


Linear Kernels for Outbranching Problems in Sparse Digraphs

Marthe Bonamy¹ · Łukasz Kowalik²  ·
Michał Pilipczuk² · Arkadiusz Socała²

Received: 10 November 2015 / Accepted: 4 November 2016 / Published online: 15 November 2016
© The Author(s) 2016. This article is published with open access at Springerlink.com

Abstract In the k -LEAF OUT- BRANCHING and k -INTERNAL OUT- BRANCHING problems we are given a directed graph D with a designated root r and a nonnegative integer k . The question is whether there exists an outbranching rooted at r that has at least k leaves, or at least k internal vertices, respectively. Both these problems have been studied from the points of view of parameterized complexity and kernelization, and in particular for both of them kernels with $O(k^2)$ vertices are known on general graphs. In this work we show that k -LEAF OUT- BRANCHING admits a kernel with $O(k)$ vertices on \mathcal{H} -minor-free graphs, for any fixed family of graphs \mathcal{H} , whereas k -INTERNAL OUT- BRANCHING admits a kernel with $O(k)$ vertices on any graph class of bounded expansion.

Work partially supported by the ANR Grant EGOS (2012–2015) 12 JS02 002 01 (MB), by the National Science Centre of Poland, Grants Number 2013/09/B/ST6/03136 (Ł.K., AS). This work was done while Michał Pilipczuk has been holding a post-doc position at Warsaw Centre of Mathematics and Computer Science, and has been supported by the Foundation for Polish Science via the START stipend programme. A preliminary conference abstract of this work was presented at IPEC'15 and it is published in the conference proceedings [6].

✉ Łukasz Kowalik
kowalik@mimuw.edu.pl

Marthe Bonamy
marthe.bonamy@labri.fr

Michał Pilipczuk
michal.pilipczuk@mimuw.edu.pl

Arkadiusz Socała
arkadiusz.socala@mimuw.edu.pl

¹ CNRS, LaBRI, Université de Bordeaux, Talence, France

² Institute of Informatics, University of Warsaw, Warsaw, Poland

Keywords Kernelization · Outbranching · Sparse graph · Bounded expansion · H -minor-free graphs

1 Introduction

Kernelization is a thriving research direction within parameterized complexity that aims at understanding the computational power of polynomial-time preprocessing procedures via a rigorous mathematical framework. Its central notion is the definition of a *kernelization algorithm*, or simply a *kernel*: Given an instance (I, k) of some parameterized problem L , a kernelization algorithm reduces (I, k) in polynomial time to an equivalent instance (I', k') of L so that $|I'|, k' \leq f(k)$ for some computable function f of the parameter k only; function f is called the *size* of the kernel. While for a decidable problem L the existence of *any* kernelization algorithm is equivalent to fixed-parameter tractability of the problem, we are most interested in finding small kernels, possibly of polynomial or even linear size. For concreteness, in this paper we concentrate on parameterized graph problems, so we always assume that the input instance is a graph.

One of the most influential ideas in the search for small kernels was to restrict the input graph to belong to some *sparse* graph class, e.g., to be planar, bounded-genus, or H -minor-free for some fixed H . Starting with the groundbreaking work of Alber et al. [1], who showed a kernel of size $335k$ for DOMINATING SET on planar graphs, numerous strong kernelization results were shown on planar, bounded genus, and H -minor-free graphs; these results often concern problems that on general graphs are intractable in the parameterized sense. A milestone in this theory is the development of the technique of *meta-kernelization* by Bodlaender et al. [5], further refined by Fomin et al. [19]. Informally speaking, using this methodology one can explain the existence of linear kernels for many parameterized problems by proving that the problem behaves in a “bidimensional” way and possesses certain finite-state properties. Whereas verifying the latter usually boils down to a quick technical check, the bidimensionality requirement is quite restrictive. Roughly speaking, it says that the optimum solution size is large whenever a large two-dimensional structure (like a grid minor) can be found in the graph, and the problem behaves monotonically under minor operations.

The concept of bidimensionality was initially introduced by Demaine et al. [11] as a technique for obtaining subexponential parameterized algorithms, in this case typically with the running time of the form $2^{\tilde{O}(\sqrt{k})} \cdot n^{O(1)}$. Note that, provided the considered problem can be solved in time $2^{\tilde{O}(t)} \cdot n^{O(1)}$ on graphs of treewidth t , the existence of a kernel with $O(k)$ vertices for the problem on planar/bounded genus/ H -minor free graphs immediately implies an algorithm for the problem with running time $2^{\tilde{O}(\sqrt{k})} + n^{O(1)}$, as graphs from these classes have $O(\sqrt{n})$ treewidth. Thus, for many natural problems the existence of a linear kernel on sparse graphs is a stronger property than admitting a subexponential parameterized algorithm.

While the techniques of bidimensionality and meta-kernelization are elegant and have many important applications, they have certain limitations that make them inapplicable to several important families of problems, for instance problems on

directed graphs or problems with prescribed sets of distinguished vertices like STEINER TREE. Therefore, significant effort has been put into investigating the existence of subexponential parameterized algorithms and small kernels outside the framework of bidimensionality [13, 18, 24, 26, 30, 31].

In this work we are interested in two problems investigated by Dorn et al. [13], namely k -LEAF OUT- BRANCHING (LOB) and k -INTERNAL OUT- BRANCHING (IOB). In both problems, we are given a directed graph D with a specified root r and a nonnegative integer k . By an *outbranching rooted at r* we mean a spanning tree of D with all the edges oriented away from r . A vertex of D is a *leaf* in an outbranching T if it has outdegree 0 in T , and is *internal* otherwise. In LOB the question is to verify the existence of an outbranching rooted at r that has at least k leaves, whereas in IOB we instead ask for an outbranching rooted at r with at least k internal vertices. Both problems enjoy the existence of kernels with $O(k^2)$ vertices on general graphs [9, 23], however up to this work no better kernels were known even in the case of planar graphs. Indeed, the directed nature of both problems prevents them from satisfying even the most basic properties needed for the bidimensionality tools to be applicable. A one more hint on the particular hardness of this kind of problems is the NP-hardness of the MINIMUM LEAF OUTBRANCHING problem even in the case of directed path-width (DAG-width, directed tree-width, respectively) equal to 1, as proven by Dankelmann et al. [10].

Dorn et al. [13] designed subexponential parameterized algorithms with running time $2^{\tilde{O}(\sqrt{k})} \cdot n^{O(1)}$ for both problems on H -minor-free graphs.¹ They did it, however, by circumventing in both cases the need of obtaining a linear kernel. In the case of LOB they show how to apply preprocessing rules to obtain an instance that can be still large in terms of k , but has treewidth $O(\sqrt{k})$ so that the dynamic programming on a tree decomposition can be applied. In the case of IOB they apply a variant of Baker's layering technique.

Our results and techniques In this work we fill the gap left by Dorn et al. [13] and prove that both LOB and IOB admit linear kernels on H -minor-free graphs. In fact, for IOB our approach works even in the more general setting of graph classes of bounded expansion (see Sect. 2 for a definition). By slightly abusing notation, in what follows we say that a directed graph D belongs to some class of undirected graphs (e.g., is H -minor free) if the underlying undirected graph of D has this property.

Theorem 1 *Let H be a fixed graph. There is an algorithm that, given an instance (D, k) of LOB where D is H -minor-free, in polynomial time either solves the instance (D, k) , or outputs an equivalent instance (D', k') of LOB where $|V(D')| = O(k)$, $k' \leq k$, and D' is H -minor free. The algorithm does not need to know H .*

Note that Theorem 1 implies also a kernel of linear size for any minor-closed family of graphs \mathcal{G} . Indeed, by the Robertson and Seymour's graph minor theorem there exists a fixed finite family \mathcal{H} such that \mathcal{G} contains exactly graphs that are H -minor free for

¹ We remark that Dorn et al. state the result for IOB only for apex-minor-free graphs, but a combination of their approach with the contraction decomposition technique of Demaine et al. [12] immediately generalizes the result to H -minor-free graphs.

every $H \in \mathcal{H}$. By Theorem 1, for any input graph $D \in \mathcal{G}$, the output graph D' is H -minor free for every $H \in \mathcal{H}$. Hence, D' is in \mathcal{G} . In particular, it follows that Theorem 1 implies linear kernels for planar graphs and other graphs embeddable on a surface of bounded genus.

We also point out that the algorithm in Theorem 1 is the same for every fixed graph H . However, the fixed graph H matters in the analysis. In particular, the bound $O(k)$ on the size of the kernel hides a constant which depends on H .

The second of our main results is as follows.

Theorem 2 *Let \mathcal{G} be a hereditary graph class of bounded expansion. There is an algorithm that, given an instance (D, k) of IOB where $D \in \mathcal{G}$, in polynomial time either resolves the instance (D, k) , or outputs an equivalent instance (D', k) of IOB where $|V(D')| = O(k)$ and D' is an induced subgraph of D .*

By applying these kernelization algorithms and then running dynamic programming on a tree decomposition of the obtained graph, we easily obtain the following corollary.

Theorem 3 *Let H be a fixed graph. Then both LOB and IOB can be solved in time $2^{O(\sqrt{k})} + n^{O(1)}$ when the input is an n -vertex H -minor-free graph.*

Algorithms with a similar running time—but with additional $\log k$ factor in the exponent—were obtained by Dorn et al. [13]. If one follows their approach, then for LOB it is possible to shave off this factor in the exponent just by improving the dynamic programming on a tree decomposition. However, for IOB the logarithmic factor is caused also by an application of the layering technique, and hence such a replacement and manipulation of parameters in layering would only improve $\log k$ to $\sqrt{\log k}$. By constructing a truly linear kernel we are able to shave this factor completely off. We remark that the running time given by Theorem 3 is optimal under the Exponential Time Hypothesis even on planar graphs; see Sect. 5 for further details.

To prove Theorems 1 and 2, we revisit the quadratic kernels on general graphs given by Daligault and Thomassé [9] (for LOB) and by Gutin et al. [23] (for IOB). For LOB we need to modify the approach substantially, as the core reduction rule used by Daligault and Thomassé is the following: whenever there is a *cut-vertex* in the graph—a vertex whose removal makes some other vertex not reachable from r —then it is safe to *shortcut* it: remove it and add an arc from every its in-neighbor to every its out-neighbor. Observe that an application of this rule does not preserve H -minor-freeness, so the kernel of Daligault and Thomassé [9] may start with an H -minor free graph and go outside of this class.

To circumvent this problem, we exploit the structural approach proposed by Dorn et al. [13]. While not achieving a linear kernel in the precise sense, Dorn et al. are able to simplify the structure of the instance so that it fits their purposes. The main idea is to contract cut-edges instead of shortcutting cut-vertices, which is a weaker operation that, however, preserves H -minor-freeness. For a graph G and a set X let $G \setminus X = G[V(G) \setminus X]$. Dorn et al. are able to expose a set of so-called *special* vertices S of size linear in k such that $G \setminus S$ has constant pathwidth; this is already enough to employ the bidimensionality technique. To obtain a linear kernel, we need to perform a much more refined analysis of the instance. More precisely, we construct a set S

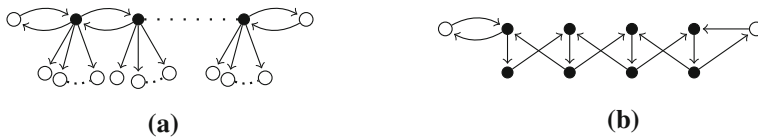


Fig. 1 Different types of bipaths. **a** A weak bipath. **b** A fat bipath (the black vertices may have out-neighbors other than those depicted and some of the vertical edges may be contracted)

with $|S| = O(k)$ such that $G \setminus S$ consists of *fat bipaths*: chains as depicted in Fig. 1, possibly with some vertical (cut)edges contracted, and with outgoing edges with heads in S . After contracting the vertical edges, such a fat bipath becomes a weak bipath: a bidirectional path possibly with outgoing edges with heads in S . Weak bipaths are crucial in the structural approach of Daligault and Thomassé [9], and our fat bipaths can be thought of as more fuzzy variants of weak bipaths that cannot be reduced due to the inability to shortcut cut-vertices.

