that corresponds to (f, Z), v has at most ℓ neighbors in $V_i \setminus V'$, and v has no neighbors in $V \setminus V_i$; the latter is true by the properties of tree decompositions. So, this entry does not lead to i-ℓ-MDD sets.

In all other cases, a j-ℓ-MDD set with fingerprint (f, Z) is also an i-ℓ-MDD set. Let g be the restriction of f to $X_i \setminus Z$. Then such a set has fingerprint $(g, Z \cap X_i)$ with respect to i. So we set $A_i(g, Z \cap X_i)$ to the minimum of its current value and $A_j(f, Z)$.

Join nodes. We now look at the case that i is a join node. Suppose j_1 and j_2 are the children of i. Note that $V_i = V_{j_1} \cup V_{j_2}$ and $X_i = X_{j_1} = X_{j_2} = V_{j_1} \cap V_{j_2}$.

For each i-ℓ-MDD set $W \subseteq V_i \setminus \{w_c\}$, $W \cap V_{j_1}$ is a j_1-ℓ-MDD set and $W \cap V_{j_2}$ is a j_2-ℓ-MDD set. Our procedure thus looks at all pairs of fingerprints of j_1-ℓ-MDD sets and of j_2-ℓ-MDD sets that agree on which vertices in X_i belong to the set, and sees if they can be combined.

As in earlier steps, we first initialize all values $A_i(f, Z)$ to ∞. Now, for each $Z \subseteq X_i \setminus \{w_c\}$, each $f_1 : X_i \setminus Z \to \{0, 1, \dots, \ell + 1\}$, and each $f_2 : X_i \setminus Z \to \{0, 1, \dots, \ell + 1\}$, we do the following.

We first compute a function $g : X_i \setminus Z \to \{0, 1, \dots, \ell + 1\}$ by setting, for each $v \in X_i \setminus Z$, $g(v)$ to $f_1(v) + f_2(v) - |\{w \in X_i \setminus Z \mid \{v, w\} \in E\}|$. Then set $A_i(g, Z)$ to the minimum of its current value and $A_{j_1}(f_1, Z) + A_{j_2}(f_2, Z) - |Z|$.

We now explain why this step is correct. Suppose that W_1 is a j_1-ℓ-MDD set with fingerprint (f_1, Z) and W_2 is a j_2-ℓ-MDD set with fingerprint (f_2, Z). Then $v \in X_i \setminus Z$ has degree $g(v)$ in $G[V_i \setminus (W_1 \cup W_2)]$: we add its number of neighbors in $G[V_{j_1} \setminus W_1]$ (which is $f_1(v)$) to its number of neighbors in $G[V_{j_2} \setminus W_2]$ (which is $f_2(v)$), and subtract the number of edges that we counted twice (that is, $|\{w \in X_i \setminus Z \mid \{v, w\} \in E\}|$.) So, $W_1 \cup W_2$ has fingerprint (g, Z).

If the size of such W_1 is $A_{j_1}(f_1, Z)$ and the size of such W_2 is $A_{j_2}(f_2, Z)$, then the size of $W_1 \cup W_2$ equals $A_{j_1}(f_1, Z) + A_{j_2}(f_2, Z) - |Z|$ since $W_1 \cap W_2 = Z$. Thus, we set the table entry for $A_i(g, Z)$ correctly.

As all pairs are considered, the table entries for A_i have their correct values at the end of the step for a join node.

Final step. In a bottom-up order we compute for all nodes i in the nice tree decomposition a table with values A_i, using the methods described above. The last of these steps computes the table for the root r of the tree. As $X_r = \emptyset$, all r-ℓ-MDD sets have the same fingerprint $(\Psi, \emptyset)$, with Ψ the function with empty domain. As each ℓ-MDD set is an r-ℓ-MDD set with fingerprint $(\Psi, \emptyset)$, the minimum size of an ℓ-MDD set is given by the value $A_r(\Psi, \emptyset)$; we just test whether this value is at most k.

Running time and space analysis. We first analyze the size of the dynamic programming table. To this end, we consider the number of entries of A_i for a node $i \in I$ that has maximum bag size $t+1$. For each of the 2^{t+1} subsets of X_i, there are at most $(l+2)^{t+1}$ fingerprints. Since $l \leq t$ it follows that A_i contains at most $2^{t+1} \cdot (t+2)^{t+1} = (2t+4)^{t+1}$ entries. Regarding the running time, it is easy to observe that leaf, forget, and introduce nodes can be handled in time linear in the number of entries of the corresponding table. For the join nodes one needs to compare all pairs of entries of the two children. This leads to an overall running time bound of $O((2t+4)^{2t+2} \cdot n)$ since the nice tree decomposition has $O(n)$ nodes. □

Corollary 1 *For each fixed t, there is a linear-time algorithm for* Min-Degree Deletion *on graphs of treewidth at most t.*

3.2.2 Parameter distinguished feedback vertex set number

Next, we investigate the parameter *distinguished feedback vertex set number* s_v^* denoting the "size of a feedback vertex set not containing the distinguished vertex w_c". Since for a graph with treewidth tw it holds that $s_v^* \geq \mathrm{tw} - 1$, Theorem 2 implies that MDD is fixed-parameter tractable with respect to s_v^* giving an upper bound of $(2s_v^* + 6)^{2s_v^* + 4}$ for the exponential part of the running time. In the following, we improve this bound by providing an algorithm specifically designed for the parameter s_v^* with running time $O((2s_v^* + 4)^{s_v^*} \cdot n^4 \cdot \deg(w_c)^2)$. There are several efficient approximation [1,3,4] and fixed-parameter algorithms [13,34] for computing small feedback vertex sets whereas this task seems harder in case of treewidth.

Let $(G = (V, E), w_c, k)$ be an MDD-instance and let V_f be a feedback vertex set that does not contain w_c. Our algorithm basically branches into all possible subsets V_f^* of V_f and investigates whether there is a solution containing all vertices from V_f^* and not containing any vertex from $V_f \setminus V_f^*$. Furthermore, the algorithm iterates over the "final" degree that w_c might assume in the graph G after deleting a set of "solution vertices". After applying some simple branching and preprocessing steps, it will remain to solve the following problem.

