Adapting the Directed Grid Theorem into an FPT Algorithm

Victor Campos, Raul Lopes, Ana Karolinna Maia, Ignasi Sau

ADAPTING THE DIRECTED GRID THEOREM INTO AN FPT ALGORITHM*

VICTOR CAMPOS†, RAUL LOPES†‡, ANA KAROLINNA MAIA†, AND IGNASI SAU‡

Abstract. The Grid Theorem of Robertson and Seymour [JCTB, 1986] is one of the most important tools in the field of structural graph theory, finding numerous applications in the design of algorithms for undirected graphs. An analogous version of the Grid Theorem in digraphs was conjectured by Johnson et al. [JCTB, 2001], and proved by Kawarabayashi and Kreutzer [STOC, 2015]. Namely, they showed that there is a function $f(k)$ such that every digraph of directed tree-width at least $f(k)$ contains a cylindrical grid of order k as a butterfly minor, and stated that their proof can be turned into an XP algorithm, with parameter k, that either constructs a decomposition of the appropriate width, or finds the claimed large cylindrical grid as a butterfly minor. In this paper, we adapt some of the steps of the proof of Kawarabayashi and Kreutzer to improve this XP algorithm into an FPT algorithm. Towards this, our main technical contributions are two FPT algorithms with parameter k. The first one either produces an arboreal decomposition of width $3k-2$ or finds a haven of order k in a digraph D, improving on the original result for arboreal decompositions by Johnson et al. [JCTB, 2001]. The second algorithm finds a well-linked set of order k in a digraph D of large directed tree-width. As tools to prove these results, we show how to solve a generalized version of the problem of finding balanced separators for a given set of vertices T in FPT time with parameter $|T|$, a result that we consider to be of its own interest.

Key words. Digraph, directed tree-width, grid theorem, FPT algorithm.

AMS subject classifications. 05C20, 05C85, 68R10

1. Introduction.

Width parameters can be seen as an estimation of how close a given graph is to a typical structure. For example, the *tree-width* of a graph, a parameter of particular interest in the literature, measures how tightly a graph can be approximated by a tree. Namely, a *tree decomposition* of a graph G with bounded tree-width shows how one can place the vertices of the original graph into "bags" of bounded size which, in turn, can be arranged as the vertices of a tree T such that the intersection between adjacent bags in T are separators in G. Thus, a tree decomposition exposes a form of global connectivity measure for graphs: as only a bounded number of vertices can be placed in each bag, many small separators can be identified through the decomposition. The tree-width of graphs was first introduced by Bertele and Brioschi [6], then again by Halin [33], and finally reintroduced by Robertson and Seymour [49]. For a survey on the subject, we refer the reader to [8].

A number of hard problems can be efficiently solved in graphs of bounded tree-width, either by making use of classical algorithmic techniques like dynamic programming, or by making use of Courcelle's Theorem [18]. Applications of algorithms based on tree decompositions range from frequency allocation problems to the Traveling Salesman problem [17, 40].

Given the enormous success achieved by applications based on width parameters

*Submitted to the editors March 9, 2023. An extended abstract of this paper appeared in the *Proc. of the X Latin and American Algorithms, Graphs and Optimization Symposium (**LAGOS**), volume 346 of ENTCS, pages 229–240, Belo Horizonte, Brazil, June **2019**.*

Funding: Work supported by projects CAPES/STIC-AmSud 88881.197438/2018-01, CNPq - Universal project 425297/2016-0, FUNCAP - PRONEM PNE-011200061.01.00/16, DEMOGRAPH (ANR-16-CE40-0028), ESIGMA (ANR-17-CE23-0010), ELIT (ANR-20-CE48-0008-01), and UTMA (ANR-20-CE92-0027).

†ParGO group, Universidade Federal do Ceará, Fortaleza, Brazil (campos@lia.ufc.br, karol-maia@ufc.br, raul@alu.ufc.br).

‡LIRMM, Université de Montpellier, CNRS, Montpellier, France (ignasi.sau@lirmm.fr).

in undirected graphs, it is no surprise that there is interest in finding similar definitions for digraphs. Johnson et al. [35] proposed an analogous measure for tree-width in the directed case. The *directed tree-width* of a digraph measures its distance to being a directed acyclic graph (DAG for short), and an *arboreal decomposition* exposes a (strong) connectivity measure of a digraph. Reed [48] provided an intuitive exposition of the similarities between the undirected and directed cases.

Similarly to the undirected case, some hard problems become tractable when restricted to digraphs of bounded directed tree-width. For example, Johnson et al. [35] showed that the Directed k-Disjoint Paths problem, which Fortune et al. [29] showed to be NP-hard even for $k = 2$ in general digraphs, is solvable in polynomial (more precisely, in XP) time in digraphs of directed tree-width bounded by a constant. A similar approach given in [35] can be applied to the Hamilton Path and Hamilton Cycle problems, Hamilton Path with Prescribed Ends, and others. It is worth mentioning that Slivkins [54] proved that the Directed k-Disjoint Paths problem is W[1]-hard even when restricted to DAGs. As DAGs have directed tree-width zero, there is little hope for the existence of a fixed-parameter tractable (FPT for short) algorithm for the Directed k-Disjoint Paths problem in digraphs of bounded directed tree-width. As another example of application, a Courcelle-like theorem for directed tree-width was proved by de Oliveira Oliveira [21], but running in XP time.

It is natural to ask what can be said of a graph with large tree-width. One of the most relevant results in structural graph theory states that undirected graphs with large tree-width contain large grid minors. More precisely, the Grid Theorem by Robertson and Seymour [49] states that there is a function $f : \mathbb{N} \to \mathbb{N}$ such that every graph of tree-width at least $f(k)$ contains a $(k \times k)$-grid as a minor. Recently, Chekuri and Chuzhoy [13] gave a polynomial bound on the function $f(k)$, which was further improved by Chuzhoy and Tan [16].

Sometimes, large tree-width (and therefore, the existence of a large grid minor) implies that we are actually working with a positive instance of a particular problem. In this direction, Demaine et al. [22] presented a framework that generates FPT algorithms for many such problems, known as *bidimensional* problems. This list includes Vertex Cover, Feedback Vertex Set, Longest Path, Minimum Maximal Matching, Dominating Set, Edge Dominating Set, and many others. This seminal work is currently known as *Bidimensionality* [28].

Another application of the Grid Theorem is in the *irrelevant vertex* technique, introduced by Robertson and Seymour [50–52] to solve the k-Disjoint Paths problem. The goal is to show that every instance whose input graph violates a set of conditions contains a vertex that is "irrelevant", that is, a vertex whose removal generates an equivalent instance of the problem. This leads to an iterative algorithm, reducing the problem to a smaller instance, until it satisfies sufficient conditions for its tractability. This technique was used to solve the k-Disjoint Paths problem in FPT time with parameter k, and a number of other problems (cf. for instance [32, 39]). For the directed case, Cygan et al. [20] used a similar technique to provide an FPT algorithm for the Directed k-Disjoint Paths problem in planar digraphs.

A result analogous to the Grid Theorem for digraphs was conjectured by Johnson et al. [35] and Reed [48], and recently proved by Kawarabayashi and Kreutzer [38][1], after having proved it for digraphs with forbidden minors [37][2]. Namely, it is shown

[1]The full version of [38] is available at https://arxiv.org/abs/1411.5681v2.