To obtain a linear kernel, we need to reduce the total length of the fat bipaths. For this, we use concepts borrowed from the analysis of graph classes of bounded expansion, of which H -minor-free classes are special cases. Very recently Drange et al. [15] announced a linear kernel for DOMINATING SET on graph classes of bounded expansion, and the main tool used there is the analysis of the number of different neighborhoods that can arise in a graph G from a bounded expansion graph class \mathcal{G} . Essentially, there is a constant c such that for every $X \subseteq V(G)$ there are only $O(|X|)$ vertices in $V(G) \setminus X$ that are adjacent to more than c vertices in X , while the vertices of $V(G) \setminus X$ that are adjacent to at most c vertices in X can be grouped into $O(|X|)$ classes with exactly the same neighborhoods. We apply this idea to the instance at hand with the interior of every fat bipath contracted to one vertex. Thus, we infer that there are only $O(k)$ fat bipaths that are adjacent to more than c special vertices, and their total length can be bounded by $O(k)$ using reduction rules. On the other hand, fat bipaths with neighborhoods of size at most c are reduced within their neighborhood classes, whose number is also $O(k)$.

The same neighborhood diversity argument plays the key role also in our kernel for IOB (Theorem 2). The idea of Gutin et al. [23] is that if a solution to the instance cannot be found immediately by a simple local search, then one can expose a vertex cover U of size at most $2k$ in the graph. The vertices of $V(D) \setminus U$ are reduced using an argument involving crown decompositions in an auxiliary graph where vertices of $V(D) \setminus U$ are matched to pairs of adjacent vertices of U ; this gives a quadratic dependence on k of the size of the kernel. We observe that in case D belongs to a class of bounded expansion, then there is only $O(|U|)$ [thus in particular $O(k)$] vertices of $V(D) \setminus U$ that have super-constant neighborhood size in U , while the others are grouped into $O(|U|)$ [thus in particular $O(k)$] neighborhood classes, each of which can be reduced to constant size using the same approach via crown decompositions.

For IOB we did not need any edge contractions in the reduction rules, so the kernelization procedure works on any graph class of bounded expansion. However, for LOB it seems necessary to apply contractions of subgraphs of unbounded diameter, e.g., to reduce long paths that contribute with at most one leaf to the solution. While the last phase relies mostly on the bounded expansion properties of the graph class,

we need to allow contractions in the reduction rules and hence we do not achieve the same level of generality as for IOB.

We see the additional advantage of our approach in its simplicity. Instead of relying on complicated decomposition theorems for H -minor free graphs, which is a standard technique in such a setting, we use the methodology proposed by Drange et al. [15]: To exploit purely combinatorial, abstract notions of sparsity, like the concept of bounded expansion, and in this manner obtain a much cleaner treatment of the considered graph classes. Of particular interest is the usefulness of the approach of grouping vertices according to their neighborhoods in some fixed modulator X , which is the key idea in [15].

Organization of the paper In Sect. 2 we give preliminaries on tools borrowed from the analysis of graph classes of bounded expansion. Sections 3 and 4 are devoted to the proofs of Theorems 1 and 2, respectively. In Sect. 5 we derive Theorem 3 as a corollary, and discuss the optimality of the obtained algorithms. We conclude with some closing remarks in Sect. 6.

Definitions In this paper we deal with digraphs. Let $D = (V, E)$ be a digraph. Consider an edge $(u, v) \in E$. We say that v is an *out-neighbor* of u and u is an *in-neighbor* of v and also that v and u are *neighbors* of each other. We also say that v is a *head* and u is a *tail* of (u, v) . For any vertex v we denote the sets of all its neighbors, out-neighbors and in-neighbors by $N_D(v)$, $N_D^+(v)$ and $N_D^-(v)$, respectively. Moreover, the *degree*, *out-degree*, and *in-degree* of v are defined as $\deg_D(v) = |N(v)|$, $\deg_D^+(v) = |N^+(v)|$, and $\deg_D^-(v) = |N^-(v)|$. We omit the subscripts and write simply $N(v)$ or $\deg(v)$ whenever it does not lead to ambiguity. For any set $S \subseteq V$ we denote $N_D^-(S) = \bigcup_{v \in S} N_D^-(v) \setminus S$ and $N_D^+(S) = \bigcup_{v \in S} N_D^+(v) \setminus S$. For any arc $e \in E$, we define $D - e = (V, E \setminus \{e\})$. For any vertex $v \in V$, we define $D - v = D[V \setminus \{v\}]$ as the digraph obtained from D by removing v and every arc incident to v . Let $G = (V, E)$ be a graph. To avoid defining V , we sometimes denote by $|G|$ the order $|V|$ of G .

2 Preliminaries on Sparse Graphs

In this section we recall some definitions and basic properties of sparse graphs, in particular d -degenerate graphs, bounded expansion graphs and H -minor-free graphs. Although in this section we refer to undirected graphs, all the notions and claims apply also to digraphs, by looking at the underlying undirected graph.

Definitions We say that graph G is k -degenerate when every subgraph of G has a vertex of degree at most k . This implies (and in fact is equivalent to) that we can remove all the edges of G by repeatedly removing vertices of degree at most k . It follows that G has at most $k|V(G)|$ edges. The *degeneracy* of a graph is the smallest value of k for which it is k -degenerate. Degeneracy is closely linked to *arboricity*, i.e., minimum number $\text{arb}(G)$ of forests that cover the edges of G : it is known that degeneracy is between $\text{arb}(G)$ and $2 \text{arb}(G)$.

Recall that a graph H is a *minor* of graph G if there exists a *minor model* $(I_u)_{u \in V(H)}$ of H in G that satisfies the following properties:

- sets I_u for $u \in V(H)$ are pairwise disjoint subsets of $V(G)$ that induce connected subgraphs;
- for each $(u, v) \in E(H)$, there exist $x_u \in I_u$ and $x_v \in I_v$ such that $x_u x_v \in E(G)$.

For any fixed graph H , the class of H -minor-free graphs contains all the graphs G that do not have H as a minor. Note that H -minor free graphs are closed under minor operations: vertex and edge deletions, and edge contractions. For example, graphs embeddable into a constant genus surface are H -minor-free for some fixed H ; in particular, by Kuratowski's theorem, planar graphs are K_5 -minor free and $K_{3,3}$ -minor free. The celebrated Decomposition Theorem of Robertson and Seymour implies, in a sense, a reverse implication: every H -minor-free graph can be decomposed into parts that are close to being embeddable into surfaces of bounded genus. The following lemma provides a connection between H -minor-free graphs and degeneracy.

Lemma 1 (see Lemma 4.1 in [27]) *Any H -minor free graph is d_H -degenerate for $d_H = O(|H|\sqrt{\log|H|})$.*

Definitions Let r be a nonnegative integer. If a minor model $(I_u)_{u \in V(H)}$ satisfies in addition that $G[I_u]$ has radius at most r for each $u \in V(H)$, then $(I_u)_{u \in V(H)}$ is an r -shallow minor model of H , and we say that H is an r -shallow minor of G . If \mathcal{G} is a class of graphs, then by $\mathcal{G} \nabla r$ we denote the class of all r -shallow minors of graphs from \mathcal{G} ; note that $\mathcal{G} \nabla 0$ are all subgraphs of graphs of \mathcal{G} . We now define the *greatest reduced average degree (grad)* of a class \mathcal{G} at depth r as

$$\nabla_r(\mathcal{G}) = \sup_{H \in \mathcal{G} \nabla r} \frac{|E(H)|}{|V(H)|}.$$

That is, we take the greatest edge density among the r -shallow minors of \mathcal{G} . Class \mathcal{G} is said to be of *bounded expansion* if $\nabla_r(\mathcal{G})$ is a finite constant for every r . Observe that then the graphs from \mathcal{G} are in particular d -degenerate for $d = \lfloor 2\nabla_0(\mathcal{G}) \rfloor$ (see e.g. [27], Prop. 3.1). For a single graph G , we denote $\nabla_r(G) = \nabla_r(\{G\})$.

Consider the class \mathcal{G}_H of H -minor-free graphs. By Lemma 1, every graph $G \in \mathcal{G}_H$ has at most $d_H \cdot |V(G)|$ edges. Since \mathcal{G}_H is closed under taking minors, it follows that $\mathcal{G}_H \nabla r = \mathcal{G}_H$ for every nonnegative r , so also $\nabla_r(\mathcal{G}_H) \leq d_H$. Thus, H -minor-free graphs form a class of bounded expansion with all the grads bounded independently of r .

In this paper we do not use the original definition of bounded expansion graphs, but we rather rely on the point of view of diversity of neighborhoods, which was found to be very useful in [15]. More precisely, we now use the following result from [20, Lemma 6.6]; the statement with adjusted notation is taken verbatim from [15].

Proposition 1 (Proposition 2.5 of [15]) *Let G be a graph, $X \subseteq V(G)$ be a vertex subset, and $R = V(G) \setminus X$. Then for every integer $p \geq \nabla_1(G)$ it holds that*

1. $|\{v \in R : |N(v) \cap X| \geq 2p\}| \leq 2p \cdot |X|$, and
2. $|\{A \subseteq X : |A| < 2p \text{ and } \exists_{v \in R} A = N(v) \cap X\}| \leq (4p + 2p)|X|$.

Consequently, the following bound holds:

$$|\{A \subseteq X : \exists_{v \in R} A = N(v) \cap X\}| \leq (4^{\nabla_1(G)} + 4\nabla_1(G)) \cdot |X|.$$

See also Lemma 6.6 from [20]. We need a strengthening of the first claim of Proposition 1.

Lemma 2 *Let $G = (X, Y, E)$ be a bipartite graph of degeneracy at most d . Then*

$$\sum_{\substack{y \in Y \\ \deg_G(y) > 2d}} \deg_G(y) \leq 2d|X|.$$

Proof Let $Z = \{y \in Y : \deg_G(y) > 2d\}$. Consider $G' = G[X \cup Z]$, and observe that $|E(G')| = \sum_{y \in Z} \deg_G(y)$. Since G' is a subgraph of G , it is d -degenerate and we obtain that

$$d \geq \frac{|E(G')|}{|V(G')|} = \frac{\sum_{y \in Z} \deg_G(y)}{|X| + |Z|}.$$

Consequently $\sum_{y \in Z} \deg(y) \leq d|X| + d|Z|$, so

$$\sum_{y \in Z} (\deg_G(y) - d) \leq d|X|.$$

Observe that since $\deg_G(y) > 2d$ for each $y \in Z$, we have that $\deg_G(y) - d > \deg_G(y)/2$. Hence it follows that $\sum_{y \in Z} \deg_G(y) \leq 2d|X|$. \square

Note that Proposition 1 has the following corollary when applied to H -minor-free graphs.

Corollary 1 *Let H be a graph. There exists $c_H = 2^{O(|H|\sqrt{\log |H|})}$ such that in any H -minor-free bipartite graph $G = (X, Y, E)$, there are at most $c_H \cdot |X|$ vertices in Y with pairwise distinct neighborhoods in X .*

3 k -LEAF OUT-BRANCHING in H -Minor-Free Graphs

In this section we deal with rooted digraphs, i.e., digraphs with one chosen vertex r of in-degree 0, called *root*. In such digraphs we redefine some standard connectivity notions as follows.

Definitions Let (D, r) be a rooted digraph. We say that D is *connected* when every vertex of D is reachable from r . A *cut-vertex* is any vertex $v \in V(D) \setminus \{r\}$ such that $D - v$ is not connected. The set of all cut-vertices of D is denoted by $\text{cv}(D)$. We say that D is *2-connected* if D has no cut-vertex (equivalently, for every vertex $v \in V(D) \setminus \{r\}$ there are at least two paths from r to v that do not share internal vertices). Note that

this condition is trivially satisfied for any vertex that is the endpoint of an arc from r , as we do not enforce that the paths have to be distinct. Similarly, a *cut-edge* is any edge $(u, v) \in E(D)$ such that $D - (u, v)$ is not connected. We say that D is *2-edge-connected* if D has no cut-edge (equivalently, for every vertex $v \in V(D) \setminus \{r\}$ there are at least two edge-disjoint paths from r to v). Note that if (u, v) is a cut-edge then u is a cut-vertex or $u = r$.