Algorithm 1 **`MDD-solv`**. The input consists of an MDD-instance (G, w_c, k) and a feedback vertex set V_f. `MDD-solv` returns "yes" iff (G, w_c, k) is a yes-instance.

1: **for** each $V_f^* \subseteq V_f$ with $|V_f^*| \leq k$ **do** ▷ solution vertices from V_f
2: remove V_f^* from G and set $k := k - |V_f^*|$
3: **for** $i := \max\{|N(w_c) \cap V_f|, \deg(w_c) - k\}$ to $\deg(w_c)$ **do** ▷ fix final degree of w_c
4: **while** there is a vertex $v \neq w_c$ with $\deg(v) \leq i$ **do** ▷ simple data reduction
5: **if** $v \in V_f$ **then goto** to line 1
6: **else** remove v from G and set $k := k - 1$
7: **if** $k < 0$ **then goto** line 3
8: **if** `MDD-annotated`(G, w_c, $V_f' := V_f \setminus V_f^*$, k, i) **then**
9: **return** "yes"
10: **return** "no"

Annotated Min-Degree Deletion (AMDD)
Given: An undirected graph $G = (V, E)$, a distinguished vertex w_c, a feedback vertex set V_f of G with $V_f \subseteq V \setminus \{w_c\}$, and two non-negative integers k and i.
Question: Is there a subset $M \subseteq V \setminus (V_f \cup \{w_c\})$ of size at most k such that, in $G - M$, $\deg(w_c) = i$ and every other vertex has degree at least $i + 1$?

Branching and preprocessing steps. Let $(G = (V, E), w_c, k)$ be an MDD-instance and let V_f be a feedback vertex set that does not contain w_c. The overall structure of our algorithm `MDD-solv` is provided by Algorithm 1. Basically, the algorithm calls a subroutine solving an annotated version of MDD after applying the following branching and preprocessing steps. In line 1 of `MDD-solv`, one branches over all subsets of V_f to be part of the solution and in line 2 the corresponding vertices are deleted and the parameter is decreased accordingly. In lines 3-9, one tries all possibilities to fix the final degree of w_c to be i and iteratively adds all vertices with degree at most i to the solution. It remains to solve the AMDD-instance $(G, w_c, V_f' := V_f \setminus V_f^*, k, i)$. It is easy to verify that `MDD-solv` takes $O(2^{|V_f|} \cdot n^2 \cdot t_{\text{MD-ann}})$ time, where $t_{\text{MD-ann}}$ denotes the running time of `MDD-annotated`$(G, w_c, V_f' := V_f \setminus V_f^*, k, i)$.

Due to the preprocessing, in the following we can assume that w_c has at most i neighbors in V_f and every other vertex has degree at least $i + 1$. Now, for an AMDD-instance $(G = (V, E), w_c, V_f, k, i)$, the algorithm makes use of the following property of $V_S := V \setminus (V_f \cup \{w_c\})$, the set consisting of all vertices that can be part of the solution.

Observation 1 *Let $n_1, \ldots, n_d$ denote the neighbors of w_c in $G - V_f$. Then, in the graph $G[V_S]$, every vertex n_x, $1 \leq x \leq d$, belongs to a connected component $T(x)$ such that $T(x)$ is a tree not containing any vertex n_y with $n_x \neq n_y$.*

Observation 1 can be seen as follows. Consider two neighbors n_x and n_y of w_c. First, assume that there would be a path from n_x to n_y that does not contain w_c. Adding w_c to this path would induce a cycle and hence V_f would not be a feedback vertex set of G. Hence, every connected component can contain at most one neighbor of w_c. Second, a cycle within a connected component would also violate that V_f is a feedback vertex set. Hence, all connected components induce trees.[1]

Now, we take a look at an arbitrary solution set M of our MDD-instance. Since the final degree of w_c is i, M must contain $\deg_G(w_c) - i$ neighbors of w_c. Putting a vertex $x \in N(w_c) \setminus V_f$ into the solution may decrease the degree of other vertices from $T(x)$ so that they also must be part of the solution. The set $A(x)$ of *affected* vertices that need to be deleted when x is deleted can be computed iteratively as follows. Start with $A(x) := \{x\}$. While there is vertex v with degree at most i in $T(x) - A(x)$, add v to $A(x)$. Since we have to put all vertices of $A(x)$ into a solution when choosing x into the solution, we define the *cost* of x as $\text{cost}(x) := |A(x)|$. Note that there might be a large number of vertices that are not affected by any x; for example, a vertex may not be a neighbor of w_c but may have many neighbors in V_f. However, such a vertex clearly can never be in a minimal solution. Formally, this leads to the following observation.

Observation 2 *Let $M \subseteq V \setminus (V_f \cup \{w_c\})$ be a minimal solution, that is, there is no solution $M' \subset M$ and, in $G - M$,* $\deg(w_c) = i$ *and* $\forall_{x \in V \setminus \{w_c\}}$: $\deg(x) \geq i + 1$. *Then,*

$$M \setminus \left(\bigcup_{x \in N(w_c) \setminus V_f} A(x) \right) = \emptyset.$$

For the graph not containing vertices from the feedback vertex set V_f, a solution could easily be computed by choosing a set of $\deg(w_c) - i$ neighbors of w_c such that the sum of the corresponding costs is minimal. The decisive point is that putting a vertex x into the solution set may also decrease the degree of vertices from V_f. By definition, we cannot remove any vertex from V_f. Thus, we must ensure that we "keep" enough vertices from V_S such that the final degree of every vertex from V_f is at least $i + 1$. For every vertex $v \in V_f$, we can easily compute the number $n_{\text{fixed}}(v)$ of neighbors which v has "for sure" in every minimal solution. More specifically, $n_{\text{fixed}}(v)$ is the number of v's neighbors in $V_f \cup V_S \setminus (\bigcup_{x \in N(w_c)} A(x))$ (see Observation 2).

We introduce some notation measuring how many neighbors of a vertex from V_f must be kept under the assumption that a certain subset $V_r \subseteq V_s$ is *not* part of a solution. More specifically, for a vertex $v \in V_f$, let $n_{V_r}(v)$ be the number of neighbors of v in V_r. Then, the number of additional neighbors that are not allowed to be deleted is defined as $s(v, V_r) := i + 1 - n_{\text{fixed}}(v) - n_{V_r}(v)$. This can be generalized as follows.