[2]In an unpublished manuscript from 2001 [36], Johnson, Robertson, Seymour and Thomas gave a proof of this result for planar digraphs.

in [38] that there is a function $f : \mathbb{N} \to \mathbb{N}$ such that every digraph of directed tree-width at least $f(k)$ contains a *cylindrical grid* (see Fig. 1) of order k as a *butterfly minor*; all the definitions are given formally in Section 2. Recently, Hatzel et al. [34] proved that the function $f(k)$ can be made polynomial in *planar* digraphs.

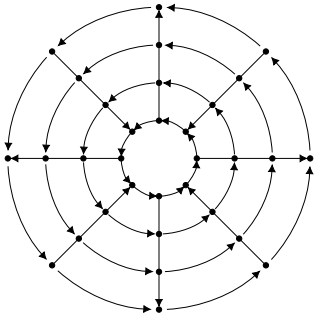

Fig. 1: A cylindrical grid of order $k = 4$.

The Directed Grid Theorem has found many applications. For instance, Amiri et al. [1] proved that a strongly connected digraph H has the Erdős-Pósa property if and only if H is a butterfly minor of some cylindrical grid of sufficiently large order. Additionally, the authors showed that for every fixed strongly connected digraph H satisfying those conditions and every fixed integer k, there is a polynomial-time algorithm that either finds k disjoint (butterfly or topological) models of H in a digraph D or a set $X \subseteq V(D)$ of size bounded by a function of k such that $D \setminus X$ does not contain a model of H.

Edwards et al. [25] applied some results used in the proof of the Directed Grid Theorem [38] to provide an XP algorithm with parameter k for a relaxed version of the Directed Disjoint Paths problem, in which every vertex of the input digraph is allowed to occur in at most two paths of a solution, when restricted to $(36k^3 + 2k)$-strongly connected digraphs. Kawarabayashi and Kreutzer [38] mentioned that the Directed Grid Theorem can be used to provide, for fixed k, an algorithm running in polynomial time that, given a digraph D and k terminal pairs $(s_1, t_1), \ldots, (s_k, t_k)$, either finds a collection of paths $P_1, \ldots, P_k$ such that P_i is a path from s_i to t_i in D and every vertex of D occurs in at most four paths of the collection, or concludes that D does not contain a collection of pairwise disjoint paths $P_1, \ldots, P_k$ such that P_i is a path from s_i to t_i in D, for $i \in [k]$. Although Chekuri et al. [14] could not use the Directed Grid Theorem since the bound on $f(k)$ (mentioned above) is larger than required, they build on the ideas used in [36] to produce their own version of the Directed Grid Theorem for planar digraphs.

The proof of the Directed Grid Theorem by Kawarabayashi and Kreutzer [38] is constructive. Namely, the authors start with an algorithm by Johnson et al. [35, 3.3] that, given a digraph D and an integer parameter k, outputs, in XP time, either an arboreal decomposition of D of width at most $3k - 2$ or a haven of order k (see Definition 2.9). Thus, if D has directed tree-width at least $3k-1$, they obtain a haven of order k. From this haven, they obtain a bramble $\mathcal{B}$ of order k and size $|V(D)|^{\mathcal{O}(k)}$. Finally, from $\mathcal{B}$ they find a path P containing a well-linked set A (see Definition 2.8) of size roughly $\sqrt{k}$ in XP time with parameter k.

We remark that the bound on the running time of those algorithms depends on

the size of $\mathcal{B}$ since, in general, one must test whether $X \cap V(B) \neq \emptyset$ for each $B \in \mathcal{B}$ to check whether a given set $X \subseteq V(D)$ is a hitting set of $\mathcal{B}$. The remainder of the proof of the Directed Grid Theorem [38] runs in FPT time, with parameter k.

Our approach, results, and techniques. By making local changes to the proofs by Johnson et al. [35] and Kawarabayashi and Kreutzer [38], we show that there is an FPT algorithm that, given a digraph D and an integer k, either constructs an arboreal decomposition of D of width at most $3k-2$, or finds a path P in D containing a well-linked set A of size roughly $\sqrt{2k}$. Our results and the remainder of the proof of the Directed Grid Theorem [38] yield an FPT algorithm that either constructs an arboreal decomposition of width at most $f(k)$ or a cylindrical grid of order k as a butterfly minor of D. For completeness, we provide in Section 2.5 an overview of how, starting from the path P and the well-linked set A found by our FPT algorithm, the proof of Kawarabayashi and Kreutzer [38] yields an algorithm to find the desired cylindrical grid in FPT time. We would like to insist on the fact that the proof of our main result is based on performing local changes to the proof of Kawarabayashi and Kreutzer given in the available full version of [38]. In what follows we detail our results and techniques, along with the organization of the article.

In Section 2 we give all the necessary definitions and preliminaries, and we formally state the two main contributions of this paper, namely Theorem 2.19 and Theorem 2.23. As discussed above, in Section 2.5 we sketch how these two results, combined with the remainder of the original proof in [38], yield the FPT algorithm stated in Corollary 2.24.

Similarly to the undirected case (see, for example, [27, Chapter 11]), the result by Johnson et al. [35, 3.3] shows that the size of a special kind of vertex separator of some set $T \subseteq V(D)$ is intrinsically connected to the directed tree-width of D. Their algorithm runs a subroutine that, given a set of vertices T with $|T| \leq 2k-1$, searches for a set $Z \subseteq V(D)$ with $|Z| \leq k-1$ such that every strong component of $D \setminus Z$ intersects at most half of the vertices of T, or decides that none exists. Such a set Z is known as a *T-balanced separator* [3]. If every such search is successful, the algorithm produces an arboreal decomposition of width at most $3k-2$. If the search fails for some set T, then we say that T is *$(k-1)$-linked* [3] and use it to construct a haven of order k (we show how to do this construction in Lemma 2.18). In Section 3, we give an FPT algorithm (Theorem 2.17) that, given a digraph D and a parameter k, outputs either an arboreal decomposition of D of width at most $3k-2$ or a $(k-1)$-linked set T with $|T| = 2k-1$, thus improving the result by Johnson et al. [35], since we can easily extract a haven of order k from T (Lemma 2.18), and proving our first main contribution (Theorem 2.19).

We acknowledge that a sketch of a proof of a similar result, with approximation factor of $5k+10$, is given in [3, Theorem 9.4.4]. In their proof, the authors mention how to compute a weaker version of T-balanced separators in FPT time with parameter $|T|$, and the increase on the approximation factor they guarantee is a consequence of this relaxation. For our approximation algorithm for directed tree-width, we introduce generalized versions of balanced separators and k-linked sets that are also used in Section 4. Namely, we say that a set Z is a *(T,r)-balanced separator* if every strong component of $D \setminus Z$ intersects at most r vertices of T and that T is *(k,r)-linked* if every (T,r)-balanced separator has size at least $k+1$ (see Definition 2.16).

In Theorem 3.5 we show that the problem of finding a (T,r)-balanced separator of size s or deciding that T is (s,r)-linked is FPT with relation to the parameter $|T|$. We refer to this problem as Balanced Separator and, to solve it, we make

use of an algorithm by Erbacher et al. [26] for a variation of the Multicut problem for digraphs, named as Multicut With Linearly Ordered Terminals by the authors.