Given a cut-vertex u , or $u = r$, we define $P(u)$ as the set of *private neighbors* of u , that is, the set of out-neighbors of u that are not reachable from the root in $D - u$. In particular, all the out-neighbors of r are its private neighbors.

By a *contraction* of edge (a, b) in D we mean the following operation: identify a and b into a newly introduced vertex $v_{(a,b)}$, replace a and b with $v_{(a,b)}$ in every edge of D , and remove all the loops and parallel edges created in this manner. Note that if D is H -minor-free, then it remains H -minor-free after contractions as well.

Following [9], we say that a vertex v of D is *special* if v is of in-degree at least 3 or there is an incoming simple edge, i.e., an edge (u, v) such that $(v, u) \notin E(D)$. The set of all special vertices of D is denoted by $\text{sp}(D)$.

A *weak bipath* P is a sequence of vertices u_1, \dots, u_p for some $p \geq 3$, such that for each $i = 2, \dots, p-1$, we have $N^-(u_i) = \{u_{i-1}, u_{i+1}\} \subseteq N^+(u_i)$. The *length* of P is $p-1$. If additionally $N^+(u_i) = N^-(u_i) = \{u_{i-1}, u_{i+1}\}$ for every $i = 2, \dots, p-1$, we say that P is *proper bipath* (or shortly a *bipath*). u_1 and u_p are called the *extremities* of P .

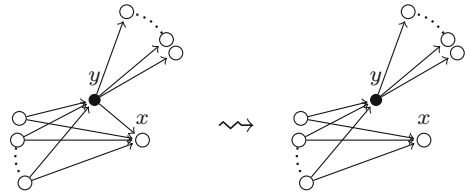
We say that a cut-edge (u, v) is *lonely* when there is no other cut-edge with the tail in u . We call a cut-edge *branching* if there is another cut-edge with the same tail. The graph obtained from D by contracting all lonely cut-edges is denoted by D_c and called the *contracted graph*. Consider a vertex v of D_c . Then either v was created by contracting some set of cut-edges Z in D or $v \in D$. In the first case we define the *bag* B of v as the set of vertices incident to edges in Z . Also, for any edge $(x, y) \in Z$ the vertex x is called a *tail* of B and y is a *head* of B . In the latter case, i.e., when $v \in D$, we define the bag as $B = \{v\}$ and v is both head and tail of B . When B is a bag of v we denote $v_B = v$ and $B_v = B$. If there is exactly one head and exactly one tail of B , then they are denoted by h_B and t_B , respectively.

We say that bags A and B are *linked* if in D there is an edge from A to B and an edge from B to A .

3.1 Our Kernelization Algorithm

In this section we describe our algorithm which outputs a kernel for k -LEAF OUT-BRANCHING. The algorithm exhaustively applies reduction rules. Each reduction rule is a subroutine which finds in polynomial time a certain structure in the graph and replaces it by another structure, so that the resulting instance is equivalent to the original one.

Definitions More precisely, we say that a reduction rule for parameterized graph problem P is *correct* when for every instance (D, k) of P it returns an instance (D', k') such that:

Fig. 2 Rule 4

- (a) (D', k') is an instance of P ,
- (b) (D, k) is a yes-instance of P if and only if (D', k') is a yes-instance of P , and
- (c) $k' \leq k$.

Below we state the rules we use. The rules are applied in the given order, i.e., in each rule we assume that the earlier rules do not apply. We begin with some rules used in the previous works [9].

Rule 1 If there exists a vertex not reachable from r in D , then reduce to a trivial no-instance.

Rule 2 If there exists a cut-vertex v with exactly one incoming edge e , then contract e . Similarly, if there exists a cut-vertex v with exactly one outgoing edge e , then contract e .

Rule 3 Let P be a proper bipath of length 4 in D . Contract any edge of P .

Rule 4 Let x be a vertex of D . If there exists $y \in N^-(x)$ such that the removal of $N^-(x) \setminus \{y\}$ disconnects y from r , then delete the edge (y, x) (Fig. 2).

The correctness of the above reduction rules was proven in [9] (most of them previously appeared in [2]). (In [9], Rule 2 is formulated in a more general way, but we restrict it so that if the input digraph was H -minor-free, then so is the resulting reduced graph.) Let us remark that Rule 4 remains true if $r \in N^-(x) \setminus \{y\}$, and in this case it triggers removal of all the incoming edges apart from the one coming from the root.

Below we introduce two simple rules which will make our argument a bit easier. Note that Rule 6 already appeared in [2], but the proof is very simple so we include it for completeness.

Rule 5 If there are two cut-edges (x_1, y_1) and (x_2, y_2) such that $(x_1, x_2), (x_2, x_1) \in E(D)$, then contract (x_1, x_2) .

Rule 6 If there is a cut-edge (u, v) such that $(v, u) \in E(D)$, then remove (v, u) .

Lemma 3 Rule 5 is correct.

Proof Let D and D' denote the graph before and after applying the reduction. Let x be the vertex obtained by contracting (x_1, x_2) . Let T' be an outbranching in D' . Then an outbranching of D can be obtained by the following procedure:

- remove x and add x_1 and x_2 ;

- replace the edge from the parent p of x by (p, x_1) and (x_1, x_2) , or (p, x_2) and (x_2, x_1) , depending whether $(p, x_1) \in E(D)$ or $(p, x_2) \in E(D)$;
- for every child c of x if $(x_1, c) \in E(D)$, add (x_1, c) , otherwise add (x_2, c) .

Clearly, the number of leaves does not change.

For the second direction, assume that T is an outbranching of D . Then T contains both (x_1, y_1) and (x_2, y_2) , because they are cut-edges. In particular, x_1 and x_2 are not leaves in T . At least one of x_1, x_2 is not a descendant of the other in T , by symmetry assume x_1 is not a descendant of x_2 . Then remove the edge from the parent of x_2 to x_2 and add the edge (x_1, x_2) . Thus we obtained an outbranching T' of D that contains the edge (x_1, x_2) and has at least as many leaves as T . By contracting the edge (x_1, x_2) in T we get an outbranching of D' with the same number of leaves. \square

Lemma 4 Rule 6 is correct.

Proof Let D and D' denote the graph before and after applying the reduction. Since $D' \subseteq D$, any outbranching of D' is also an outbranching of D . Pick any outbranching T of D . Since (u, v) is a cut-edge, $(u, v) \in E(T)$. Then $(v, u) \notin E(T)$. Hence T is also an outbranching of D' . It follows that (D, k) is a yes-instance if and only if (D', k) is a yes-instance. \square

To complete the algorithm we need a final accepting rule which is applied when the resulting graph is too big. In Sect. 3.5 we prove that Rule 7 is correct for H -minor-free graphs for some constant $c = 2^{O(|H|\sqrt{\log|H|})}$.

Rule 7 If the graph has more than $c \cdot k$ vertices, return a trivial yes-instance (conclude that there is a rooted outbranching with at least k leaves in D).

We conclude with the following lemma.

Lemma 5 Let H be a graph. If the input is an H -minor-free graph, then the output of each of the Rules 1–7 is a minor of D , and hence an H -minor-free graph. Moreover, each rule can be recognized and applied in polynomial time, and the degree of the polynomial does not depend on H .

Proof The first claim follows from the fact that the rules modify the graph by means of deletions and contractions only. The second claim is straightforward to check. \square

3.2 A Few Simple Properties of the Reduced Graph

In this section we state simple auxiliary lemmas, which will be used in the remainder of the paper.

Lemma 6 Assume reduction Rules 1–4 do not apply to D . Let u be a cut-vertex in D , or $u = r$. Then every private neighbor $v \in P(u)$ has indegree 1 and (u, v) is a cut-edge. In particular, the head of any cut-edge has indegree 1.

Proof If v has indegree at least 2 then either Rule 1 applies, or Rule 4 applies to $x = v$ and y being the other in-neighbor of v . Any edge incoming to a vertex of indegree 1 is a cut-edge, so (u, v) is a cut-edge. The head of any cut-edge is a private neighbor of its tail, so the last claim also follows. \square

Lemma 7 *If the reduction Rules 1–4 do not apply to D then the tail of any cut-edge is not a head of another cut-edge.*

Proof Assume (x, y) and (y, z) are cut-edges. By Lemma 6, $\deg_D^-(y) = 1$. It follows that Rule 2 applies, a contradiction. \square

Lemma 8 *If the reduction rules do not apply to D then every bag is of size at most two and contains at most one edge. In particular, every bag has exactly one head and one tail.*

Proof Assume that there is a bag B of size at least three. Since the cut-edges that get contracted to v_B are lonely, and their heads have indegrees 1 due to Lemma 6, then these edges form a directed path, a contradiction with Lemma 7. The fact that a bag of size 2 cannot contain two edges follows from Rule 6. \square

Lemma 9 *If the reduction Rules 1–4 do not apply to D , then for arbitrary pair of bags A and B every edge from A to B has head in t_B .*

Proof If $|B| = 1$, then the claim is trivial, so assume $|B| \geq 2$. By Lemma 8, $|B| = 2$, i.e., (t_B, h_B) is a cut-edge. If there is an edge from A to B with head in h_B , then $\deg_D^-(h_B) \geq 2$, a contradiction with Lemma 6. \square

Lemma 10 *Assume reduction Rules 1–4 do not apply to D . If bags A and B are linked then there is exactly one edge from A to B and exactly one edge from B to A .*

Proof It suffices to show that there is exactly one edge from A to B , since the other claim is symmetric. Assume for the contradiction that there are two edges $(a_1, b_1), (a_2, b_2) \in A \times B$. Note that $b_1 = b_2$, for otherwise we get a contradiction with Lemma 9. It follows that $a_1 \neq a_2$, since there are no two identical edges in D . Assume without loss of generality that (a_1, a_2) is a cut-edge. Then Rule 4 applies (with $x = b_1$ and $y = a_2$), a contradiction. \square

Lemma 11 *Assume reduction rules do not apply to D . Then $\deg_D^+(r) \geq 2$, all the edges going out of r are branching cut-edges, and each of the out-neighbors of r is a special vertex in D_c .*

Proof We have that $\deg_D^+(r) \geq 2$ because otherwise Rule 2 would apply. Therefore, it suffices to show that every edge (r, u) is a cut-edge, because the head of a branching cut-edge is always special in D_c by Rule 6. This, however, follows from inapplicability of Rule 4 to u . \square

3.3 Decomposition into Weak Bipaths

The following lemma gives a structural connection between weak bipaths and special vertices.

Lemma 12 *Assume no reduction rule applies to D . Let $S \subseteq V(D_c)$ be any set of vertices that contains the root r and every special vertex of D_c . Then one can find weak bipaths P_1, P_2, \dots, P_q , such that:*

- (i) *The sets of internal vertices of P_1, P_2, \dots, P_q form a partition of $V(D_c) \setminus S$.*
- (ii) *The extremities of each P_i belong to S and are distinct.*
- (iii) *The out-neighbors of the internal vertices of each P_i belong to S .*

Proof Consider any vertex $v \in V(D_c)$ such that $v \notin S$. Assume first that $\deg_{D_c}^-(v) = 1$, and let $N_{D_c}^-(v) = \{u\}$. Since v is not special in D_c , we have that also $(v, u) \in E(D_c)$. If $u, v \in D$, then (u, v) would be a cut-edge in D and Rule 6 would apply, a contradiction. Otherwise, the bags of u and v are linked and by Lemmas 9 and 10, there is one edge from the bag of u to the tail of the bag of v ; clearly, this edge is a cut-edge in D . If v was obtained from the contraction of a lonely cut-edge (v_1, v_2) , then this would be a contradiction with Lemma 7. Hence assume $v \in D$. From Lemma 9 we infer that in D there is an edge from v to t_{B_u} . However, the edge from B_u to v has tail in t_{B_u} by Lemma 7. Then again Rule 6 would apply, a contradiction.