[1] Observation 1 does not hold for a feedback vertex set containing the distinguished vertex. Hence, the following approach does not transfer to this more general case.

Algorithm 2 **`MDD-annotated`** returns "yes" iff there is solution set for given AMDD-instance (G, w_c, k, V_f, i). Remain(S', n_x) denotes the remain-tuple obtained from S' by additionally fixing n_x to be not contained in the solution set.

for $z = 1$ to $\deg(w_c)$ **do** ▷ Initialization
  for each $S' \in \mathcal{S}$ **do**
    $D(0, z, S') = +\infty$
for each $S' = (s'_1, s'_2, \ldots, s'_{|V_f|}) \in \mathcal{S}$ **do**
  if $s'_j < n_{\text{fixed}}(v_j)$ for any $j \in \{1, \ldots, |V_f|\}$ **then**
    $D(0, 0, S') = +\infty$
  else
    $D(0, 0, S') = 0$
for $x = 1, \ldots, |N(w_c)|$ **do** ▷ Table update
  for $z = 1, \ldots, x$ **do**
    for each $S' \in \mathcal{S}$ **do**
      minCostRemove := $D(x-1, z-1, S') + \text{cost}(n_x)$
      minCostNotRemove := $D(x-1, z, \text{Remain}(S', n_x))$
      $D(x, z, S')$:= min(minCostRemove, minCostNotRemove)
if $D(\deg(w_c), \deg(w_c) - i, (0, \ldots, 0)) \leq k$ **then return** "yes"
else return "no"

Definition 1 For $V_f = \{v_1, \ldots, v_{|V_f|}\}$, the *remain-tuple* with respect to $V_r \subseteq V_S$ is $S = (s_1, \ldots, s_{|V_f|})$ where $s_j := s(v_j, V_r)$, $1 \leq j \leq |V_f|$.

Recall that the task is to search for a set $N \subseteq N(w_c) \setminus V_f$ of $\deg(w_c) - i$ neighbors of w_c with minimum cost such that the degree of every vertex from V_f is at least $i+1$. Now, let $N' \subseteq N(w_c) \setminus V_f$ be a subset of w_c-neighbors that are fixed to be *not* part of a solution. The effect of not deleting N' to decide which of the other w_c-neighbors may be removed can be described by a remain-tuple. More specifically, a subset $N' \subseteq N(w_c) \setminus V_f$ *realizes* a remain-tuple $(s'_1, \ldots, s'_{|V_f|})$ when, for every $v \in V_f$, the number of neighbors of v in $\bigcup_{x \in N'} A(x)$ is at least $i + 1 - n_{\text{fixed}}(v) - s'_i$. Then, a cost-$k$ set $N \subseteq N(w_c)$ containing $\deg(w_c) - i$ neighbors of w_c such that set $N(w_c) \setminus N$ realizes the remain-tuple $(0, \ldots, 0)$ corresponds to a solution.

Dynamic programming table. Based on the previous definitions, the dynamic programming table D has entries $D(x, z, S')$ with

- $x \in \{1, \ldots, d\}$ where $d := |N(w_c) \cap V_s|$,
- $z \leq \min\{x, d - i\}$, and
- $S' \subseteq \mathcal{S} := \{(s'_1, \ldots, s'_{|V_f|}) \mid 0 \leq s'_j \leq i + 1, 1 \leq j \leq |V_f|\}$.

The entry $D(x, z, S')$ contains the minimum cost of deleting a size-z subset $N' \subseteq \{n_i \in N(w_c) \mid i \leq x\}$ such that $N'_r := N(w_c) \setminus N'$ "realizes" the remain-tuple S'. It follows that $D(\deg(w_c), \deg(w_c) - i, (0, \ldots, 0)) \leq k$ if and only if (G, V_f, w_c, k, i) is a yes-instance of AMDD. It follows from Proposition 1 that the size of table D is upper-bounded by $\deg(w_c)^2 \cdot (s^*_v + 2)^{s^*_v}$.

Dynamic programming algorithm. Consider Algorithm 2. Note that for every $v \in V_f$ the values of $n_{\text{fixed}}(v)$ can be precomputed in quadratic time. In the initialization phase, the cost of "removing at least one vertex from an empty set" is set to infinity (lines 1-3). Furthermore, without fixing any neighbor to be "not contained in the solution set", it is not possible to realize a remain-tuple $S' := (s'_1, \ldots, s'_{|V_f|})$ with $s'_j < i + 1 - n_{\text{fixed}}(v_j)$ for any j. Hence, initializing these table values with infinity is correct (lines 4-6). All entries with a remain-tuple $S' := (s'_1, \ldots, s'_{|V_f|})$ with $s'_j \geq i + 1 - n_{\text{fixed}}(v_j)$ for all j are always realized with zero costs (lines 4-8). Now, consider the update step. It is easy to verify that the three loops (lines 9-11) ensure that every value that is used on the righthandside (lines 12-14) has been computed before.

It remains to prove the correctness of this computation. Consider a table entry $D(x, z, S')$ containing the minimum costs for removing z vertices from N'. For the current neighbor n_x, either it is part of the solution or not. If n_x is removed, then the minimum costs are exactly the minimum costs for removing $z - 1$ vertices from $N' \subseteq \{n_i \in N(w_c) \mid i \leq x - 1\}$ plus $\text{cost}(n_x)$ (line 12). Otherwise, the costs are exactly the minimum costs for removing z vertices from $N' \subseteq \{n_i \in N(w_c) \mid i \leq x - 1\}$, while realizing the remain-tuple $S' = (s'_1, \ldots, s'_{|V_f|})$ under the condition that n_x is fixed to be not part of the solution. This is ensured by the Remain-function (line 13) which can be formally defined as follows.

$$\text{Remain}(S', n_x) := (s''_1, \ldots, s''_{|V_f|}) \text{ with } s''_j := \max\{0, s'_j - |N_{A(n_x)}(v_j)|\},$$

where $N_{A(n_x)}(v_j)$ denotes the neighbors of v_j in $A(n_x)$.