Next, we prove our second main contribution (Theorem 2.23). For this, we need to find a bramble $\mathcal{B}$ when the second output of the algorithm for approximate arboreal decompositions (the set T) is obtained, and use it to find a path P containing a well-linked set $A \subseteq V(P)$ of size roughly $\sqrt{2k}$. In order to prove Theorem 2.23, we proceed as follows.

In Section 4.1 we show how to construct, from a $(k-1)$-linked set T with $|T| = 2k-1$, a bramble $\mathcal{B}_T$ that is easier to work with than the general case in a number of ways. We characterize hitting sets of $\mathcal{B}_T$ by T-balanced separators (Lemma 4.2) and thus applying our algorithm for Balanced Separator, we conclude that we can decide if the order of $\mathcal{B}_T$ is at most s in FPT time. In fact, we prove a slightly stronger result stating that the same can be done for some particular choices of subsets ("sub-brambles") of $\mathcal{B}_T$. This is a considerable improvement on the running time of the naive approach to find hitting sets of brambles, which involves going through every element of the bramble. In particular, our characterization of hitting sets of $\mathcal{B}_T$ allows us to test if given a set $X \subseteq V(D)$ is a hitting set of $\mathcal{B}_T$ in polynomial time by enumerating the strong components of $D \setminus X$. This is an easy observation that also holds for the bramble used in the proof of the Directed Grid Theorem [38].

In Section 4.2 we show how to find P and A. To find P, we iteratively grow a path until it is a hitting set of $\mathcal{B}_T$, at each time adding one vertex and testing if the current set of vertices of the growing path is a hitting set of $\mathcal{B}_T$ (Lemma 4.7). To find A, we produce an ordered sequence of subpaths of P each being a hitting set of a "sub-bramble" of $\mathcal{B}_T$ of adequate order, and pick the vertices of A from the vertices between those subpaths (Lemma 4.9). The key ingredient of the first procedure is the fact that we can decide if a given set of vertices X is a hitting set of $\mathcal{B}_T$ in polynomial time, as a consequence of the characterization given by Lemma 4.2. For the second procedure, we iteratively use our algorithm for Balanced Separator (Theorem 3.5) to test if a bramble that is formed by a particular subset of $\mathcal{B}_T$ has adequate order. Thus, in contrast with what is done in [38][3], our version of the first procedure runs in polynomial time and our version of the second procedure runs in FPT time, assuming that $\mathcal{B}_T$ and T are given in both cases. In order to prove Theorem 2.23, we introduce the notion of "(i)-split" (see Definition 4.8), and we prove several lemmas about (i)-splits, namely Lemma 4.9 and Lemma 4.10.

A roadmap of the aforementioned algorithm is given in Fig. 2. We mark by a dashed arc the steps of [38] which are already FPT and do not need to be adapted. All others arcs represent steps that we adapt in this paper.

We conclude the article in Section 5 with some remarks and potential algorithmic applications of our results.

2. Formal definitions and preliminaries.

In this section we give the definitions relevant to this paper, mention some known results, and present a more detailed discussion of our main contributions.

2.1. Graphs and digraphs.

We refer the reader to [10] for basic background on graph theory, and recall here only some basic definitions. For a graph $G = (V, E)$, directed or not, and a set $X \subseteq V(G)$, we write $G \setminus X$ for the graph resulting from the deletion of X from G. If e is an edge of a directed or undirected graph with *endpoints*

[3] Specifically, in Lemmas 4.3 and 4.4 of the full version.

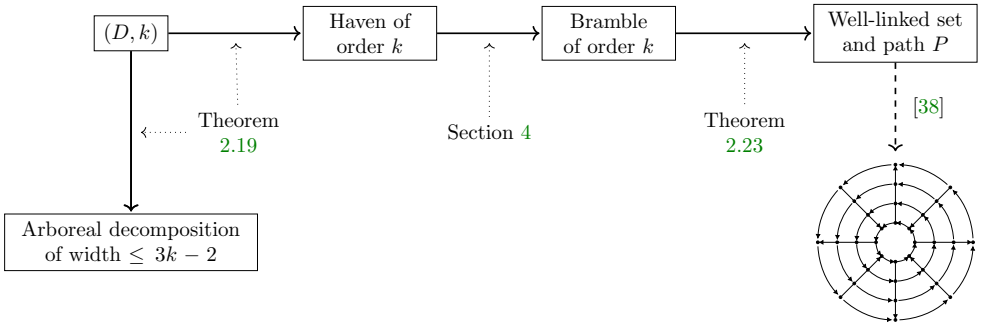

Fig. 2: Sketch of the algorithm used in the proof of the Directed Grid Theorem [38].

u and v, we may refer to e as (u, v) and say that e is *incident* to u and v. If e is an edge from u to v of a directed graph, we say that e has *tail* u, *head* v, and is *oriented* from u to v. We also allow for loops and multiple edges.

The *in-degree* (resp. *out-degree*) of a vertex v in a digraph D is the number of edges with head (resp. tail) v. The *in-neighborhood* $N_D^-(v)$ of v is the set $\{u \in V(D) \mid (u, v) \in E(G)\}$, and the *out-neighborhood* $N_D^+(v)$ is the set $\{u \in V(D) \mid (v, u) \in E(G)\}$. We say that u is an *in-neighbor* of v if $u \in N_D^-(v)$ and that u is an *out-neighbor* of v if $u \in N_D^+(v)$.

A *walk* in a digraph D is an alternating sequence W of vertices and edges that starts and ends with a vertex, and such that for every edge (u, v) in the walk, vertex u (resp. vertex v) is the element right before (resp. right after) edge (u, v) in W. If the first vertex in a walk is u and the last one is v, then we say this is a *walk from u to v*. A *path* is a digraph containing exactly a walk that contains all of its vertices and edges without repetition. If P is a path with $V(P) = \{v_1, \ldots, v_k\}$ and $E(P) = \{(v_i, v_{i+1}) \mid i \in [k-1]\}$, we say that v_1 is the *first* vertex of P, that v_k is the *last* vertex of P, and for $i \in [k-1]$ we say that v_{i+1} is the *sucessor in P* of v_i. All paths mentioned henceforth, unless stated otherwise, are considered to be directed.

An *orientation* of an undirected graph G is a digraph D obtained from G by choosing an orientation for each edge $e \in E(G)$. The undirected graph G formed by ignoring the orientation of the edges of a digraph D is the *underlying graph* of D.

A digraph D is *strongly connected* if, for every pair of vertices $u, v \in V(D)$, there is a walk from u to v and a walk from v to u in D. We say that D is *weakly connected* if the underlying graph of D is connected. A *separator* of D is a set $S \subsetneq V(D)$ such that $D \setminus S$ is not strongly connected. If $|V(D)| \geq k+1$ and k is the minimum size of a separator of D, we say that D is *k-strongly connected*. A *strong component* of D is a maximal induced subdigraph of D that is strongly connected, and a *weak component* of D is a maximal induced subdigraph of D that is weakly connected.

For a positive integer k, we denote by $[k]$ the set containing every integer i such that $1 \leq i \leq k$.