It follows that $\deg_{D_c}^-(v) \geq 2$ for each $v \notin S$. Since v is not special, we get that $\deg_{D_c}^-(v) = 2$, and the two of its in-neighbors are also its out-neighbors. Since $r \in S$ and D_c is connected, we have that $D_c - S$ is a set of bidirectional paths, with each endpoint connected by two edges with opposite directions with a vertex of S . Thus we immediately obtain weak bipaths P_1, P_2, \dots, P_q that satisfy (i), (iii), as well as (ii) apart from the claim that the extremities are distinct. Suppose there is a weak bipath $P_i = u, v_2, v_3, \dots, v_{p-1}, u$ such that both its extremities are in fact one vertex $u \in S$. By Lemma 11, $u \neq r$. Regardless whether $u \in D$ or u is obtained by contracting some lonely cut-edge in D , we have that x , the tail of the bag of u , is a cut-vertex in D whose removal disconnects all the bags of the internal vertices of P_i from r . However, by Lemma 9 and the definition of a bipath we have that x has an in-neighbor in the bag of v_2 . Then Rule 4 would apply to x , a contradiction. \square

Definitions Weak bipaths P_1, \dots, P_q given by Lemma 12 are called *maximal bipaths*. Note that for every such maximal bipath $P = v_1, v_2, \dots, v_p$ and every $j = 2, \dots, p-1$, bag B_{v_j} is linked to $B_{v_{j-1}}$ and $B_{v_{j+1}}$, and to no other bag.

3.4 New Lower Bounds on the Number of Leaves

In this section our goal is to establish a number of lower bounds on the number of leaves. Each of the lower bounds is a linear function of a number of some type of vertices or structures in D . These bounds will help us prove that Rule 7 is correct. Indeed, to this end it suffices to focus on a no-instance and prove that it has at most ck vertices. Hence, if we know that $\text{maxleaf}(D)$ is large when there are many vertices of some kind A, then we know that in our no-instance there are few vertices of kind A. In other words vertices of type A are “easy”. In the next section we will show that because of sparsity arguments the number of the remaining vertices (not corresponding to an “easy type”) is linear in the number of “easy” vertices.

In fact, instead of looking for “easy” vertices in D , we focus of D_c . This is justified by the fact that by Lemma 8 we have $|V(D)| \leq 2|V(D_c)|$, so if we prove that $|V(D_c)| = O(k)$ then also $|V(D)| = O(k)$. Moreover, the following lemma shows that a lower bound on $\text{maxleaf}(D_c)$ imply the same lower bound on $\text{maxleaf}(D)$.

Lemma 13 *Let D be a connected digraph, and let D' be the digraph obtained from D by contracting a cut-edge. Then $\text{maxleaf}(D) \geq \text{maxleaf}(D')$.*

Proof Let (u, v) be the contracted cut-edge and let x be the resulting vertex in D' . Consider any outbranching T' of D' . Then let T be obtained by the following procedure: (1) remove x from T' , (2) add vertices u and v , (3) add edge (u, v) , (4) if p is the parent of x in T' , add edge (p, u) , (5) for every edge $(x, y) \in E(T')$, add (u, y) to T if $(u, y) \in E(D)$ and add (v, y) otherwise. Then clearly T is an outbranching with at least the same number of leaves as T' . Hence it suffices to show that T is a subgraph of D . Otherwise, $(p, u) \notin E(D)$. However, then $(p, v) \in E(D)$. Also, in T' there is a path from the root to p that avoids x . It follows that this path, extended by the edge (p, v) is contained also in D , (u, v) is not a cut-edge, a contradiction. \square

Since all heads of cut-edges have indegree 1, and contraction of lonely cut-edges cannot spoil this property for other cut-edges, we infer that every cut-edge of D remains a cut-edge in the process of obtaining D_c from D by contracting lonely cut-edges one by one. This yields the following.

Corollary 2 $\text{maxleaf}(D) \geq \text{maxleaf}(D_c)$.

A bound on special vertices Daligault and Thomassé [9] show the following lower bound.

Theorem 4 ([9]) *Let D be a 2-connected rooted digraph. Then $\text{maxleaf}(D) \geq |\text{sp}(D)|/30$.*

Unfortunately, D_c is not necessarily 2-connected so we cannot use the above bound. However, we can generalize Theorem 4 as follows.

Theorem 5 *Let D be a connected rooted digraph such that every cut-edge is branching. Then $\text{maxleaf}(D) \geq (|\text{sp}(D)|/30) - \text{cv}(D)$ and $\text{maxleaf}(D) \geq |\text{sp}(D)|/60$.*

To prove Theorem 5, we first need two lemmas and the following definition.

Definitions By *duplicating* a vertex v in a digraph D we mean creating a new digraph D' with $V(D') = V(D) \cup \{v'\}$ and $E(D') = E(D) \cup \{(x, v') : (x, v) \in E(D)\} \cup \{(v', x) : (v, x) \in E(D)\}$.

Lemma 14 *Let D be a digraph, and let D' be the digraph obtained from D by duplicating a vertex v . Then $|\text{sp}(D')| \geq |\text{sp}(D)|$, and $\text{maxleaf}(D) \geq \text{maxleaf}(D') - 1$.*

Proof Every special vertex in D is still a special vertex in D' : duplicating can neither decrease the in-degree of a vertex, nor remove a simple in-edge. Hence $|\text{sp}(D')| \geq |\text{sp}(D)|$. Take a rooted maximum leaf outbranching T' in D' . By symmetry, suppose that v is not a descendant of v' . Let $T = T' \setminus \{v'\} \cup \{(v, w) : (v', w) \in E(T')\}$. Note that T is an outbranching in D . If both v and v' were leaves in T' then v is a leaf in T , so T has one leaf less than T' . Otherwise even if v is not a leaf in T the number of leaves drops by at most one. This finishes the proof. \square

Lemma 15 *In any digraph D such that Rules 1–4 do not apply and every cut-edge is branching we have $\text{maxleaf}(D) \geq \text{cv}(D) + 1$.*

Proof Let T be the spanning tree of D obtained through a Breadth-First-Search started in r . Consider any cut-vertex u . Since u is a cut-vertex, we have $|P(u)| \geq 1$. If $|P(u)| = 1$, let v be the only private neighbor of u . The edge (u, v) is then a lonely cut-edge, a contradiction. Therefore $|P(u)| \geq 2$. By Lemma 6 all the edges from u to $P(u)$ are cut-edges. It follows that every cut-vertex in D has at least two out-neighbors in T . Hence T has at least $\text{cv}(D) + 1$ leaves. \square

We are now ready to prove Theorem 5.

Proof of Theorem 5 Let S be the set of cut-vertices in D . Take the digraph D' obtained from D by duplicating each vertex in S (in any order). We claim that D' is 2-connected. Indeed, assume that D' contains a cut-vertex u ; since a vertex and its duplicate are twins, we can assume that $u \in D$. Since D is a subgraph of D' , it follows that u is a cut-vertex in D . Now D' contains a duplicate u' of u , so every vertex reachable from r in D' is still reachable in $D' - u$, a contradiction. Therefore D' is 2-connected and by Theorem 4 we get $\text{maxleaf}(D') \geq |\text{sp}(D')|/30$. By Lemma 14, we have $\text{maxleaf}(D) \geq \text{maxleaf}(D') - \text{cv}(D)$ and $|\text{sp}(D')| \geq |\text{sp}(D)|$. Thus

$$\text{maxleaf}(D) \geq \frac{|\text{sp}(D)|}{30} - \text{cv}(D). \quad (1)$$

From Lemma 15 we have

$$\text{maxleaf}(D) \geq \text{cv}(D) + 1. \quad (2)$$

The claim now follows from adding (1) and (2). \square

Now it suffices to show that Theorem 5 can be applied to graph D_c .

Lemma 16 *Suppose D is a rooted digraph that is connected. Then for any vertex $u \neq r$ that is not the head of a cut-edge, one can find two simple paths P_1, P_2 from r to u that end with different edges.*

Proof Let R be the set of in-neighbors of u that are reachable from r in $D - u$. Since D is connected we have $R \neq \emptyset$, and if $|R| \geq 2$ then we would be done. Suppose therefore that $R = \{v\}$ for some vertex v such that $(v, u) \in E(D)$. Then (v, u) would be a cut-edge, a contradiction. \square

Lemma 16 will be most often used in the following setting. Suppose that we know that in D the head of every cut-edge has indegree 1. Then if we know that some edge (v, u) is not a cut-edge, then u is not the head of any cut-edge, and hence we can apply Lemma 16 to it.

Lemma 17 *Assume that Rules 1–4 do not apply to D . Let S be the set of lonely cut-edges in D . Consider any subset $S' \subseteq S$. Let D_1 be the graph obtained from D by contracting all edges of S' . Then D_1 does not contain a new cut-edge.*

Proof Induction on $|S'|$. The claim is trivially true for $|S'| = 0$. Assume $|S'| > 0$. Pick any cut-edge $(x, y) \in S'$ and let D_0 be the graph obtained from D by contracting all edges of $S' \setminus \{(x, y)\}$; obviously D_0 is connected. From the induction hypothesis we have that the set of cut-edges of D_0 is a subset of the set of cut-edges of D , and hence from the fact that in D all the heads of cut-edges have indegrees equal to 1, the same conclusion follows for D_0 as well. Hence, whenever in D_0 we conclude that an edge (u, v) is not a cut-edge, then all the edges incoming to v are also not cut-edges. We will show that contracting (x, y) in D_0 does not create a new cut-edge in D_1 .

Assume for a contradiction that (u, v) is a new cut-edge in D_1 , i.e., either $(u, v) \notin E(D_0)$ or $(u, v) \in E(D_0)$ and is not a cut-edge in D_0 . In the former case we have two subcases: contracting (x, y) creates vertex u or v .

Case 1 v is obtained by contracting (x, y) . Then there is an edge (u, x) or (u, y) in D_0 . However, the latter situation is impossible because then an edge enters y in D , a contradiction with Lemma 6. Hence $(u, x) \in D_0$, and in particular $x \neq r$. Edges entering x in D are not cut-edges by Lemma 7, and hence by induction hypothesis no cut-edge enters x in D_0 . By Lemma 16 it follows that in D_0 there are two paths P_1, P_2 from r to x , each entering x via a different edge, say P_1 by (a_1, x) and P_2 by (a_2, x) , with $a_1 \neq a_2$. Note that $a_1 \neq y$ and $a_2 \neq y$ because by Rule 6 we have that $(y, x) \notin E(D)$. By replacing (a_1, x) with (a_1, v) in P_1 and (a_2, x) with (a_2, v) in P_2 we get two paths P'_1 and P'_2 from r to v in D_1 that end with different edges. It follows that (u, v) is not a cut-edge in D_1 , a contradiction.

Case 2 u is obtained by contracting (x, y) . Then D_0 contains (x, v) or (y, v) . No other edge leaving x is a cut-edge in D because (x, y) is lonely in D . Also no edge leaving y is a cut-edge in D by Lemma 7. Hence by induction hypothesis neither (x, v) nor (y, v) can be a cut-edge in D_0 . Since v has an incoming edge that is not a cut-edge in D_0 , as explained before we infer that no edge incoming to v in D_0 is a cut-edge.

From Lemma 16 it follows that in D_0 there are two paths P_1, P_2 from r to v , each entering v via a different edge. If (u, v) is a cut-edge in D_1 , then it means that P_1 ends with (x, v) and P_2 ends with edges $(x, y), (y, v)$, because (x, y) is a cut-edge. If $v \in D$ then Rule 4 would apply to D (with v as x), a contradiction. Otherwise v is obtained by contracting a cut-edge (v_1, v_2) . However, by Lemma 9, no other edge enters v_2 in D , so D contains both edges (x, v_1) and (y, v_1) . Again, we see that Rule 4 applies to D (with v_1 as x), a contradiction.

Case 3 Neither u nor v is obtained by contracting (x, y) . Since (u, v) is not a cut-edge in D_0 , as in the previous case we infer that in fact no edge incoming to v is a cut-edge in D_0 . By Lemma 16, in D_0 there are two paths P_1 and P_2 from r to v , each ending with a different edge. Let us assume that P_1 ends with (a, v) and P_2 ends with (b, v) , for some $a \neq b$. Let P'_1 and P'_2 be the paths in D_1 obtained from P_1 and P_2 by contracting edge (x, y) , and possibly omitting a loop in case both x and y were traversed by P_1 or P_2 . Then P'_1 and P'_2 end with different edges unless $\{x, y\} = \{a, b\}$. By symmetry suppose that $(x, y) = (a, b)$. However, (x, y) is a cut-edge. Hence if $v \in D$ then Rule 4 would apply to D (with v as x), a contradiction. Otherwise v is obtained by contracting a cut-edge (v_1, v_2) , and the same reasoning as in the previous case also gives a contradiction. \square

Lemma 18 *If Rules 1–4 do not apply to D , then in graph D_c all cut-edges are branching.*

Proof By Lemma 17 applied to all lonely cut-edges, in D_c all lonely cut-edges are contracted and no new cut-edges appear. Moreover all the cut-edges that are branching in D are also cut-edges in D_c (since indegrees of their heads are 1), so they are also branching in D_c . This finishes the proof. \square

By Lemma 18 and Corollary 2 we get the following lower bound.