Finally, we analyze the running time and table size: Each of the first two dimensions of D is bounded from above by $\deg(w_c)$. The remain-tuple is defined such that each of the $|V_f|$ entries has an integer value between 0 and $|V_f| + 1$ (see Definition 1). Hence, $|\mathcal{S}| = (|V_f| + 2)^{|V_f|}$. Clearly, the remaining steps can be accomplished in $O(n^2)$ time. Hence, together with the running time for the overall branching into all subsets of a feedback vertex set, one ends up with the following.

Theorem 3 MIN-DEGREE DELETION *can be decided in* $O((2s^*_v + 4)^{s^*_v} \cdot n^4 \cdot \deg(w_c)^2)$ *time with* s^*_v *being the size of a feedback vertex set not containing* w_c.

3.2.3 Parameter feedback edge set number

As discussed in the beginning of this section, the feedback edge set number is the weakest of our parameters measuring the tree-likeness of graphs. Hence, not surprisingly, we achieve our best running time bounds here, based on kernelization and simple structural observations.

Our problem kernel result relies on the following "low-degree removal" procedure. Let $G = (V, E)$ be an undirected graph and let k be a positive integer. Denote by $\text{RLD}(G, k)$ (for "remove low degree") the graph resulting from the following data reduction:

> If deleting all or all but one neighbors of w_c (and iteratively all further vertices with degree at most zero or one, respectively) leads to a solution of size at most k, then return "yes". Otherwise, iteratively remove every vertex with degree at most two from G and decrease k accordingly and return the resulting graph.

If deleting all or all but one neighbors of w_c does not lead to a solution, then w_c has degree at least two for every solution. Hence, it is easy to verify that RLD(G, k) is sound and can be executed in $O(n^2 \cdot k)$ time. Note that every vertex different from w_c in RLD(G, k) has degree at least three.

Theorem 4 Min-Degree Deletion *admits a* $2s_e$*-vertex problem kernel which can be computed in* $O(n^2 \cdot k)$ *time, where* s_e *denotes the feedback edge set number.*

Proof Let $(G' = (V', E'), k') :=$ RLD(G, k) and let E_d be a size-s_e feedback edge set for G'. In the following, we argue that $|V'| \leq 2s_e$. The graph $G' - E_d$ is a forest. Let l denote the number of leaves in $G' - E_d$. Since each vertex in G' has degree at least three, each leaf in $G' - E_d$ is incident to at least two edges in E_d. Thus, since each edge of E_d has at most two leaf endpoints, it follows that $l \leq s_e$. Analogously, each degree-two vertex in $G' - E_d$ is incident to at least one edge in E_d. Since there are l leaves in $G' - E_d$ and there are at most $2s_e$ vertices that are incident to an edge of E_d, $G' - E_d$ contains at most $2s_e - 2l$ inner vertices with degree two. Moreover, all remaining vertices must have degree at least three and a tree with l leaves can clearly have at most l vertices of degree at least three. Altogether, G' consists of at most $l + 2s_e - 2l + l = 2s_e$ vertices. □

A simple brute-force strategy for solving Min-Degree Deletion is to branch into all possible cases of deleting a subset of the neighbors of the distinguished vertex and then to iteratively delete all vertices with degree at most the new degree of the distinguished vertex. We show that this strategy leads to an algorithm with exponential running time factor 2^{s_e} since in reduced instances the degree of the distinguished vertex is bounded by s_e.

Lemma 1 *Let* $G = (V, E)$ *and let* w_c *denote a distinguished vertex of* G*. If* $\deg_G(v) \geq 3$ *for every* $v \in V \setminus \{w_c\}$*, then* $\deg_G(w_c) \leq s_e$*, where* s_e *denotes the feedback edge set number of* G*.*

Proof First, we argue that there is a minimum-cardinality feedback edge set that does not contain any edge incident to w_c. To this end, consider a spanning tree that results from a breadth-first search of G starting at w_c. Clearly, such a tree contains all neighbors of w_c and, hence, the edges that are not contained in such a spanning tree form a feedback edge set not containing any edge incident to w_c. With this observation the correctness of the lemma is easy to verify. Let E_d denote such a feedback edge set. With the same arguments as in the proof of Theorem 4, it follows that there are at most s_e leaves in $G - E_d$. Clearly, in trees the degree of each vertex is bounded from above by the number of leaves. Since $\deg_{G-E_d}(w_c) = \deg_G(w_c)$ the degree of w_c is at most s_e. □

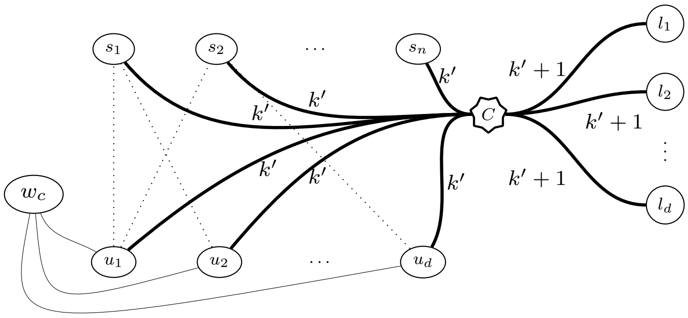

Fig. 2 MDD–instance obtained from a polynomial time and parameter transformation from a HS–instance $(U, \mathcal{S}, k')$ with $|U| = d$. The star C represents a clique on $k' + 1$ vertices. A bold line labeled by a weight j represents j edges. (Lines from star C indicate edges from arbitrarily vertices of the clique.) Moreover, edges between V_U and $V_{\mathcal{S}}$ are represented by dotted lines.

Theorem 5 MIN-DEGREE DELETION *can be decided in* $O(2^{s_e} \cdot {s_e}^3 + n^2 \cdot k)$ *time, where* s_e *is the feedback edge set number.*

Proof Let (G, w_c, k) denote an instance of MIN-DEGREE DELETION. To solve MIN-DEGREE DELETION proceed as follows. First call $\text{RLD}(G, k)$. If $\text{RLD}(G, k)$ does not return "yes", then let $(G' = (V', E'), k') := \text{RLD}(G, k)$. Next, systematically enumerate all subsets of neighbors of w_c. For each subset $V'' \subseteq N_{G'}(w_c)$ check whether deleting all vertices of V'' from G' and then iteratively deleting all vertices whose degree does not exceed the new degree of w_c leads to a solution of size at most k' for G'. The correctness of this solving strategy is obvious.