2.2. Parameterized complexity. We refer the reader to [19,24] for basic background on parameterized complexity, and we recall here only the definitions used in this article. A *parameterized problem* is a language $L \subseteq \Sigma^* \times \mathbb{N}$. For an instance $I = (x, k) \in \Sigma^* \times \mathbb{N}$, k is called the *parameter*.

A parameterized problem L is *fixed-parameter tractable* (FPT) if there exists an algorithm $\mathcal{A}$, a computable function f, and a constant c such that given an instance

$I = (x, k)$, $\mathcal{A}$ (called an FPT *algorithm*) correctly decides whether $I \in L$ in time bounded by $f(k) \cdot |I|^c$. For instance, the Vertex Cover problem parameterized by the size of the solution is FPT.

A parameterized problem L is in XP if there exists an algorithm $\mathcal{A}$ and two computable functions f and g such that given an instance $I = (x, k)$, $\mathcal{A}$ (called an XP *algorithm*) correctly decides whether $I \in L$ in time bounded by $f(k) \cdot |I|^{g(k)}$. For instance, the Clique problem parameterized by the size of the solution is in XP.

Within parameterized problems, the class W[1] may be seen as the parameterized equivalent to the class NP of classical decision problems. Without entering into details (see [19, 24] for the formal definitions), a parameterized problem being W[1]-*hard* can be seen as a strong evidence that this problem is *not* FPT. The canonical example of W[1]-hard problem is Clique parameterized by the size of the solution.

2.3. Arboreal decompositions and obstructions. By an *arborescence* R with *root* r_0, we mean an orientation of a tree such that R contains a path from r_0 to every other vertex of the tree. If a vertex v of R has out-degree zero, we say that v is a *leaf* of R. We now define guarded sets and arboreal decompositions of digraphs. From here on, we refer to oriented edges only, unless stated otherwise. D will always stand for a digraph, and G for an undirected graph. Unless stated otherwise, we define $n = |V(D)|$ and $m = |E(D)|$ when D is the input digraph of some algorithm.

For $X, Y \subseteq V(D)$, an (X, Y)-*separator* is a set of vertices S such that there are no paths in $D \setminus S$ from any vertex in X to any vertex in Y. We make use Menger's Theorem [46] for digraphs.

Theorem 2.1 (Menger's Theorem [46]). *Let D be a digraph and $X, Y \subseteq V(D)$. Then the minimum size of an (X, Y)-separator in D equals the maximum number of pairwise internally vertex-disjoint paths from X to Y in D.*

Definition 2.2 (Z-guarded sets). *Let D be a digraph, $Z \subseteq V(D)$, and $S \subseteq V(D) \setminus Z$. We say that S is* Z-guarded *if there is no directed walk in $D \setminus Z$ with first and last vertices in S that uses a vertex of $D \setminus (Z \cup S)$.*

That is, informally speaking, a set S is Z-guarded if whenever a walk starting in S leaves S, it is impossible to come back to S without visiting a vertex in Z. See Fig. 3 for an illustration of a Z-guarded set. If a set S is Z-guarded, we may also say that Z

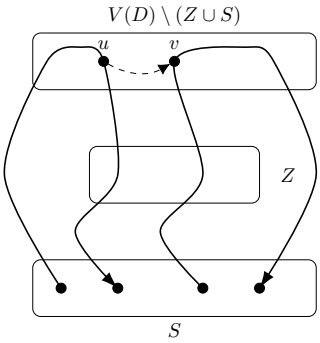

Fig. 3: A Z-guarded set S. The dashed line indicates that there is no path from u to v in $V(D) \setminus (Z \cup S)$.

is a *guard* for S. We remark that in [35], the authors use the terminology of Z-*normal* sets instead of Z-guarded sets.

Let R be an arborescence, $r \in V(R)$, $e \in E(R)$, and r' be the head of e. We say that $r > e$ if there is a path from r' to r in R. We also say that $e \sim r$ if r is the head or the tail of e. To define the tree-width of directed graphs, we first need to introduce arboreal decompositions.

DEFINITION 2.3 (Arboreal decomposition). *An* arboreal decomposition β *of a digraph* D *is a triple* $(R, \mathcal{X}, \mathcal{W})$ *where* R *is an arborescence,* $\mathcal{X} = \{X_e : e \in E(R)\}$, $\mathcal{W} = \{W_r : r \in V(R)\}$, *and* $\mathcal{X}, \mathcal{W}$ *are collections of sets of vertices of* D *(called* bags*) such that*

(i) $\mathcal{W}$ *is a partition of* $V(D)$ *into non-empty sets, and*

(ii) *if* $e \in E(R)$, *then* $\bigcup\{W_r : r \in V(R)$ *and* $r > e\}$ *is* X_e*-guarded.*

We also say that r *is a* leaf *of* $(R, \mathcal{X}, \mathcal{W})$ *if* r *has out-degree zero in* R.

The left hand side of Fig. 4 contains an example of a digraph D, while the right hand side shows an arboreal decomposition for it. In the illustration of the arboreal decomposition, squares are guards X_e and circles are bags of vertices W_r. For example, consider the edge $e \in E(R)$ with $X_e = \{b, c\}$ from the bag W_1 to the bag W_2. Then $\bigcup\{W_r : r \in V(R)$ and $r > e\} = V(D) \setminus \{a\}$ and, by item **(ii)** described above, this set must be $\{b, c\}$-guarded since $X_e = \{b, c\}$. In other words, there cannot be a walk in $D \setminus \{b, c\}$ starting and ending in $V(D) \setminus \{a\}$ using a vertex of $\{a\}$. This is true in D since every path reaching $\{a\}$ from the remaining of the graph must do so through vertices b or c. The reader is encouraged to verify the same properties for the other guards in the decomposition.

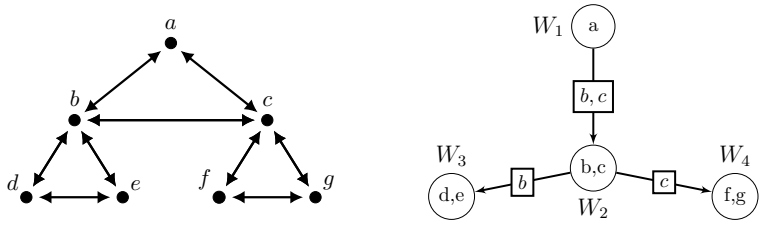

Fig. 4: A digraph D and an arboreal decomposition of D of width two. A bidirectional edge is used to represent a pair of edges in both directions.

DEFINITION 2.4 (Nice arboreal decompositions). *We say that an arboreal decomposition* $(R, \mathcal{X}, \mathcal{W})$ *of a digraph* D *is* nice *if*

(iii) *for every* $e \in E(R)$, $\bigcup\{W_r : r \in V(R), r > e\}$ *induces a strong component of* $D \setminus X_e$, *and*

(iv) *if* $r \in V(R)$ *and* $r_1, \ldots, r_\ell$ *are the out-neighbors of* r *in* R, *then*

$$\left(\bigcup_{1 \le i \le \ell} W_{r_i}\right) \cap \left(\bigcup_{e \sim r} X_e\right) = \emptyset.$$

DEFINITION 2.5 (Directed tree-width). *Let* $(R, \mathcal{X}, \mathcal{W})$ *be an arboreal decomposition of a digraph* D. *For a vertex* $r \in V(R)$, *we denote by* $\mathsf{width}(r)$ *the size of the set* $W_r \cup (\bigcup_{e \sim r} X_e)$. *The* width *of* $(R, \mathcal{X}, \mathcal{W})$ *is the least integer* k *such that, for all*

$r \in V(R)$, $\mathsf{width}(r) \leq k+1$. *The* directed tree-width *of D, denoted by* $\mathsf{dtw}(D)$*, is the least integer k such that D has an arboreal decomposition of width k.*

We remark that DAGs have directed tree-width zero.