Lemma 19 $\text{maxleaf}(D) \geq |\text{sp}(D_c)|/60$.

A bound on isolated vertices

Definitions We say that a bag B is *special* when v_B is special in D_c . We say that a bag B is *isolated* when B is a non-special bag of size 2 and there is no edge from t_B to a special bag. Vertex $v \in V(D_c)$ is *isolated* if $v = v_B$ for some isolated bag B .

The set of all isolated vertices in D_c is denoted by $\text{iso}(D_c)$.

By *shortcutting* a vertex $v \neq r$ in a digraph D we mean creating a new digraph D' obtained from D by removing v and adding an edge (x, y) for every directed path (x, v, y) in D .

Let D_s be the graph obtained from D by (i) contracting all lonely cut-edges that form a non-isolated bag, and then (ii) shortcutting every tail of an isolated bag. Note that D_s is not necessarily H -minor-free, but we will use it only as an auxiliary construction when establishing a lower bound on $\text{maxleaf}(D)$ in terms of $\text{iso}(D_c)$.

The proof of following lemma can be found in [9] [Lemma 4, point (1)]:

Lemma 20 *Let D be a digraph, and let D' be the digraph obtained from D by shortcutting a cut-vertex v . Then $\text{maxleaf}(D) = \text{maxleaf}(D')$.*

Lemmas 20 and 13 imply the following

$$\text{maxleaf}(D) \geq \text{maxleaf}(D_s). \quad (3)$$

We also observe the following property.

Lemma 21 *Suppose D is a connected rooted digraph where every head of a cut-edge has indegree 1. Let u be a vertex and suppose that $r \notin N^-(u)$ and there is no vertex $v \in N^-(u)$ such that u becomes disconnected from r after removing v . Then after shortcutting u no new cut-edges appear in D .*

Proof Note that the assumption of the lemma implies that in D there is no cut-edge that enters u , so we can apply Lemma 16 to u . Let D and D' denote the graph before and after shortcutting u . Assume that a new cut-edge (x, y) appears in D' .

Case 1 $(x, y) \in E(D)$ and (x, y) is not a cut-edge in D . Since every head of a cut-edge has indegree 1, we infer that no cut-edge enters y . By Lemma 16, in D there are two paths P_1 and P_2 from r to y , ending with different edges e_1 and e_2 . Let P'_1 and P'_2 be

the paths obtained from P_1 and P_2 by shortcutting u . If P'_1 and P'_2 end with the same edges as P_1 and P_2 , then (x, y) is not a cut-edge in D' , a contradiction. Otherwise observe that exactly one of P'_1 and P'_2 has changed the last edge, because otherwise $e_1 = e_2 = (u, y)$. By symmetry assume $e_1 = (u, y)$ and $e_2 = (w, y)$, for some $w \neq u$. Then P'_1 ends with (w, y) and (w, u) is the second last edge of P_1 , or otherwise we are done. By the assumption of the lemma, removal of w does not disconnect u from r , so there is a path Q from r to u that avoids w . If this path traverses y , then some prefix of it is a path from r to y in D' that enters y from a different vertex than w . Otherwise after prolonging Q with (u, y) and shortcutting u we obtain a path from r to y in D' that enters u from a different vertex than w . In both cases we obtained two paths from r to y in D' that end with different edges, which means that no edge incoming to y can be a cut-edge. This is a contradiction with (x, y) being a cut-edge.

Case 2 $(x, y) \notin E(D)$, i.e., (x, y) is obtained by shortcutting u and (x, y) was not present in D . By Lemma 16, in D there are two simple paths P_1 and P_2 from r to u , ending with different edges (a, u) and (b, u) , for some $a \neq b$. If any of these paths traverses y , then some prefix of it is a path in D' from r to y that avoids the new edge (x, y) , due to (x, y) being not present in D . This is a contradiction with (x, y) being a cut-edge. Suppose then that neither P_1 nor P_2 traverses y ; in particular $a \neq y$ and $b \neq y$. Then by replacing (a, u) by (a, y) and (b, u) by (b, y) we get two paths in D' from r to y ending with different edges, so (x, y) is not a cut-edge, a contradiction. \square

Let S be the set containing r and all the special vertices of D_c . Let us invoke Lemma 12 on the set S , and thus obtain a family of maximal weak bipaths P_1, P_2, \dots, P_q with properties as in this lemma.

Consider the process of creating D_s . After contracting all lonely cut-edges corresponding to non-isolated bags, by Lemma 17, no new cut-edges appear. We would like to derive the same conclusion for D_s as well, however we must be careful due to the non-trivial prerequisites of Lemma 21.

Lemma 22 *If the reduction rules do not apply to D then every cut-edge in graph D_s is branching.*

Proof Let D' be the graph after contracting the lonely cut-edges corresponding to non-isolated bags. As argued above, from Lemma 17 it follows that D' has no new cut-edge, i.e., all cut-edges of D' are either original branching cut-edges of D , or original lonely cut-edges of D that correspond to isolated bags.

In D_c , every isolated vertex is some internal vertex on one of the bipaths P_i . We can view the construction of D_s from D' as follows: We iterate through the bipaths P_1, P_2, \dots, P_q one by one. For each of them, we iterate through the internal vertices w of the bipath from left to right, and in D' we shortcut the tail of the bag corresponding to w provided this bag is isolated. We prove now that during this process we maintain the following invariant:

- (1) No new cut-edge has been created, and in particular all the heads of cut-edges in the current digraph have indegree 1.
- (2) For every $v \in D_c$ that is an isolated vertex on some weak bipath, and t_v is the tail of its bag, the following holds: as long as t_v is not yet shortcut, in D' there is

no in-neighbor of t_v which is a cut-vertex whose removal disconnects t_v from the root.

We now show that invariant (2) holds for every such v throughout the process, up to the point when t_v is shortcut. Let us fix v , and suppose $v = v_i$ lies on a maximal weak bipath $P_\alpha = v_1, v_2, \dots, v_p$ in D_c , for some $\alpha \in \{1, 2, \dots, q\}$. For simplicity, denote $B_j = B_{v_j}$. By Lemmas 11 and 12, $v_1, v_p \in S$, $r \notin \{v_1, v_p\}$, and $v_1 \neq v_p$. Note that vertices v_1, v_p are already present in D' . Let W be the set of (a) all vertices of D' that are contained in bags B_i , for $i = 2, \dots, p-1$, and (b) all vertices v_i , for $i = 1, 2, \dots, p$, for which $v_i \in D'$.

We now claim that in D' there is a path Q_1 from r to v_1 that avoids the vertices of W . Indeed, if v_1 was disconnected from r in $D' - W$, then any path from r to v_1 would need to use the unique edge from v_p to B_{p-1} (or v_{p-1}), so this edge would be a cut-edge in D' . This is a contradiction, because this edge was not a cut-edge in D , since cut-edges of D not residing in one bag must be branching and the head of each branching cut-edge in D is special in D_c . Similarly, there is a path Q_2 from r to v_p that avoids W .

From Lemmas 9 and 10 it follows that in D' there is a path R_1 from v_1 to t_v that traverses consecutive bags $B_1, B_2, \dots, B_{i-1}, B_i$ (possibly contracted when constructing D'), and in each it visits either only the tail, or first the tail and then the head. Similarly, there is a path R_2 from v_p to t_v that traverses consecutive bags $B_p, B_{p-1}, \dots, B_{i+1}, B_i$ (possibly contracted when constructing D'), and in each it visits either only the tail, or first the tail and then the head. In particular, since $v_1 \neq v_p$, we have that R_1 and R_2 are vertex-disjoint apart from the last vertex t_v .

Let M_1 be the concatenation of Q_1 and R_1 , and similarly define M_2 . We now examine what happens with paths M_1 and M_2 during the process of obtaining D_s from D' . Every shortcutting of a vertex gives rise to a natural transformation of simple paths in D' , where the traversal of the shortcut vertex is replaced by the usage of a newly introduced edge. Observe that the prefix Q_1 can only get shortcut during the process, and a similar observation holds for the prefix Q_2 . However, v_1 and v_p are not being shortcut. Finally, the internal vertices of both R_1 and R_2 also can get shortcut, but we maintain the invariant that these suffixes remain vertex-disjoint.

Concluding, during the process of obtaining D_s from D' , M_1 and M_2 are always two paths from r to t_v , and their suffixes beginning from v_1 and v_p are always vertex-disjoint apart from the last vertex. Moreover, the vertices appearing before v_1 on M_1 cannot become the in-neighbors of t_v during the shortcutting process due to not belonging to $W \cup \{v_1, v_p\}$, and the symmetrical claim holds for M_2 as well. We conclude that at any moment of the process, the removal of any in-neighbor of t_v cannot affect both paths M_1 and M_2 at the same time.

Hence invariant (2) holds throughout the process. Invariants (1) and (2) are exactly the prerequisites of Lemma 21 when applied to shortcutting t_v . Hence, by iteratively applying Lemma 21 we conclude that no new cut-edge appears in D_s , and in particular every application shows that invariant (1) is maintained in the next step. Therefore, the cut-edges of D_s are simply the branching cut-edges of the original digraph D . \square

Motivated by Lemma 22 and Eq. (3) we are going to show that if there are many isolated vertices in D_c , then there are many special vertices in D_s , which, together with

Theorem 5, implies the desired lower bound. Note that every non-special (in particular, every isolated) vertex in D_c is an internal vertex of some weak bipath P_i , and hence a non-special bag is linked to exactly two other bags—neighbors on the bipath.

Let us recall that, when B is a bag of v we denote $v_B = v$ and $B_v = B$. If there is exactly one head and exactly one tail of B , then they are denoted by h_B and t_B , respectively.

Lemma 23 *Assume reduction rules do not apply to D . Suppose bag A is isolated. Then h_A is special in D_s or there is a non-special bag B linked to A such that h_B , or v_B if B gets contracted, is special in D_s .*

Proof Since A is isolated, there is some bipath $P_i = v_1, v_2, \dots, v_p$ such that $v_A = v_a$ for some $2 \leq a \leq p - 1$. Denote $B_i = B_{v_i}$. Since Rule 2 does not apply, we infer that $d^+(t_A) \geq 2$. One of these edges goes to h_A , whereas the second needs to go to one of the two neighboring bags on P , because A is isolated. By symmetry, suppose that there is an edge from t_A to $B = B_{a+1}$. Of course, B is linked to A and B is not special, because there is an edge from t_A to B and A is isolated. We consider two cases regarding the size of B .

Case 1 $|B| = 2$. We will show that at least one of h_A, h_B is special in D_s . Let us assume that h_B is not special in D_s . By Lemma 9, the edge from A to B is (t_A, t_B) . Then by Rule 5, $(t_B, t_A) \notin E(D)$. By Lemma 9 it follows that $(h_B, t_A) \in E(D)$. By Lemma 10, (t_A, t_B) and (h_B, t_A) are the only edges between A and B .

Let t be the minimum index $i < a$ such that B_{i+1}, \dots, B_a are all isolated and there is an edge from t_{B_j} to $t_{B_{j+1}}$ for each $i < j < a$. By the minimality of t and Lemma 10, it follows that either B_t is not isolated or there is an edge from h_{B_t} to $t_{B_{t+1}}$. In either case, D_s has an edge e incoming to h_B (or v_B , if B gets contracted) from a vertex corresponding to bag B_t (i.e., either from v_{B_t} or h_{B_t}). If in D_s there is no edge from h_B (or v_B) to a vertex that corresponds to B_t , then h_B (v_B) is special in D_s , and we are done. So assume that there is such an edge. It means that in D there must be an edge from t_A to $t_{B_{a-1}}$. Then $t = a - 1$, because otherwise Rule 5 would apply. By Lemma 10 in D there is no edge from h_A to B_{a-1} . We argued earlier that there is also no edge in D from h_A to $B_{a+1} = B$. It follows that $(h_A, h_B) \notin E(D_s)$. However, after shortcutting A we get $(h_B, h_A) \in E(D_s)$. Hence, h_A is special in D_s .