Next, we analyze the running time of this strategy. Since each vertex of G' has degree at least three, Lemma 1 implies that $\deg_{G'}(w_c) \leq s_e$. Hence, there are at most 2^{s_e} subsets of neighbors of w_c. Clearly, for each subset all steps can be applied in $O(s_e^3)$ time, leading to an overall running time bound of $O(2^{s_e} \cdot {s_e}^3 + n^2 \cdot k)$. □

3.3 Some non-existence results regarding polynomial-size problem kernels

We show that, unless coNP ⊆ NP / poly, there is no polynomial-size problem kernel for MDD with respect to the parameter $s_c^* :=$"size of a vertex cover that does not contain w_c". Indeed, we prove the even stronger result that it is unlikely that there is a polynomial-size problem kernel for the combined parameter (s_c^*, k), where k denotes the solution size. Since the treewidth t_w, the feedback vertex set number s_v, and the distinguished feedback vertex set number s_v^* of a graph are bounded from above by s_c^*, this non-kernelization result carries over to all these parameterizations.

Theorem 6 MIN-DEGREE DELETION *does not admit a polynomial kernel with respect to the combined parameter* (s_c^*, k), *with* s_c^* *being the size of a vertex cover not containing* w_c *and* k *being the solution size, unless* $\mathrm{coNP} \subseteq \mathrm{NP} / \mathrm{poly}$.

Proof Our proof relies on a reduction from HITTING SET (HS) defined as follows. Given a set family $\mathcal{S} := \{S_1^*, \dots, S_m^*\}$ over a universe $U := \{u_1^*, \dots, u_d^*\}$ and an integer $k' \geq 0$, HS asks for a subset $U' \subseteq U$ with $|U'| \leq k'$ such that $S_i^* \cap U' \neq \emptyset$ for every i, $1 \leq i \leq m$. Herein, U' is called a *hitting set*. The reduction is illustrated by Figure 2.

Dom et al. [14] have shown that HS does not admit a problem kernel of size $(d + k')^{O(1)}$, unless $\mathrm{coNP} \subseteq \mathrm{NP} / \mathrm{poly}$. Since HS and MDD are NP-complete, it directly follows from a result of Bodlaender et al. [12] that if there is a polynomial-time reduction from HS to MDD such that $(s_c^* + k) \leq (d + k')^{O(1)}$, then MDD does not admit a polynomial kernel with respect to (s_c^*, k) unless $\mathrm{coNP} \subseteq \mathrm{NP} / \mathrm{poly}$. In the following, we provide such a reduction (which is referred to as polynomial time and parameter transformation in the literature [12]).

Let $(\mathcal{S}, U, k')$ be an HS-instance. We construct an undirected graph $G = (V, E)$ with a distinguished vertex w_c as follows. The vertex set V is the disjoint union of the sets $\{w_c\}$, V_U, $V_{\mathcal{S}}$, C, and L. Herein, $V_U := \{u_i \mid u_i^* \in U\}$, $V_{\mathcal{S}} := \{s_j \mid S_j^* \in \mathcal{S}\}$, $C := \{c_1, \dots, c_{k'+1}\}$, and $L := \{l_1, \dots, l_d\}$. There is an edge between u_i and s_j if and only if $u_i^* \in S_j^*$. Moreover, the following edges are added. First, w_c is made adjacent to every vertex in V_U. Furthermore, C is transformed into a clique, and each l_i, $1 \leq i \leq d$, is made adjacent to each vertex in C. Finally, each vertex $x \in V_U \cup V_{\mathcal{S}}$ is made adjacent to k' arbitrarily chosen vertices of C. This completes the construction. Note that the degree of w_c is d and, for each other vertex, at least $k' + 1$.

Now, observe that each edge of G is incident to a vertex in $C \cup V_U$. Hence, G has a vertex cover of size $k' + 1 + d$ which does not contain w_c. For the correctness of the reduction it remains to show that $(\mathcal{S}, U, k')$ is a yes–instance of HS if and only if $(G, w_c, d - k')$ is a yes–instance of MDD.

"$\Rightarrow$": Let $U' \subseteq U$ with $|U'| = k'$ denote a hitting set of $\mathcal{S}$. We show that $M := \{u_j \mid u_j^* \in U \setminus U'\}$ is a solution for $(G, w_c, d - k')$. First, observe that w_c has degree k' in $G - M$. Moreover, since U' is a hitting set, every vertex in $V_{\mathcal{S}}$ has at least one neighbor in $V_U \setminus M$, and, hence, degree at least $k' + 1$ in $G - M$. For this reason and since we do not delete vertices from L or neighbors of vertices from L, each vertex in $V \setminus \{w_c\}$ has degree at least $k' + 1$. Hence, $(G, w_c, d - k')$ is a yes-instance of MDD.

"$\Leftarrow$": Let $M \subseteq V$ with $|M| \leq d - k'$ denote a solution for $(G, w_c, d - k')$. First, we argue that w_c has degree k' in $G - M$. Clearly, w_c cannot have degree smaller than k'. Furthermore, w_c cannot have degree greater than k' in $G - M$; otherwise, since w_c is the only vertex with minimum degree in $G - M$ and each vertex in L has degree $k' + 1$, M must contain every vertex in L. However, $|L| = d > d - k'$. Thus, $\deg_{G-M}(w_c) = k'$ and, as a consequence, $M \subseteq V_U$ and $|M| = d - k'$.

Next, we show that $U' := \{u_i^* \in U \mid u_i \in V_U \setminus M\}$ is a hitting set of size k'. By the observation above, $|U'| = k'$. Assume towards a contradiction that there is a set S_j^*, $1 \leq j \leq m$, with $S_j^* \cap U' = \emptyset$. Thus, for each element $u_i^* \in S_j^*$ the corresponding vertex u_i is in M. Due to the construction of G, vertex s_j has degree k' in $G - M$; since $\deg_{G-M}(w_c) = k'$ this contradicts the fact that w_c is the only vertex with minimum degree. □

Since treewidth, distinguished feedback vertex set size, and feedback vertex set size of a graph are bounded from above by s_c^*, we arrive at the following.

Corollary 2 Min-Degree Deletion *has no polynomial problem kernel with respect to any of the parameters feedback vertex set, distinguished feedback vertex set, or treewidth, respectively, unless* coNP ⊆ NP / poly.