If G is an undirected graph and D the digraph obtained from G by replacing every edge of G with two directed edges in opposite directions then, as shown by Johnson et al. [35], the tree-width of G is equal to the directed tree-width of D. Thus, deciding if a digraph D has directed tree-width at most k, for a given integer k, is NP-complete since deciding if the tree-width of an undirected graph is at most k is an NP-complete problem [2].

We now formally define cylindrical grids, butterfly contractions, butterfly minors, and some blocking structures for large directed tree-width.

Definition 2.6 (Cylindrical grid). *A* cylindrical grid of order k *is a digraph formed by the union of k disjoint cycles $C_1, \ldots, C_k$ and $2k$ disjoint paths P_1, P_2, …, P_{2k} where*

1. *for $i \in [k], V(C_i) = \{v_{i,1}, v_{i,2}, \ldots, v_{i,2k}\}$ and $E(C_i) = \{(v_{i,j}, v_{i,j+1} \mid j \in [2k-1])\} \cup \{(v_{i,2k}, v_{i,1})\}$,*
2. *for $i \in \{1, 3, \ldots, 2k-1\}$, $E(P_i) = \{(v_{1,i}, v_{2,i}), (v_{2,i}, v_{3,i}), \ldots, (v_{k-1,i}, v_{k,i})\}$, and*
3. *for $i \in \{2, 4, \ldots, 2k\}$, $E(P_i) = \{(v_{k,i}, v_{k-1,i}), (v_{k-1,i}, v_{k-2,i}), \ldots, (v_{2,i}, v_{1,i})\}$.*

In other words, path P_i is oriented from the first circle to the last one if i is odd, and the other way around if i is even. Furthermore, every vertex of a cylindrical grid occurs in the intersection of a path and a cycle. See Fig. 1 for an example of a cylindrical grid of order $k = 4$.

Definition 2.7 (Butterfly contraction and butterfly minors). *Let D be a digraph. An edge e from u to v of D is* butterfly contractible *if e is the only outgoing edge of u or the only incoming edge of v. By* butterfly contracting *e in D, we obtain a digraph D' with vertex set $V(D') = V(D) \setminus \{u, v\} \cup \{x_{u,v}\}$, where $x_{u,v}$ is a new vertex, and $E(D') = E(D) \setminus \{e\}$. Every incidence of an edge $f \in E(D')$ to u or v in D becomes an incidence to $x_{u,v}$ in D'. If D' is generated from a subgraph of D by a series a butterfly contractions, we say that D' is a* butterfly minor *of D.*

Notice that, in the above definition, the newly introduced vertex $x_{u,v}$ has in D' the same neighbors of u and v in D. It is not hard to see that butterfly contractions cannot generate any new paths, and that there is no such guarantee if no restrictions are imposed on which edges of a digraph can be contracted. See Fig. 5 for an example of this.

Fig. 5: Butterfly contractions preserve separations. In each example the dashed edge is contracted to generate the digraph on the right. Edge e_1 is *not* butterfly contractible.

Definition 2.8 (Well-linked sets). *Let D be a digraph and $A \subseteq V(D)$. We say that A is* well-linked *in D if, for all disjoint $X, Y \subseteq A$ with $|X| = |Y|$, there are $|X|$ vertex-disjoint paths from X to Y in D. The* order *of a well-linked set A is $|A|$. We*

denote by $\mathsf{wlink}(D)$ *the size of a largest well-linked set in* D.

DEFINITION 2.9 (Havens in digraphs). *Let* D *be a digraph. A* haven of order k *in* D *is a function* β *assigning to every set* $Z \subseteq V(D)$, *with* $|Z| \leq k-1$, *the vertex set of a strong component of* $D \setminus Z$ *in such way that if* $Z' \subseteq Z \subseteq V(D)$ *then* $\beta(Z) \subseteq \beta(Z')$. *The* haven number *of a digraph* D, *denoted by* $\mathsf{hn}(D)$, *is the maximum* k *such that* D *admits a haven of order* k.

A k-strongly connected digraph, for example, admits a haven of order k: it suffices to choose $\beta(Z) = V(D) \setminus Z$ for any $Z \subseteq V(D)$ with $|Z| \leq k-1$. Fig. 6 illustrates the defining property of havens.

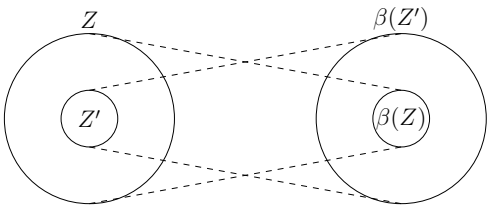

Fig. 6: Illustration of the haven property.

DEFINITION 2.10 (Brambles in digraphs). *A* bramble $\mathcal{B} = \{B_1, \ldots, B_\ell\}$ *in a digraph* D *is a family of strongly connected subgraphs of* D *such that if* $\{B, B'\} \subseteq \mathcal{B}$ *then* $V(B) \cap V(B') \neq \emptyset$ *or there are edges in* D *from* $V(B)$ *to* $V(B')$ *and from* $V(B')$ *to* $V(B)$. *A* hitting set *of a bramble* $\mathcal{B}$ *is a set* $C \subseteq V(D)$ *such that* $C \cap V(B) \neq \emptyset$ *for all* $B \in \mathcal{B}$. *The* order *of a bramble* $\mathcal{B}$, *denoted by* $\mathsf{ord}(\mathcal{B})$, *is the minimum size of a hitting set of* $\mathcal{B}$. *The* bramble number *of a digraph* D, *denoted by* $\mathsf{bn}(D)$, *is the the maximum* k *such that* D *admits a bramble of order* k.

There is a direct relation between the haven number and the tree-width of undirected graphs. A haven in an undirected graph is defined similarly: the function β retains all its properties, but mapping sets of at most $k-1$ vertices to components of the graph resulting from the deletion of those vertices.

PROPOSITION 2.11 (Seymour and Thomas [53]). *Let* G *be an undirected graph and* $k \geq 1$ *be an integer. Then* G *has a haven of order* k *if and only if its tree-width is at least* $k-1$.

For digraphs, only one implication of the previous result is known to be true.

PROPOSITION 2.12 (Johnson et al. [35]). *Let* D *be a digraph and* k *be a non-negative integer. If* D *has a haven of order* k, *then* $\mathsf{dtw}(D) \geq k-1$.

For the reverse direction of Proposition 2.12, only an approximate version is known.

PROPOSITION 2.13 (Johnson et al. [35]). *Let* D *be a digraph and* k *be a positive integer. If* $\mathsf{dtw}(D) \geq 3k-1$ *then* D *admits a haven of order* k.

Finally, the following two lemmas show that brambles of large order and large well-linked sets are obstructions to small directed tree-width. The proof of the first lemma can be done by converting brambles into havens and back. For the second lemma, it is sufficient to show that any minimum hitting set of a bramble of order k is well-linked and to extract a bramble of order k from a well-linked set of order $4k+1$. The proofs are simple and can be found, for example, in [45, Chapter 6].