Case 2 $|B| = 1$. By Lemmas 9 and 10, the only edges between A and B are (t_A, t_B) and (t_B, t_A) . Then D_s contains edge (t_B, h_A) . If $(h_A, t_B) \notin D_s$, then h_A is special in D_s and we are done. Otherwise, denoting $C = B_{a-1}$, it must hold that C is isolated (so in particular non-special) and there must be edges $(h_A, t_C), (t_C, t_A)$ in D . Then by the same argument as in Case 1 (with C playing the role of A and A playing the role of B), h_A or h_C is special in D_s . \square

Lemma 24 *If the reduction rules do not apply to D then $\maxleaf(D) \geq \frac{|\text{iso}(D_c)|}{180}$.*

Proof By Lemmas 13 and 20, $\maxleaf(D) \geq \maxleaf(D_s)$. By Lemma 22 and Theorem 5 we get $\maxleaf(D_s) \geq |\text{sp}(D_s)|/60$. By Lemma 23, to every isolated bag A we can assign a non-special bag B , such that h_B is special in D_s and either $B = A$ or B is linked to A . By the definition, there are at most two bags linked to a non-special bag (corresponding to the neighbors of v_B on a weak bipath in D_c). It follows

that $|\text{sp}(D_s)| \geq |\text{iso}(D_c)|/3$. Together with the previous inequalities this implies $\text{maxleaf}(D) \geq |\text{iso}(D_c)|/180$. \square

Definitions We will say that a vertex v of D_c is *easy* when $v = r$, or v is special, or v is isolated in D_c . A vertex that is not easy is called *hard*. We now invoke once more Lemma 12, but this time instead of S we take the set of all the easy vertices. Every maximal bipath obtained in this decomposition will be called a *maximal hard bipath*. In other words, a weak bipath in D_c is *hard* if all its internal vertices are hard. The sets of all easy and hard vertices in D_c are denoted by $\text{ea}(D_c)$ and $\text{hd}(D_c)$, respectively. For any maximal hard bipath P' in D_c we define $\text{Out}(P') = N_{D_c}^+(V(P') \setminus \{u, v\})$, where u and v are the extremities of P' .

A bound on slaves

Definitions For every pair of easy vertices $u, v \in \text{ea}(D_c)$ and a subset $S \subseteq V(D_c)$ with $\{u, v\} \subseteq S$, if there is a hard bipath P' between u and v such that $\text{Out}(P') = S$, we choose arbitrarily two such paths (or one, if only one exists) and we call them *masters*, while all the remaining hard bipaths P'' between u and v with $\text{Out}(P'') = S$ are called *slaves* of respective masters, or just *slaves*. The number of all slaves in D_c is denoted by $\text{sl}(D_c)$.

Lemma 25 $\text{maxleaf}(D) \geq \text{sl}(D_c)$.

Proof By Lemma 13 it suffices to show that $\text{maxleaf}(D_c) \geq \text{sl}(D_c)$. We will show that in fact every outbranching T of D_c has at least $\text{sl}(D_c)$ leaves.

Fix an arbitrary outbranching T of D_c . It is easy to see that in any outbranching T , the number of leaves is equal to $1 + \sum_{u \in V(T)} \max(\deg_T^+(u) - 1, 0)$. Consider a slave $Z = v_1, \dots, v_\ell$ with $\text{Out}(Z) = S$ and extremities $v_1, v_\ell \in S$, and let M_1, M_2 be its masters. Then either $(v_1, v_2) \in E(T)$, or $(v_\ell, v_{\ell-1}) \in E(T)$. Let slave Z charge vertex v_1 in the former case, and charge vertex v_ℓ in the latter case. Also on M_1 and M_2 at least one edge outgoing from v_1 and one edge outgoing from v_ℓ is present in T . We conclude that the total contribution to the outdegrees in T of v_1 and v_ℓ from M_1, M_2 and their slaves is at least the number of times v_1 and v_ℓ are charged by the slaves of M_1, M_2 , plus 2 for M_1 and M_2 .

Let $X \subseteq \text{ea}(D_c)$ be the set of easy vertices that are the extremities of some slave. Then $1 + \sum_{u \in V(T)} \max(\deg_T^+(u) - 1, 0) \geq (\sum_{u \in X} \deg_T^+(u)) - |X|$. On the other hand, from what we argued in the previous paragraph it follows that $\sum_{u \in X} \deg_T^+(u) \geq \text{sl}(D_c) + 2|F|$, where F is the set of equivalence classes of slaves partitioned according to their masters. However, since every bipath has two extremities, it follows that $|X| \leq 2|F|$. Hence $\sum_{u \in X} \deg_T^+(u) - |X| \geq \text{sl}(D_c)$ and T has at least $\text{sl}(D_c)$ leaves. \square

3.5 The Size Bound

In this section we prove the following theorem which imply the correctness of Rule 7.

Theorem 6 *Let H be a graph. Let D be an H -minor-free digraph such that Rules 1–6 do not apply. If $\text{maxleaf}(D) < k$, then $|V(D)| = 2^{O(|H|\sqrt{\log |H|})} k$.*

Throughout the section we assume that rules 1–6 do not apply to D . The results from the previous section give a bound of $O(k)$ on the number of easy vertices. Our plan in this section is to show a linear bound on the number of hard vertices in terms of $|ea(D_c)| + sl(D_c)$ and next get a bound on $|V(D)|$ as a corollary.

It follows that our task is to show that the total length of hard weak bipaths in D_c is not too large. Let us state a few useful properties of such bipaths.

Lemma 26 *Let $\ell \geq 9$ and let $P' = v_1, \dots, v_\ell$ be a hard bipath in D_c such that v_1 and v_ℓ are easy. For every $i = 3, \dots, \ell - 6$ there is at least one edge in D from t_{B_x} , for some $x = v_j$ and $j = i, \dots, i + 4$, to a vertex outside $\bigcup_{j'=2}^{\ell-1} B_{v_{j'}}$.*

Proof Fix $i \in \{3, \dots, \ell - 6\}$ and consider the length 4 bipath v_i, \dots, v_{i+4} . For convenience denote $B_j = B_{v_j}$. If for some $j = i + 1, i + 2, i + 3$ there is an edge from B_j with head not in $B_{j-1} \cup B_{j+1}$, then by Lemma 12 (iii) this head is outside $\bigcup_{j'=2}^{\ell-1} B_{v_{j'}}$, and we are done. Hence the edges leaving B_{i+1}, B_{i+2} , and B_{i+3} go only to the neighboring bags. Since Rule 3 does not apply, for some $j = i, \dots, i + 4$ the bag B_j is of size 2. Since v_j is hard, B_j is not isolated. Hence, there is an edge e in D from t_{B_j} to a special bag B . Since $v_2, \dots, v_{\ell-1}$ are hard, B is none of $B_2, \dots, B_{\ell-1}$. \square

Lemma 27 *For any maximal hard weak bipath P' in D_c , we have $|hd(D_c) \cap V(P')| \leq 10|Out(P')| + 6$.*

Proof Let $P' = v_1, \dots, v_\ell$. We can assume that $\ell \geq 9$, for otherwise $|hd(D_c) \cap V(P')| \leq 6$ and the claim holds trivially. For convenience denote $B_i = B_{v_i}$. By Lemma 26 there are at least $\lfloor (\ell - 4)/5 \rfloor$ edges from tails of bags $B_3, \dots, B_{\ell-2}$ to vertices outside $\bigcup_{i=2}^{\ell-1} B_{v_i}$. Let Z denote the set of these edges. We claim that for every vertex $u \in V(D)$ there are at most two edges from Z with heads in u . Indeed, assume that u has got three in-neighbors $t_{B_a}, t_{B_b}, t_{B_c}$ in D , with $a < b < c$. Then $N^-(u) \setminus \{t_{B_b}\}$ cuts t_{B_b} (and all vertices of B_{a+1}, \dots, B_{c-1}) from r , a contradiction to the fact that D is reduced with respect to Rule 4. Hence the edges in Z have at least $\lfloor (\ell - 4)/5 \rfloor / 2 \geq ((\ell - 8)/5)/2$ different heads. By Lemma 9 these heads are tails of bags, and by Lemma 8 each of them corresponds to a different vertex in D_c . It follows that the vertices $v_3, \dots, v_{\ell-2}$ have in D_c at least $(\ell - 8)/10$ neighbors in $Out(P')$, so $|Out(P')| \geq (\ell - 8)/10$. Since $|hd(D_c) \cap V(P)| = \ell - 2$ it follows that $|hd(D_c) \cap V(P)| \leq 10|Out(P')| + 6$. \square

In what follows we are going to bound the size of D_c using its sparsity properties.

Definitions To this end we use an auxiliary bipartite graph G , called the *bipath minor* of D_c , constructed as follows. We put $V(G) = A \cup B$, where $A = ea(D_c)$, and B is the set of all maximal hard bipaths in D_c . For every maximal hard bipath P' in D_c with extremities $u, v \in ea(D_c)$, the neighborhood of the corresponding vertex in B is exactly $Out(P')$.

Lemma 28 *Let H be a graph. If D is H -minor-free, then $|hd(D_c)| = 2^{O(|H|\sqrt{\log |H|})} (|ea(D_c)| + sl(D_c))$.*

Proof Consider an arbitrary hard vertex v of D_c . Consider the maximal hard weak bipath P' in D_c that contains v . Then P' corresponds to a vertex in B and by Lemma 27, it has at most $10|\text{Out}(P')| + 6$ internal vertices. It follows that

$$|\text{hd}(D_c)| \leq \sum_{v \in B} (10 \deg_G(v) + 6) \leq \sum_{v \in B} 16 \deg_G(v). \quad (4)$$

Note that G is a minor of (the undirected version of) D since it can be obtained from D_c by edge contractions and deletions, and D_c in turn is obtained from D by contractions. Hence, G is H -minor-free. Moreover, G is simple. By Lemma 1, we know that G is d_H -degenerate, for $d_H = O(|H|\sqrt{\log |H|})$. Let B_m and B_s denote the vertices in B for which the corresponding maximal hard bipath is master and slave, respectively. By (4) we get

$$\begin{aligned} |\text{hd}(D_c)| &\leq 16 \sum_{v \in B} \deg_G(v) \\ &\leq 16 \sum_{\substack{v \in B \\ \deg_G(v) > 2d_H}} \deg_G(v) + 16 \sum_{\substack{v \in B_s \\ \deg_G(v) \leq 2d_H}} \deg_G(v) + 16 \sum_{\substack{v \in B_m \\ \deg_G(v) \leq 2d_H}} \deg_G(v). \end{aligned}$$

Let us bound each of the terms separately. By Lemma 2, we have

$$\sum_{\substack{v \in B_s \\ \deg_G(v) > 2d_H}} d(v) \leq 2d_H |A| = O(|H|\sqrt{\log |H|} \cdot |\text{ea}(D_c)|).$$

Obviously,

$$\sum_{\substack{v \in B_s \\ \deg_G(v) \leq 2d_H}} \deg_G(v) \leq 2d_H \text{sl}(D_c) = O(|H|\sqrt{\log |H|} \text{sl}(D_c)).$$

Finally,

$$\begin{aligned} \sum_{\substack{v \in B_m \\ \deg_G(v) \leq 2d_H}} \deg_G(v) &= \sum_{\substack{S \subseteq A \\ |S| \leq 2d_H}} |S| \cdot |\{v \in B_m : N_G(v) = S\}| \\ &\leq 2d_H \sum_{\substack{S \subseteq A \\ |S| \leq 2d_H}} |\{v \in B_m : N_G(v) = S\}|. \end{aligned}$$

By Corollary 1, there is a constant $c_H = 2^{O(|H|\sqrt{\log |H|})}$ such that there are at most $c_H |A|$ distinct neighborhoods of vertices in B . For each such neighborhood $S \subseteq A$ and for every pair of vertices $u, v \in S$ there are at most two master bipaths P' with endpoints u and v and such that $\text{Out}(P') = S$. Therefore for a fixed neighborhood S of size at most $2d_H$ we have $|\{v \in B_m : N_G(v) = S\}| \leq 2^{\binom{|S|}{2}} \leq 2^{\binom{2d_H}{2}} = O(d_H^2)$. Hence

$$\sum_{S \subseteq A, |S| \leq 2d_H} |\{v \in B_m : N_G(v) = S\}| = O(c_H \cdot d_H^2 \cdot |\text{ea}(D_c)|) \\ = 2^{O(|H| \sqrt{\log |H|})} |\text{ea}(D_c)|.$$

The claim follows. \square

Now we can finish the proof of Theorem 6. Assume $\text{maxleaf}(D) < k$. By Lemmas 19 and 24, $\text{ea}(D_c) < 60k + 180k$. Moreover, by Lemma 25, $\text{sl}(D_c) < k$. This, with Lemma 28 gives the claim of Theorem 6.