4 Min-Indegree Deletion

Complementing previous work [6], we show that MID is W[2]-hard with respect to the parameter feedback arc set number s_a. This also implies W[2]-hardness with respect to the parameter feedback vertex set number s_v since $s_a \geq s_v$. We contrast these negative results by showing fixed-parameter tractability with respect to the combined parameter s_v and number k of vertices to delete.

To show W[2]-hardness with respect to s_a, we provide a parameterized reduction from the W[2]-complete Dominating Set (DS) problem [15]. Given an undirected graph $G = (V, E)$ and an integer k, DS asks whether there is a size-k vertex subset $V' \subseteq V$ such that every vertex from $V \setminus V'$ has a neighbor in V'. Such a subset V' is called *dominating set*. Thus, vertex u dominates vertex v if and only if $u = v$ or $\{u, v\} \in E$.

Theorem 7 Min-Indegree Deletion *is W[2]-hard with respect to the feedback arc set number s_a.*

Proof Given a DS-instance $(G^* = (V^*, E^*), k)$ with $V^* = \{v_1^*, v_2^*, \ldots, v_n^*\}$, we construct a directed graph $G = (W, E)$ with feedback arc set number at most $(k+1)^2$ such that $(G, w_c, n - k)$ is a yes-instance of MID if and only if (G^*, k) is a yes-instance of DS. The construction is illustrated in Figure 3.

The vertex set W of G consists of w_c and the union of the following disjoint vertex sets. The sets $V := \{v_i \mid v_i^* \in V^*\}$ and $D := \{d_i \mid v_i^* \in V^*\}$, where d_i represents that the corresponding vertex v_i^* has to be dominated and v_i represents that v_i^* can dominate its neighbors (and itself). In addition, there are four sets of auxiliary vertices, namely a set S containing n vertices and three sets X, Y, and Z, each containing $k + 1$ vertices. The arcs of G are as follows.

- One arc from v_i to d_j if and only if $v_j^* \in N[v_i^*]$.
- One arc from each vertex in V to w_c.
- One arc from each vertex in X to each vertex in Y, from each vertex in Y to each vertex in Z, and from each vertex in Z to each vertex in X.

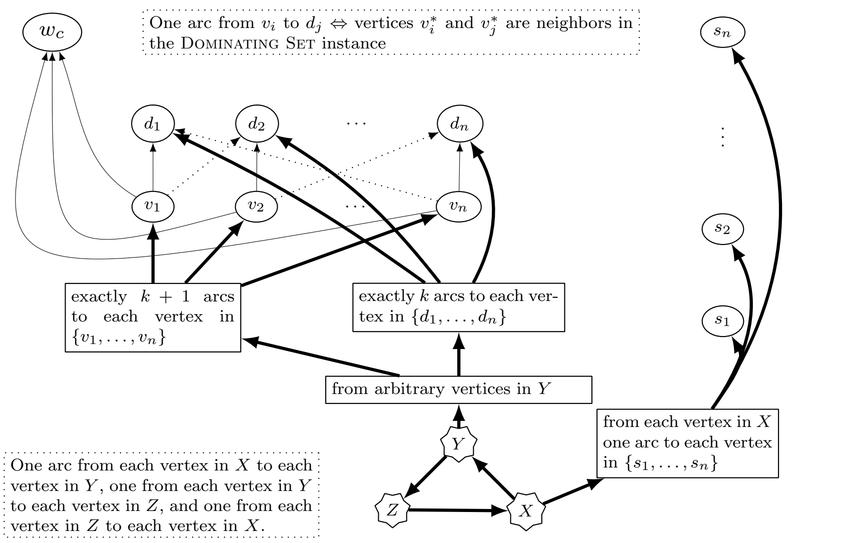

Fig. 3 MID–instance obtained by a parameterized reduction from a DS–instance. Stars represent sets of $k+1$ vertices. Thin lines represent individual arcs whereas bold lines represent multiple arcs. Dotted lines represent individual arcs whose existence depends on the original instance. More details can be found in the explanation boxes.

– One arc from each of k arbitrarily chosen vertices in Y to each vertex in D.
– One arc from each vertex in Y to each vertex in V.
– One arc from each vertex in X to each vertex in S.

This finishes the description of the construction. From the above construction, we immediately get that the distinguished vertex w_c has indegree n and each vertex in $V \cup X \cup Y \cup Z \cup S$ has indegree $k+1$. Since each vertex d_i has an ingoing arc from v_i and k in-neighbors from Y, the vertices in D have indegree at least $k+1$.

Furthermore, it is easy to verify that $(W, E \setminus (X \times Y))$ is acyclic (see Figure 3) and, since there are $(k+1)^2$ arcs between X and Y, the feedback arc set number s_a is at most $(k+1)^2$. Hence, it remains to prove the following.

Claim: (G^*, k) is a yes-instance of DS if and only if $(G, w_c, n-k)$ is a yes-instance of MID.

"⇒": Let $V_d^* \subseteq V^*$ be a size-k dominating of G^*. We show that $M_d := \{v_i \in V \mid v_i^* \notin V_d^*\}$ is a solution for MID. Since $|M_d| = n-k$ and w_c has indegree n in G, w_c has indegree k in $G - M_d$. We show that all other vertices have degree at least $k+1$. By construction, every vertex in G has indegree at least $k+1$. Since from the vertices in V_d^* there are only arcs to $D \cup \{w_c\}$, only vertices from $D \cup \{w_c\}$ can have smaller indegrees in $G - M_d$ than in G. Because V_d^* is a dominating set, every d_i has at least one in-neighbor within $V \setminus M_d$. Moreover, every d_i has k further in-neighbors in Y. Hence, each vertex in D has indegree at least $k+1$. Thus, $(G, w_c, n-k)$ is a yes-instance of MID.

"$\Leftarrow$": Consider a yes-instance $(G, w_c, n-k)$ of MID with solution M_d. We show that $V_d^* := \{v_i^* \in V^* \mid v_i \in V \setminus M_d\}$ is a size-k dominating set of G^*.