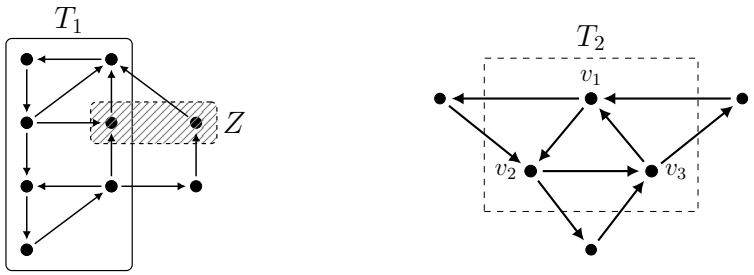

Fig. 7: Examples of balanced separators. On the left, Z is a $(T_1, 3)$-balanced separator, and T_1 is $(3,3)$-linked. On the right, each vertex v_i with $i \in [3]$ constitutes a $(T_2, 1)$-balanced separator.

LEMMA 2.14. *Let D be a digraph. Then* $\mathsf{bn}(D) \leq \mathsf{hn}(D) \leq 2\mathsf{bn}(D)$.

LEMMA 2.15. *Let D be a digraph. Then* $\mathsf{bn}(D) \leq \mathsf{wlink}(D) \leq 4\mathsf{bn}(D)$.

The proof of Proposition 2.13 given in [35] yields an XP algorithm that correctly states that D has a haven of order k or produces an arboreal decomposition of D of width at most $3k-2$. Furthermore, although not explicitly mentioned in the paper, this algorithm actually produces a nice (as in Definition 2.4) arboreal decomposition for D, and can be used as a procedure that, given a digraph D' such that $\mathsf{dtw}(D') \leq k-2$, generates a nice arboreal decomposition for D' of width at most $3k-2$. At each iteration, the algorithm tests whether the strong components intersecting a given set $T \subseteq V(D)$ with $|T| \leq 2k-1$ can be separated into parts containing at most a small portion of T. Namely, the algorithm tests whether there is a set $Z \subseteq V(D)$ with $|Z| \leq k-1$ such that every strong component of $D \setminus Z$ contains at most $k-1$ vertices of $T \setminus Z$. Such a set Z is known as a *balanced separator*. In this paper we consider a generalization of such sets where we can choose how many vertices of T each strong component of $D \setminus Z$ can have.

DEFINITION 2.16 ((T,r)-balanced separators and (k,r)-linked sets). *Let D be a digraph, $T \subseteq V(D)$, and r be a positive integer. A* (T,r)-balanced separator *is a set of vertices $Z \subseteq V(D)$ such that every strong component of $D \setminus Z$ contains at most r vertices of T. If the minimum size of a (T,r)-balanced separator is at least $k+1$, we say that T is* (k,r)-linked.

If $r = \lfloor |T|/2 \rfloor$, (T,r)-balanced separators are exactly T-balanced separators in the classical sense as defined, for instance, in [3, Chapter 9]. If D admits a (T,r)-balanced separator Z, we know that we can split $T \setminus Z$ into small strongly connected parts which are guarded by Z. See Fig. 7 for two examples of (T,r)-balanced separators. A DAG, for instance, admits a $(T,1)$-balanced separator (the empty set) for any $T \subseteq V(D)$ since every strong component of a DAG is formed by a single vertex.

Deciding whether a digraph D admits a $(T, k-1)$-balanced separator is a key ingredient for the algorithm given by Johnson et al. [35]. Moreover, the cost of this procedure has the largest impact on the running time of their algorithm: it is the only step which is (originally) done in XP time, while the remaining parts of the algorithm can be done in polynomial time. In Section 3.2, we use of a variation of the MULTICUT problem introduced in [26] to show how to find (T,r)-balanced separators in FPT time with parameter $|T|$, if any exists with size bounded from above by an integer s with

$s \leq |T| - 1$. In our first main contribution, we use this result to improve on the algorithm for arboreal decompositions given in [35]. Namely, we prove the following.

THEOREM 2.17. *Let D be a digraph and k be a non-negative integer. There is an algorithm running in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$ that either produces a nice arboreal decomposition of D of width at most $3k-2$ or outputs a $(k-1, k-1)$-linked set T with $T = 2k-1$.*

It is also not hard to see how to use (k, r)-linked sets to construct havens. The following lemma is a generalization of a result shown as part of the proof of [35, 3.3].

LEMMA 2.18. *Let D be a graph, $T \subseteq V(D)$ with $|T| = s$, and $r \geq \lfloor s/2 \rfloor$. If T is (k, r)-linked then D admits a haven of order $k+1$.*

Proof. By hypothesis, it holds that, for every set $Z \subseteq V(D)$ with $|Z| \leq k$, there is a strong component C of $D \setminus Z$ such that $|V(C) \cap T| \geq r+1$. Let $\beta(Z) = V(C)$. We claim that β is a haven of order $k+1$ in D. It suffices to show that if $Z' \subseteq Z$, then $\beta(Z) \subseteq \beta(Z')$. Notice that $\beta(Z)$ induces a strongly connected subgraph of D and is disjoint from Z', since it is disjoint from Z, and thus all paths in the graph induced by $\beta(Z)$ are in $D \setminus Z'$. Furthermore, since $|T| = s$ and $r \geq \lfloor s/2 \rfloor$, we have $\beta(Z) \cap \beta(Z') \neq \emptyset$ and the result follows as $\beta(Z')$ is a strong component of $D \setminus Z'$, which is a supergraph of $D \setminus Z$, and thus it must contain completely the strongly connected subgraph induced by $\beta(Z)$. ∎

Applying this lemma on a $(k-1, k-1)$-linked set T with $|T| = 2k-1$ we obtain a haven of order k and therefore we can write Theorem 2.17 with havens instead of (k, r)-linked sets, as done by Johnson et al. [35, 3.3], with the guarantee that the procedure runs in FPT time.

THEOREM 2.19 (First main contribution). *Let D be a digraph and k be a non-negative integer. There is an algorithm running in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$ that correctly states that D admits a haven of order k or produces an arboreal decomposition of D of width at most $3k-2$.*

Next, we discuss some of the steps in the proof of the Directed Grid Theorem.

2.4. Brambles and the Directed Grid Theorem. The Directed Grid Theorem is as stated below.

THEOREM 2.20 (Kawarabayashi and Kreutzer [38]). *There is a function $f : \mathbb{N} \to \mathbb{N}$ such that given any directed graph and any fixed constant k, in polynomial time, we can obtain either*

1. *an arboreal decomposition of D of width at most $f(k)$, or*
2. *a cylindrical grid of order k as a butterfly minor of D.*

The proof of the Directed Grid Theorem [38] starts by asking if a digraph D satisfies $\mathsf{dtw}(D) \leq f(k)$, for some integer k. By Theorem 2.19, an approximate answer to this question can be computed in FPT time with parameter $k \geq 0$. If a haven is obtained, the next step uses it to construct a bramble of large order. In order to justify our following results, we now discuss how to construct brambles from havens.