4 k -INTERNAL OUT-BRANCHING in Graphs of Bounded Expansion

In this section we give a linear kernel for IOB on any graph class \mathcal{G} of bounded expansion. To this end, we modify the approach of Gutin et al. [23]. Before we proceed to the argumentation, let us remark that Gutin et al. work with a slightly more general problem, where the root of the outbranching is not prescribed; of course, the outbranching is still required to span the whole vertex set. Note that the variant with a prescribed root r can be reduced to this variant simply by removing all in-arcs of r , which forces r to be the root of any outbranching of the given digraph. Since our kernel will be an induced subgraph of D and r will not be removed by any reduction, it will be still true that r is the only candidate for the root of an outbranching. Hence, the resulting instance will be equivalent in both variants. Therefore, from now on we work with variant without prescribed root in order to be able to use the observations of Gutin et al. as black-boxes.

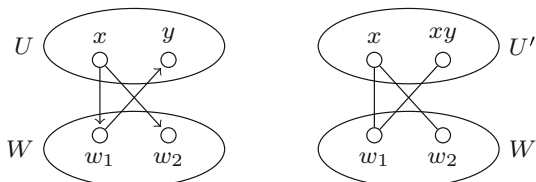
First, Gutin et al. observe that in an instance that cannot be easily resolved, one can find a small vertex cover (of the underlying undirected graph).

Lemma 29 ([23]) *Given a digraph D , we can either build an outbranching with at least k internal vertices or obtain a vertex cover of size at most $2k - 2$ in $O(n^2 m)$ time.*

For a given directed graph D and a vertex cover U in D we build an undirected bipartite graph $B_{D,U}$ as follows (see Fig. 3). Let $W = V(D) \setminus U$. Then,

$$V(B) = U' \cup W, \text{ where } U' = N^-(W) \cup (U \times U); \\ E(B) = \{\{xy, w\} : xy \in U \times U, w \in W, (x, w) \in E(D), (w, y) \in E(D)\} \cup \\ \{\{x, w\} : x \in U, w \in W, (x, w) \in E(D)\}.$$

Fig. 3 The digraph D with vertex cover U (left) and the corresponding graph $G_{D,U}$ (right)



Definitions A *crown decomposition* of an undirected graph G is a partitioning of $V(G)$ into three parts C , H and R , such that

- C is an independent set.
- There are no edges between vertices of C and R . That is, H separates C and R .
- C can be partitioned into $C_m \cup C_u$ with $|C_m| = |H|$, such that $G[C_m \cup H]$ contains a perfect matching that matches each vertex of C_m with a vertex of H .

Crown decompositions are used in many kernelization algorithms. In particular, the following lemma shows that in certain situations a crown decomposition can be found efficiently. (For more context see [7, 16].)

Lemma 30 (see [23]) *Suppose G is an undirected graph on n vertices, and suppose I is an independent set in G such that $|I| \geq 2n/3$. Then G admits a crown decomposition $(C = C_u \uplus C_m, H, R)$ with $C \subseteq I$, $H \subseteq V(G) \setminus I$ and $C_u \neq \emptyset$. Moreover, given I , the decomposition $(C = C_u \uplus C_m, H, R)$ can be found in $O(nm)$ time.*

The main idea of Gutin et al. is to search for crowns in $B_{D,U}$ with $C \subseteq W$ and $C_u \neq \emptyset$. Such crowns can be conveniently reduced using the following reduction rule, whose correctness is proved in Lemma 4.4 of [23].

Rule 8 Let U be a vertex cover in D and let $W = V(D) \setminus U$. Assume there is a crown decomposition $(C = C_m \cup C_u, H, R)$ in $B_{D,U}$ with $C \subseteq W$ and $C_u \neq \emptyset$. Then remove C_u from D .

Our idea is to combine Rule 8 with the knowledge that D belongs to a graph class of bounded expansion \mathcal{G} , and hence Proposition 1 can be used to reason about the sparseness of the adjacency structure between U and W . Let us introduce some notation.

Definitions Consider a vertex cover U and an independent set $W = V(D) \setminus U$ in D . Let $W_s = \{w \in W : \deg_D(w) < 2\nabla_0(\mathcal{G})\}$, and let $W_b = W \setminus W_s$. Moreover, for $N \subseteq U$ with $|N| < 2\nabla_0(\mathcal{G})$, let $W_N = \{w \in W_s : N(w) = N\}$. Let $\mathcal{N}(U) = \{N \subseteq U : |N| < 2\nabla_0(\mathcal{G}), W_N \neq \emptyset\}$. Note that $|\mathcal{N}(U)| \leq |W_s|$.

Our kernelization algorithm is as follows.

1. If the algorithm from Lemma 29 returns an outbranching, answer YES and terminate; otherwise it returns a vertex cover U of size at most $2k-2$. Let $W = V(D) \setminus U$.
2. Construct the graph $B := B_{D,U}$ and compute W_s , $\mathcal{N}(U)$, and nonempty sets W_N .
3. If there is a set $N \in \mathcal{N}(U)$ such that $|W_N| > 2|N_B(W_N)|$, then apply Lemma 30 to graph $B[N_B[W_N]]$ with $I = W_N$. This gives us a crown decomposition $(C = C_u \uplus C_m, H, R)$ of $B[N_B[W_N]]$ with $C \subseteq W_N$, $H \subseteq N_B(W_N)$, and $C_u \neq \emptyset$. Observe that $(C = C_u \uplus C_m, H, R \cup (V(B) \setminus N_B[W_N]))$ is a crown decomposition of B . Apply Rule 8 to this crown decomposition in order to remove C_u from D , and restart the algorithm in the reduced graph.
4. Otherwise, return D .

In case we have a prescribed root r of the outbranching that we would like to preserve in the kernelization process, we can add it to the constructed vertex cover U ,

thus increasing its size up to at most $2k - 1$. The reduction rules never remove any vertex of U . Note that the time of one iteration of the algorithm is $O(n^2m)$, hence the total time of the algorithm is bounded by $O(n^3m)$.

Given this algorithm, we can restate and prove our main result for IOB.

Proof The correctness of our kernelization algorithm and a polynomial bound on its running time follows from Lemmas 29 and 30. Note that the kernelization algorithm never decrements the budget k , so it suffices to show that it outputs an instance (D, k) such that $|V(D)| = O(k)$.

We can assume that the algorithm constructed a vertex cover U of D of size at most $2k - 2$ ($2k - 1$ if we want to preserve a prescribed root), because otherwise the algorithm would terminate and provide a positive answer. Let $W = V(D) \setminus U$. Then $V(D) = U \cup W_s \cup W_b$. By the first claim of Proposition 1 we get $|W_b| \leq 2\nabla_0(G)|U| \leq 4\nabla_0(\mathcal{G})k$. Hence it suffices to bound the size of W_s . Note that $W_s = \bigcup_{N \in \mathcal{N}(U)} W_N$. By the second claim of Proposition 1 we get $|\mathcal{N}(U)| \leq (4^{\nabla_1(G)} + 2\nabla_1(G))|U| = O(4^{\nabla_1(\mathcal{G})}k)$. However, since Step 3 of the kernelization algorithm cannot be applied, for every $N \in \mathcal{N}(U)$ we have $|W_N| \leq 2|N_{B_{D,U}}(W_N)|$. However, by the construction of $B_{D,U}$ it is clear that $|N_{B_{D,U}}(W_N)| \leq |N|^2 + |N| < 4\nabla_0(\mathcal{G})^2 + 2\nabla_0(\mathcal{G})$, and hence $|W_N| < 8\nabla_0(\mathcal{G})^2 + 4\nabla_0(\mathcal{G})$. It follows that $|W_s| = \sum_{N \in \mathcal{N}(U)} |W_N| = O(4^{\nabla_1(\mathcal{G})}\nabla_0(\mathcal{G})^2k)$, and hence $|V(D)| = |U| + |W_s| + |W_b| = O(4^{\nabla_1(\mathcal{G})}\nabla_0(\mathcal{G})^2k)$. This finishes the proof. \square

Let us remark that in the proof of Theorem 2 we used only the boundedness of $\nabla_0(\mathcal{G})$ and $\nabla_1(\mathcal{G})$, so our algorithm works as well in any graph class where only these two grads are finite constants. Also, the kernelization algorithm has polynomial running time, where the degree of the polynomial is a constant independent of \mathcal{G} .

5 Subexponential Algorithms

Theorems 1 and 2 enable us to design subexponential parameterized algorithms for LOB and IOB on H -minor-free graphs using the standard approach via treewidth. To this end, we compose two facts: First, for a fixed forbidden minor H , every H -minor-free graph on n vertices has treewidth at most $O(\sqrt{n})$ [22]. Second, both k -LEAF OUT-BRANCHING and k -INTERNAL OUT-BRANCHING can be solved in time $2^{O(t)} \cdot n^{O(1)}$ on n -vertex graphs given together with their tree decompositions of width at most t , as explained next.

For the latter ingredient, a standard approach to dynamic programming on tree decompositions would yield algorithms with running time $2^{O(t \log t)} \cdot n^{O(1)}$, since we consider all possible partitions of a bag in the states of the dynamic programming table. However, both problems are amenable to recently developed new techniques for constructing dynamic programming algorithms with running time $2^{O(t)} \cdot n^{O(1)}$. An application of the Cut&Count technique [8] immediately yields randomized algorithms with such a running time for both these problems. Actually, the existence of such algorithms also follows from expressibility in the logical formalism ECML+C proposed by the third author [28, 29], which provides a meta-result on applicability of Cut&Count. The full paper [29], Appendix D, contains a formula for the problem of finding an outbranching with exactly k leaves, which can be trivially adjusted

to express both k -LEAF OUT- BRANCHING and k -INTERNAL OUT- BRANCHING as below. The Cut&Count technique has been recently derandomized by Bodlaender et al. [3], who proposed the so-called *rank based approach* that yields deterministic $2^{O(t)} \cdot n^{O(1)}$ -time algorithms for many problems amenable to Cut&Count. It is a simple exercise to see that using this technique one can also design such algorithms for k -LEAF OUT- BRANCHING and k -INTERNAL OUT- BRANCHING. Thus, we have the following proposition.

Proposition 2 *k -LEAF OUT- BRANCHING and k -INTERNAL OUT- BRANCHING can be solved in deterministic time $2^{O(t)} \cdot n^{O(1)}$ on an n -vertex graph given together with its tree decomposition of width t .*

Gathering all the tools, we obtain the subexponential algorithms promised in Sect. 1.

Proof Let (D, k) be the input instance of LOB or IOB, where D is H -minor free. First, we apply the kernelization algorithm of Theorem 1 or 2 (depending on the problem) to reduce the size of the instance to $O(k)$; note that the application of neither of these algorithms can increase the parameter. Having the reduced instance (D', k') in hand, where $k' \leq k$, D' is H -minor-free, and $|V(D')| = O(k)$, we infer that the treewidth of D' is in $O(\sqrt{k})$. Hence we apply any constant-factor approximation algorithm for treewidth, e.g., [4], to compute a tree decomposition of D' of width $O(\sqrt{k})$ in time $2^{O(\sqrt{k})}$. We conclude by applying the appropriate algorithm of Proposition 2; this application also takes time $2^{O(\sqrt{k})}$. \square

We remark that the running time of the algorithms given by Theorem 3 is essentially optimal under the assumption of the Exponential Time Hypothesis (ETH), even already on planar directed graphs. More precisely, from the known NP-hardness reductions it follows that the existence of an algorithm for LOB or IOB working in time $2^{o(\sqrt{N})}$ on a planar directed graph with N vertices would contradict ETH. For completeness, we sketch now how this conclusion can be derived.