We first prove that V_d^* has cardinality k. To this end, we show by contradiction that the indegree of w_c in $G - M_d$ is k and hence M_d contains only vertices from V. Assume that w_c has indegree at least $k+1$ in $G - M_d$. Then, every other vertex must have indegree greater than $k+1$ in $G - M_d$. Since every vertex in S has indegree $k+1$, it follows that $S \subseteq M_d$ and hence $|M_d| \geq n$; a contradiction. Consequently, $|V \cap M_d| = n-k$ and, hence, V_d^* has cardinality k.

It remains to show that V_d^* is a dominating set. Assume that there is a vertex $v_i^* \in V^*$ not dominated by any vertex in V_d^*. This implies that d_i has no in-neighbor from V in $G - M_d$. Moreover, by construction, d_i has only k in-neighbors in $G - V$. As argued above, d_i is not in M_d since M_d contains only vertices from V. Hence, d_i and w_c have indegree k in $G - M_d$, that is, w_c is not the only vertex with minimum indegree; a contradiction. Altogether, it follows that V_d^* is a size-k dominating set. □

In the remainder of this section, we show fixed-parameter tractability for MID with respect to the combined parameter "feedback vertex set number s_v and number k of vertices to delete". The corresponding branching algorithm relies on the following lemma that bounds the number of neighbors that w_c can have in a yes-instance with the help of the parameter.

Lemma 2 *Consider a yes-instance $(G = (V, E), w_c, k)$ of* MID *and a corresponding solution $S \subseteq V \setminus \{w_c\}$. Let $i := |S \cap N_{\text{in}}(w_c)|$. Then, the indegree of w_c in G is at most $i + s_v$, where s_v denotes the feedback vertex set number of G.*

Proof The proof is by contradiction. Let $V_f \subseteq V$ be a feedback vertex set of size s_v. Assume that $\deg_{\text{in}}(w_c) > s_v + i$. For every subgraph G' of G obtained by deleting k vertices from $G - \{w_c\}$, observe the following. First, since $G' - V_f$ is acyclic, there must be a vertex v with indegree zero in $G' - V_f$. Hence, the indegree of v in G' is at most s_v (in case that v has one ingoing arc from every vertex in V_f). Second, since $\deg_{\text{in}}(w_c) > s_v + i$ in G and $|S \cap N_{\text{in}}(w_c)| = i$, it follows that $\deg_{\text{in}}(w_c) > s_v$ in G'. Consequently, there is no size-k subset S containing i neighbors of w_c such its deletion makes w_c a vertex with minimum indegree; a contradiction. □

Based on Lemma 2 we obtain a branching algorithm for MID

Theorem 8 Min-Indegree Deletion *can be solved in $O((k+1)^{s_v} \cdot k \cdot n^2)$ time.*

Proof The algorithm is displayed in Algorithm 3. Basically, it branches on all up-to-size-k subsets of the in-neighborhood of w_c and checks whether a corresponding subset can be extended to a solution.

Algorithm 3 **`MID-search`**. The input consists of a MID-instance (G, w_c, k). If (G, w_c, k) is a yes-instance, `MID-search` returns a solution, otherwise "no".

1: **for** each $i := 0$ to k **do** ▷ i represents the number of deleted neighbors of w_c
2:     **if** $|N_{\text{in}}(w_c)| \leq i + s_v$ **then** ▷ otherwise there is no solution for this i
3:         **for** each size-i subset $U \subseteq N_{\text{in}}(w_c)$ **do** ▷ at most $\binom{i+s_v}{i}$
4:             remove $D := N_{\text{in}}(w_c) \setminus U$ from G
5:             $M_d := D$
6:             **while** there is a vertex $d \neq w_c$ with indegree at most i **do**
7:                 remove d from G
8:                 $M_d := M_d \cup \{d\}$
9:             **if** $|M_d| \leq k$ **then**
10:                 **return** M_d
11: **return** "no"

To see the correctness, observe that the condition $|N_{\text{in}}(w_c)| \leq i+s_v$ (line 2) directly follows from Lemma 2. The iteration loops in lines 1 and 3 explore all possible subsets of in-neighbors of w_c that can be part of a solution. For each such subset the final degree of w_c is fixed at i and hence all remaining vertices with indegree at most i must be deleted to obtain a solution (lines 6–8). If this is possible by deleting at most k vertices in total, then `MID-search` returns a corresponding solution set.

It remains to analyze the running time. In the worst case, we execute at most k times the first loop (line 1). In the second loop (line 3), we try at most $\binom{s_v+k}{k}$ subsets. Thus, we have

$$\binom{s_v + k}{k} = \frac{(s_v + k)!}{k! \cdot (s_v + k - k)!} = \frac{\prod_{i=1}^{s_v}(k + i)}{s_v!} \leq \left(\frac{k+1}{1}\right)^{s_v}$$

subsets. The third loop (line 6) can be executed in $O(n^2)$ time. □

Theorem 7 provides a relative lower bound for the parameterized complexity with respect to the feedback set parameters s_v and s_a. An upper bound, namely polynomial-time solvability for constant values of s_v (that is, membership in XP), follows from Theorem 8.

5 Conclusion

We introduced the NP-hard vertex deletion problem Min-Degree Deletion on undirected graphs. For Min-Degree Deletion and its directed counterpart Min-Indegree Deletion [6] we provided a number of results concerning their fixed-parameter tractability with respect to the parameter solution size and several parameters measuring the input graph's tree-likeness (see Table 1 in the introductory section for an overview). In particular, our fixed-parameter algorithm for Min-Indegree Deletion for the combined parameter (k, s_v) indicates that electoral control by removing candidates for Llull voting [6,16] is computationally feasible in interesting special cases. Remarkably, one can adapt the algorithm for Min-Degree Deletion on undirected graphs (Theorem 2) to directed graphs by using the tree decomposition of the underlying undirected graph.

The aim of this work was to start a systematic parameterized complexity analysis of two so far widely unstudied simple graph problems. There remain several open questions for future research. Clearly, it is desirable to spot further application scenarios for both problems. Concerning algorithmic challenges, note that regarding the parameters maximum degree and indegree, respectively, fixed-parameter tractability follows easily from a simple branching strategy [6]. However, it is unclear whether there exist polynomial-size problem kernels in these cases. Besides further parameterized complexity studies in the spirit of multivariate algorithmics [17,25,29], it also remains open to pursue studies concerning the polynomial-time approximability of these problems.