Finding a hitting set of minimum size of a bramble $\mathcal{B}$ is not an easy task. In general, in order to check whether a given set X is a hitting set of $\mathcal{B}$, the naive approach would be to go through all the elements of $\mathcal{B}$ and verify that X intersects each of them. Since a bramble $\mathcal{B}$ may contain $\Omega(2^n)$ elements, independently of its order, this procedure is not efficient. For instance, consider the digraph D shown in Fig. 8, which has vertex set $\{v_0, v_1, \ldots, v_n\}$ and edge set $\{(v_0, v_i) \cup (v_i, v_0) \mid i \in [n]\}$.

The set $\mathcal{B} = \{D[X] \mid X \subseteq V(D) \text{ and } v_0 \in X\}$ is easily seen to be a bramble in D of order one and size $2^{|V(D)|-1}$ since there is an edge in D from every vertex in $V(D) \setminus \{v_0\}$ to v_0 and vice-versa. However, when $\mathcal{B}$ is the bramble obtained by a

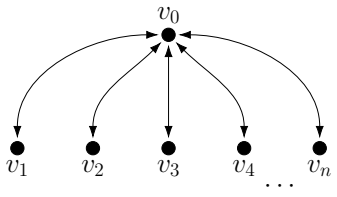

Fig. 8: Example of a digraph D having a bramble of order one and size $2^{|V(D)|-1}$. Here a bidirectional edge is used to represent a pair of edges in both directions.

construction used in a proof of Lemma 2.14, which we present below, then $|\mathcal{B}| = n^{\mathcal{O}(k)}$ and thus in this case we can find hitting sets of $\mathcal{B}$ of size k in XP time, and decide whether a given set $X \subseteq V(D)$ is a hitting set of $\mathcal{B}$ in XP time.

Lemma 2.14 implies that if D is a digraph admitting a haven of order $k+1$, then D contains a bramble of order at least $\lceil (k+1)/2 \rceil = \lfloor k/2 \rfloor + 1$. In fact, given such a haven, it is easy to construct the claimed bramble, as we proceed to explain. Namely, given a haven β of order $k+1$ in D, we define $\mathcal{B} = \{D[\beta(Z)] \mid Z \subseteq V(D) \text{ and } |Z| \leq \lfloor k/2 \rfloor\}$. Note that, since β is a haven, the elements of $\mathcal{B}$ are strongly connected subgraphs of D. We claim that any two elements of $\mathcal{B}$ intersect. Indeed, let $B, B' \in \mathcal{B}$ and let $Z, Z' \subseteq V(D)$ such that $\beta(Z) = V(B)$ and $\beta(Z') = V(B')$. Since $|Z| \leq \lfloor k/2 \rfloor$ and $|Z'| \leq \lfloor k/2 \rfloor$, we have that $|Z \cup Z'| \leq k$, and since β is a haven of order $k+1$, it follows that $\beta(Z \cup Z') \subseteq \beta(Z) \cap \beta(Z') = V(B) \cap V(B')$ and therefore, in particular, $V(B) \cap V(B') \neq \emptyset$. Finally, let us argue about the order of $\mathcal{B}$. Consider an arbitrary vertex set $X \subseteq V(D)$ with $|X| \leq \lfloor k/2 \rfloor$. Since β is a haven or order $k+1 \geq \lfloor k/2 \rfloor$, there is a bramble element $\beta(X) \in \mathcal{B}$ with $V(\beta(X)) \cap X = \emptyset$, and thus $\mathsf{ord}(\mathcal{B}) \geq \lfloor k/2 \rfloor + 1$, as we wanted to prove. Moreover, since there is one element in $\mathcal{B}$ for each $Z \subseteq V(D)$ with $|Z| \leq \lfloor k/2 \rfloor$, we conclude that $|\mathcal{B}| = n^{\mathcal{O}(k)}$.

In [38], the authors show how to obtain, from a bramble $\mathcal{B}$ of order $k(k+2)$, a path P that is a hitting set of $\mathcal{B}$ containing a well-linked set A of size k.

Proposition 2.21 (Kawarabayashi and Kreutzer [38, Lemma 4.3 of the full version]). *Let D be a digraph and $\mathcal{B}$ be a bramble in D. Then there is a path P intersecting every $B \in \mathcal{B}$.*

Proposition 2.22 (Kawarabayashi and Kreutzer [38, Lemma 4.4 of the full version]). *Let D be a digraph, $\mathcal{B}$ be a bramble of order $k(k+2)$ in D, and $P = P(\mathcal{B})$ be a path intersecting every $B \in \mathcal{B}$. Then there is a set $A \subseteq V(P)$ of size k which is well-linked.*

Although the statements of the previous two propositions in [38] are not algorithmic, algorithms for both results can be extracted from their constructive proofs. However, the naive approach to decide if a set $X \subseteq V(D)$ is a hitting set of a bramble $\mathcal{B}$ is to check if $V(B) \cap X \neq \emptyset$ for each $B \in \mathcal{B}$. Thus the running time of the algorithms yielded by the proofs of Propositions 2.21 and 2.22 is influenced by the size of the bramble given as input. Although in general this is not efficient since, as discussed above, a bramble can have size $\Omega(2^n)$ even if it has small order, in the

particular case where $\mathcal{B}$ is the bramble constructed from havens as presented above, those constructions yield XP algorithms with parameter k since $|\mathcal{B}| = n^{\mathcal{O}(k)}$.

In Section 4 we show that, when considering a particular choice of a bramble $\mathcal{B}$ which is constructed from (k, r)-linked sets, for appropriate choices of k and r, we can decide if a given set X is a hitting set of $\mathcal{B}$ in polynomial time and compute hitting sets of $\mathcal{B}$ in FPT time when parameterized by $\mathsf{ord}(\mathcal{B})$. Then, we show how to obtain a path P intersecting all elements of $\mathcal{B}$ in polynomial time, improving Proposition 2.21. We use this latter result to give an FPT algorithm with parameter $\mathsf{ord}(\mathcal{B})$ that produces, from a path P intersecting all elements of a bramble of large order, a well-linked set A of size k which is contained in $V(P)$.

THEOREM 2.23 (Second main contribution). *Let $g(k) = (k+1)(\lfloor k/2 \rfloor + 1) - 1$, D be a digraph and T be a $(g(k)-1, g(k)-1)$-linked set in D with $|T| = 2g(k)-1$. There is an algorithm running in time $2^{\mathcal{O}(k^2 \log k)} \cdot n^{\mathcal{O}(1)}$ that finds in D a bramble $\mathcal{B}$ of order $g(k)$, a path P that is a hitting set of $\mathcal{B}$, and a well-linked set A of order k such that $A \subseteq V(P)$.*

The request that we make on $\mathsf{ord}(\mathcal{B})$ is also an improvement when compared to Proposition 2.22. In the next section we give an overview of how a cylindrical grid is found in [38] from the output of Theorem 2.23. We discuss why the algorithms used in the remaining constructive steps of their proof are naturally FPT to obtain the following corollary, which is an improvement of Theorem 2.20.