Theorem 7 *Unless ETH fails, there is no algorithm solving LOB or IOB that achieves running time $2^{o(\sqrt{N})}$ on planar directed graphs with N vertices.*

Proof For IOB the statement follows easily from the known fact that the existence of such an algorithm for PLANAR HAMILTONIAN CYCLE would contradict ETH, see e.g., [25]. First, PLANAR HAMILTONIAN CYCLE can be Turing-reduced to PLANAR HAMILTONIAN PATH by guessing an edge used in the solution and replacing it with two pendant vertices attached to its endpoints. PLANAR HAMILTONIAN PATH can be now reduced to the variant of IOB on planar digraphs, where the root is not specified: simply replace every undirected edge by two directed edges with opposite orientations, and ask for an outbranching with at least $N - 1$ internal vertices. The variant with unspecified root is easily Turing-reducible to the one with specified root by simply guessing the root. Note that all the aforementioned reductions increase the instance size by at most a constant factor, and hence the statement for IOB follows.

We turn our attention to LOB. First, it is known that the PLANAR VERTEX COVER problem does not admit an algorithm with running time $2^{o(\sqrt{N})}$ on planar graphs

with N vertices, see e.g., [17,25]. Garey and Johnson [21] proposed a reduction from PLANAR VERTEX COVER to PLANAR CONNECTED VERTEX COVER that increases the number of vertices of the graph only by a constant multiplicative factor. This proves that also for PLANAR CONNECTED VERTEX COVER an algorithm with running time $2^{o(\sqrt{N})}$ can be excluded under ETH.

We further reduce PLANAR CONNECTED VERTEX COVER to PLANAR CONNECTED DOMINATING SET using the following transformation: subdivide every edge of the graph, and add a pendant to every introduced subdividing vertex. It can be easily shown that a planar graph G has a connected vertex cover of size k if and only if the planar graph G' obtained in this transformation has a connected dominating set of size $k + |E(G)|$. Hence, under ETH there is no $2^{o(\sqrt{N})}$ -time algorithm for PLANAR CONNECTED DOMINATING SET.

Now, we use the known fact that the CONNECTED DOMINATING SET is dual to the MAX LEAF problem—the problem of finding a spanning tree of an undirected graph with the maximum possible number of leaves. More precisely, a graph G has a connected dominating set of size at most k if and only if it admits a spanning tree with at least $|V(G)| - k$ leaves; cf. [14]. Hence, under ETH there is no $2^{o(\sqrt{N})}$ -time algorithm for PLANAR MAX LEAF.

Finally, PLANAR MAX LEAF can be reduced to the variant of LOB on planar graphs where the root is not specified by just replacing every undirected edge by two directed edges with opposite orientations. Again, the variant with unspecified root is easily Turing-reducible to the one with specified root by simply guessing the root. This proves the statement for LOB. \square

Below we list the auxiliary lower bounds that are showed in the proof of Theorem 7 for the ease of future referencing. (Some, or all, of them are known and should be treated as folklore.)

Proposition 3 *Unless ETH fails, for each of the following problems*

- PLANAR HAMILTONIAN PATH,
- PLANAR CONNECTED VERTEX COVER,
- PLANAR CONNECTED DOMINATING SET,
- PLANAR MAX LEAF

there is no algorithm that achieves running time $2^{o(\sqrt{N})}$ on undirected planar graphs with N vertices.

6 Concluding Remarks

In this paper we have shown linear kernels for both k -LEAF OUT-BRANCHING and k -INTERNAL OUT-BRANCHING on sparse graph classes: H -minor-free and of bounded expansion, respectively. We believe that our work is another good example of how abstract properties derived from the sparsity of the considered graph class, in particular the ones expressed in Proposition 1, can be used in the kernelization setting for a clean treatment of graph classes with excluded minors, without the need of invoking the

decomposition theorem of Robertson and Seymour. Other examples of this approach include [15, 20], and we hope that even more will appear in future.

In the light of our results, the question about the existence of linear kernels for k -LEAF OUT-BRANCHING and k -INTERNAL OUT-BRANCHING on general graphs becomes even more tantalizing. We do not intend to take a stance about the actual answer, but after investigating both problems for some time we believe that in both cases a conceptual breakthrough is needed to make an improvement.

Acknowledgements The authors are very grateful to Marcin Pilipczuk for reading the manuscript carefully and providing useful comments.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Alber, J., Fellows, M.R., Niedermeier, R.: Polynomial-time data reduction for dominating set. *J. ACM* **51**(3), 363–384 (2004)
2. Binkele-Raible, D., Fernau, H., Fomin, F.V., Lokshantov, D., Saurabh, S., Villanger, Y.: Kernel(s) for problems with no kernel: on out-trees with many leaves. *ACM Trans. Algorithms* **8**(4), 38 (2012)
3. Bodlaender, H.L., Cygan, M., Kratsch, S., Nederlof, J.: Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. In: *ICALP 2013. Lecture Notes in Computer Science*, vol. 7965, pp. 196–207. Springer, Berlin (2013)
4. Bodlaender, H.L., Drange, P.G., Dregi, M.S., Fomin, F.V., Lokshantov, D., Pilipczuk, M.: An $O(c^{kn})$ 5-approximation algorithm for treewidth. In: *FOCS 2013*, pp. 499–508. IEEE Computer Society (2013)
5. Bodlaender, H.L., Fomin, F.V., Lokshantov, D., Penninkx, E., Saurabh, S., Thilikos, D.M.: (Meta) kernelization. In: *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25–27, 2009, Atlanta, Georgia, USA*, pp. 629–638. IEEE Computer Society (2009)
6. Bonamy, M., Kowalik, L., Pilipczuk, M., Socala, A.: Linear kernels for outbranching problems in sparse digraphs. In: *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16–18, 2015, Patras, Greece, Volume 43 of LIPIcs*, pp. 199–211. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2015)
7. Chor, B., Fellows, M., Juedes, D.W.: Linear kernels in linear time, or how to save k colors in $o(n^2)$ steps. In: *Graph-Theoretic Concepts in Computer Science, 30th International Workshop, WG 2004, Bad Honnef, Germany, June 21–23, 2004, Revised Papers*, pp. 257–269 (2004)
8. Cygan, M., Nederlof, J., Pilipczuk, M., Pilipczuk, M., van Rooij, J.M.M., Wojtaszczyk, J.O.: Solving connectivity problems parameterized by treewidth in single exponential time. In: *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22–25, 2011*, pp. 150–159. IEEE Computer Society (2011)
9. Daligault, J., Thomassé, S.: On finding directed trees with many leaves. In: Chen, J., Fomin, F.V. (eds.) *Parameterized and Exact Computation*, pp. 86–97. Springer, Berlin (2009)
10. Dankelmann, P., Gutin, G., Kim, E.J.: On complexity of minimum leaf out-branching problem. *Discrete Appl. Math.* **157**(13), 3000–3004 (2009)
11. Demaine, E.D., Fomin, F.V., Hajiaghayi, M.T., Thilikos, D.M.: Subexponential parameterized algorithms on bounded-genus graphs and h -minor-free graphs. *J. ACM* **52**(6), 866–893 (2005)
12. Demaine, E.D., Hajiaghayi, M., Kawarabayashi, K.: Contraction decomposition in h -minor-free graphs and algorithmic applications. In: *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6–8 June 2011*, pp. 441–450. ACM (2011)
13. Dorn, F., Fomin, F.V., Lokshantov, D., Raman, V., Saurabh, S.: Beyond bidimensionality: parameterized subexponential algorithms on directed graphs. *Inf. Comput.* **233**, 60–70 (2013)
14. Douglas, R.J.: NP-completeness and degree restricted spanning trees. *Discrete Math.* **105**(1–3), 41–47 (1992)

15. Drange, P.G., Dregi, M.S., Fomin, F.V., Kreutzer, S., Lokshtanov, D., Pilipczuk, M., Pilipczuk, M., Reidl, F., Villaamil, F.S., Saurabh, S., Siebertz, S., Sikdar, S.: Kernelization and sparseness: the case of dominating set. In: 33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016, February 17–20, 2016, Orléans, France, Volume 47 of LIPIcs, pp. 31:1–31:14. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2016)
16. Fellows, M., Heggernes, P., Rosamond, F.A., Sloper, C., Telle, J.A.: Finding k disjoint triangles in an arbitrary graph. In: Graph-Theoretic Concepts in Computer Science, 30th International Workshop, WG 2004, Bad Honnef, Germany, June 21–23, 2004, Revised Papers, Volume 3353 of Lecture Notes in Computer Science, pp. 235–244. Springer (2004)
17. Flum, J., Grohe, M.: Parameterized Complexity Theory. Texts in Theoretical Computer Science. Springer, Berlin (2006)
18. Fomin, F.V., Lokshtanov, D., Raman, V., Saurabh, S.: Subexponential algorithms for partial cover problems. *Inf. Process. Lett.* **111**(16), 814–818 (2011)
19. Fomin, F.V., Lokshtanov, D., Saurabh, S., Thilikos, D.M.: Bidimensionality and kernels. In: SODA, pp. 503–510 (2010)
20. Gajarský, J., Hliněný, P., Obdržálek, J., Ordyniak, S., Reidl, F., Rossmanith, P., Villaamil, F.S., Sikdar, S.: Kernelization using structural parameters on sparse graph classes. In: Bodlaender, H.L., Italiano, G.F. (eds.), ESA 2013, Volume 8125 of Lecture Notes in Computer Science, pp. 529–540. Springer (2013)
21. Garey, M.R., Johnson, D.S.: The rectilinear Steiner tree problem in NP-complete. *SIAM J. Appl. Math.* **32**, 826–834 (1977)
22. Grohe, M.: Local tree-width, excluded minors, approximation algorithms. *Combinatorica* **23**(4), 613–632 (2003)
23. Gutin, G., Razgon, I., Kim, E.J.: Minimum leaf out-branching and related problems. *Theor. Comput. Sci.* **410**(45), 4571–4579 (2009)
24. Klein, P.N., Marx, D.: A subexponential parameterized algorithm for subset TSP on planar graphs. In: Chekuri, C. (ed.), Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5–7, 2014, pp. 1812–1830. SIAM (2014)
25. Lokshtanov, D., Marx, D., Saurabh, S.: Lower bounds based on the exponential time hypothesis. *Bull. EATCS* **105**, 41–72 (2011)
26. Lokshtanov, D., Saurabh, S., Wahlström, M.: Subexponential parameterized odd cycle transversal on planar graphs. In: D'Souza, D., Kavitha, T., Radhakrishnan, J. (eds.), IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012, December 15–17, 2012, Hyderabad, India, Volume 18 of LIPIcs, pp. 424–434. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2012)
27. Nešetřil, J., Ossona de Mendez, P.: Sparsity: Graphs, Structures, and Algorithms, Volume 28 of Algorithms and Combinatorics. Springer, Berlin (2012)
28. Pilipczuk, M.: Problems parameterized by treewidth tractable in single exponential time: a logical approach. In: MFCS 2011, Volume 6907 of Lecture Notes in Computer Science, pp. 520–531. Springer (2011)
29. Pilipczuk, M.: Problems parameterized by treewidth tractable in single exponential time: a logical approach. *CoRR* (2011) [arXiv:abs/1104.3057](https://arxiv.org/abs/1104.3057)
30. Pilipczuk, M., Pilipczuk, M., Sankowski, P., van Leeuwen, E.J.: Subexponential-time parameterized algorithm for Steiner tree on planar graphs. In: Portier, N., Wilke, T. (eds.), 30th International Symposium on Theoretical Aspects of Computer Science, STACS 2013, February 27–March 2, 2013, Kiel, Germany, Volume 20 of LIPIcs, pp. 353–364. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2013)
31. Pilipczuk, M., Pilipczuk, M., Sankowski, P., van Leeuwen, E.J.: Network sparsification for Steiner problems on planar and bounded-genus graphs. In: 55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18–21, 2014, pp. 276–285. IEEE Computer Society (2014)