Acknowledgements We are grateful to two anonymous referees of *Algorithmica* whose constructive feedback helped to improve the quality of our presentation.

References

1. Bafna, V., Berman, P., Fujito, T.: A 2-approximation algorithm for the undirected feedback vertex set problem. SIAM Journal on Discrete Mathematics **12**(3), 289–297 (1999)
2. Balasundaram, B., Butenko, S., Hicks, I.V.: Clique relaxations in social network analysis: The maximum k-plex problem. Operations Research **59**(1), 133–142 (2011)
3. Bar-Yehuda, R., Geiger, D.: Approximation algorithms for the feedback vertex set problem with applications to constraint satisfaction and Bayesian inference. SIAM J. Comput. **27**, 942–959 (1998)
4. Becker, A., Geiger, D.: Optimization of pearl's method of conditioning and greedy-like approximation algorithms for the vertex feedback set problem. Artificial Intelligence **83**, 167–188 (1996)
5. Betzler, N., Bredereck, R., Niedermeier, R., Uhlmann, J.: On bounded-degree vertex deletion parameterized by treewidth. Discrete Applied Mathematics **160**(1–2), 53–60 (2012)
6. Betzler, N., Uhlmann, J.: Parameterized complexity of candidate control in elections and related digraph problems. Theoretical Computer Science **410**(52), 5425–5442 (2009)
7. Bodlaender, H., Downey, R., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. Journal of Computer and System Sciences **75**(8), 423–434 (2009)
8. Bodlaender, H.L.: A linear time algorithm for finding tree-decompositions of small treewidth. SIAM Journal on Computing **25**, 1305–1317 (1996)
9. Bodlaender, H.L.: A partial k-arboretum of graphs with bounded treewidth. Theoretical Computer Science **209**(1–2), 1–45 (1998)
10. Bodlaender, H.L.: Kernelization: New upper and lower bound techniques. In: Proceedings of the 4th International Workshop on Parameterized and Exact Computation (IWPEC '09), *LNCS*, vol. 5917, pp. 17–37. Springer (2009)
11. Bodlaender, H.L., Koster, A.M.C.A.: Treewidth computations II. Lower bounds. Information and Computation **209**(7), 1103–1119 (2011)
12. Bodlaender, H.L., Thomassé, S., Yeo, A.: Kernel bounds for disjoint cycles and disjoint paths. Theoretical Computer Science **412**(35), 4570–4578 (2011)
13. Cao, Y., Chen, J., Liu, Y.: On feedback vertex set new measure and new structures. In: Proceedings of the 12th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT '10), *LNCS*, vol. 6139, pp. 93–104. Springer (2010)

14. Dom, M., Lokshtanov, D., Saurabh, S.: Incompressibility through colors and IDs. In: Proceedings of the 36th International Colloquium on Automata, Languages, and Programming (ICALP '09), *LNCS*, vol. 5555, pp. 378–389. Springer (2009)
15. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Springer (1999)
16. Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L.A., Rothe, J.: Llull and Copeland voting computationally resist bribery and constructive control. Artificial Intelligence **35**(1), 275–341 (2009)
17. Fellows, M.: Towards fully multivariate algorithmics: Some new results and directions in parameter ecology. In: Proceedings of the 20th International Workshop (IWOCA '09), *LNCS*, vol. 5874, pp. 2–10. Springer (2009)
18. Fellows, M.R., Guo, J., Moser, H., Niedermeier, R.: A generalization of Nemhauser and Trotter's local optimization theorem. Journal of Computer and System Sciences **77**(6), 1141–1158 (2011)
19. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer (2006)
20. Fortnow, L., Santhanam, R.: Infeasibility of instance compression and succinct PCPs for NP. Journal of Computer and System Sciences **77**(1), 91–106 (2011)
21. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman (1979)
22. Guo, J., Niedermeier, R.: Invitation to data reduction and problem kernelization. ACM SIGACT News **38**(1), 31–45 (2007)
23. Huberman, B.A., Romero, D.M., Wu, F.: Social networks that matter: Twitter under the microscope. First Monday **14(1)** (2009)
24. Kloks, T.: Treewidth. Computations and Approximations, *Lecture Notes in Computer Science*, vol. 842. Springer, Berlin (1994)
25. Komusiewicz, C., Niedermeier, R.: New races in parameterized algorithmics. In: Proceedings of the 37th International Symposium on Mathematical Foundations of Computer Science (MFCS '12), *LNCS*, vol. 7464, pp. 19–30. Springer (2012)
26. Lokshtanov, D., Misra, N., Saurabh, S.: Kernelization – preprocessing with a guarantee. In: The Multivariate Algorithmic Revolution and Beyond – Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday, *LNCS*, vol. 7370, pp. 129–161. Springer (2012)
27. Moser, H., Niedermeier, R., Sorge, M.: Exact combinatorial algorithms and experiments for finding maximum k-plexes. Journal of Combinatorial Optimization (2011). Available electronically
28. Niedermeier, R.: Invitation to Fixed-Parameter Algorithms. Oxford University Press (2006)
29. Niedermeier, R.: Reflections on multivariate algorithmics and problem parameterization. In: Proceedings of the 27th International Symposium on Theoretical Aspects of Computer Science (STACS '10), *LIPIcs*, vol. 5, pp. 17–32. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2010)
30. Potterat, J., Phillips-Plummer, L., Muth, S., Rothenberg, R., Woodhouse, D., Maldonado-Long, T., Zimmerman, H., Muth, J.: Risk network structure in the early epidemic phase of HIV transmission in Colorado Springs. Sexually Transmitted Infections **78** (Supplement 1), 159–163 (2002)
31. Ramachandramurthi, S.: The structure and number of obstructions to treewidth. SIAM Journal on Discrete Mathematics **10**, 146–157 (1997)
32. Romm-Livermore, C., Setzekorn, K.: Social Networking Communities and E-Dating Services: Concepts and Implications. Information Science Reference (2008)
33. Seidman, S., Foster, B.: A graph-theoretic generalization of the clique concept. Journal of Mathematical Sociology **6**(1), 139–154 (1978)
34. Thomassé, S.: A $4k^2$ kernel for feedback vertex set. ACM Transactions on Algorithms **6**(2), 32:1–32:8 (2010)
35. Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications. Cambridge University Press (1994)