COROLLARY 2.24. *Let k be a non-negative integer and D be a digraph. There is a function $f : \mathbb{N} \to \mathbb{N}$ and an FPT algorithm, with parameter k, that either*

1. *produces an arboreal decomposition of D of width at most $f(k)$, or*
2. *finds a cylindrical grid of order k as a butterfly minor of D.*

2.5. Finding a cylindrical grid. On a very high level, the proof of the Directed Grid Theorem [38] can be summarized into the following three steps. Using the terminology adopted in this paper, for a function f as in the statement of Theorem 2.20 and given a digraph D, we

(1) pipeline Theorem 2.17 and Theorem 2.23 to either produce an arboreal decomposition of D of width at most $f(k)$ or construct $\mathcal{B}$, P, and A as in the statement of the latter;
(2) use P and A to construct a well-linked *path system* that is formed by a collection of paths; and
(3) iteratively refine the paths in the path system into new structures until a (butterfly) model of a cylindrical grid is obtained.

See Fig. 9 for an illustration of those steps. As previously mentioned, we only improve on the procedures related to step (1) and, in this section, we justify why this is sufficient to obtain Corollary 2.24. The main observations are that the algorithm runs maintaining and refining a collection of paths, where the size of the collection depends only on k, and that each of those refinements can be realized by iteratively testing how a given path intersects some subset of the collection. The number of tests depends only on k and each test is done in polynomial time. We discuss here how to construct a path system from P and A, as mentioned in step (2) above. For our examples, it is convenient to adopt the following definitions from the full version of [38].

DEFINITION 2.25 (Linkages). *Let D be a digraph and $A, B \subseteq V(D)$ with $A \neq B$. A* linkage from A to B *in D, or an* (A, B)-linkage, *is a set of of pairwise vertex-disjoint*

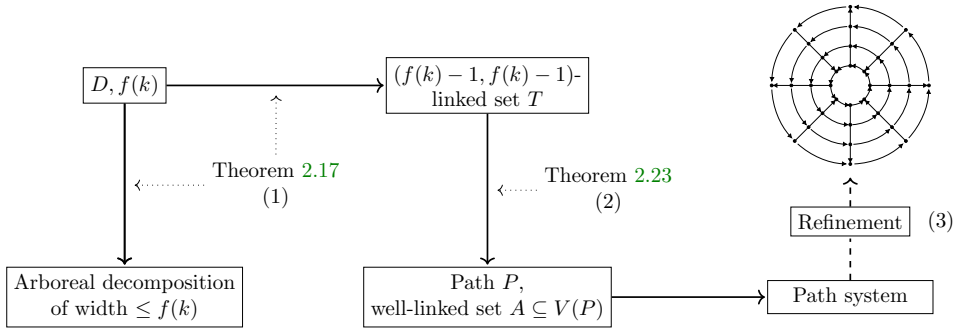

Fig. 9: Illustration of steps (1)-(2)-(3).

paths from A to B.

DEFINITION 2.26 (Path system). *Let D be a digraph and ℓ, p be two positive integers. An* ℓ-linked path system of order p *is a sequence $\mathcal{S}$ with $\mathcal{S} = (\mathcal{P}, \mathcal{L}, \mathcal{A})$ where*

- *$\mathcal{P}$ is a sequence $P_1, \ldots, P_p$ of pairwise vertex-disjoint paths such that, for all $i \in [p]$, $V(P_i) \supseteq A_i^{\mathsf{in}} \cup A_i^{\mathsf{out}}$ and every vertex in A_i^{in} appears in P_i before any vertex of A_i^{out};*
- *$\mathcal{L}$ is a collection $\{L_{i,j} \mid i, j \in [p] \text{ with } i \neq j\}$ of linkages where each $L_{i,j}$ is a linkage of size ℓ from A_i^{out} to A_j^{in}; and*
- *$\mathcal{A} = \{A_i^{\mathsf{in}}, A_i^{\mathsf{out}} \mid i \in [p]\}$ where each A_i^{in} and each A_i^{out} is a well-linked set of order ℓ;*

Although the definition of path systems is quite loaded, it is not hard to visualize; see Fig. 10 for an illustration. Notice that, knowing that the sets $A^i_{\mathsf{in}}, A^i_{\mathsf{out}}$ are well-linked,

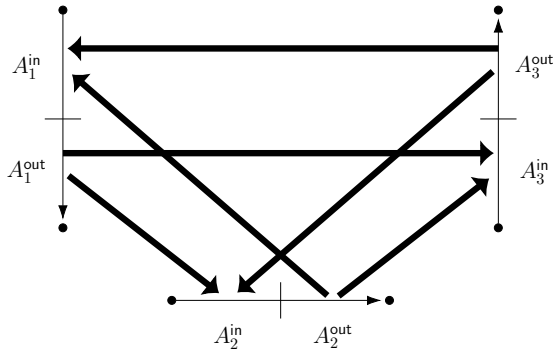

Fig. 10: An ℓ-linked path system of order $p = 3$. A thick edge denotes a linkage of size ℓ from a set A_i^{out} to a set A_j^{in}, with $i \neq j$.

a path system is entirely formed by paths behaving in a particular way: the collection $\mathcal{P}$ of size p, and the collection of paths appearing in the linkages $L_{i,j}$. Since each of those linkages has size ℓ, an ℓ-linked path system of order p is formed by $p + 2\binom{p}{2}\ell$ paths. With this observation, the task of constructing a path system from the output of Theorem 2.23 becomes an easy one, as we proceed to explain.

Assume that we are given a path P and a well-linked set A with $|A| = 2\ell \cdot p$

and $A \subseteq V(P)$. Let $\sigma = a_1, a_2, \ldots, a_{2\ell \cdot p}$ be an ordering of the vertices of A as they appear in P, from the first to the last vertex of the path. To construct an ℓ-linked path system of order p, we follow P in this order and, for $i \in [p]$, we define the path P_i to be the subpath of P from $a_{(i-1)2\ell+1}$ to $a_{i \cdot 2\ell}$. See Fig. 11 for an illustration of this procedure. Since we know that A is well-linked, and clearly every subset of a well-linked set is also well-linked, we define A_i^{in} to be the set containing the first ℓ vertices of $V(P_i) \cap A$ and A_i^{out} to be last ℓ vertices of $V(P_i) \cap A$ with respect to σ.

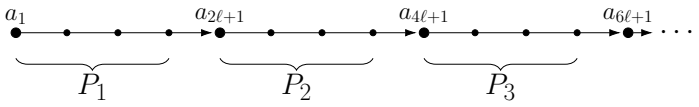

Fig. 11: Finding the paths P_i from P and A, for $i \in [p]$.

Next, for $i, j \in [p]$ with $i \neq j$, we choose $L_{i,j}$ to be a linkage from A_i^{out} to A_j^{in}. At least one choice for $L_{i,j}$ is guaranteed to exist because $A_i^{\mathsf{in}} \cup A_j^{\mathsf{out}} \subseteq A$ and A is well-linked. Moreover, we can find each linkage in polynomial time by applying Menger's Theorem (cf. Theorem 2.1) and solving a flow problem. Hence, given P and A of adequate size, we can find an ℓ-linked path system of order p in polynomial time.

As in Fig. 12, it is easy to find a cylindrical grid in a sufficiently large path system if it is "well-behaved", that is, when the paths in $\mathcal{L}$ are pairwise internally vertex-disjoint. In fact, in such cases every P_i models one vertex of a *biclique* that is a butterfly minor of D. A biclique is a digraph H having a pair of edges in both directions between any two vertices of H. Clearly, a biclique with $2k^2$ vertices contains a cylindrical grid of order k.

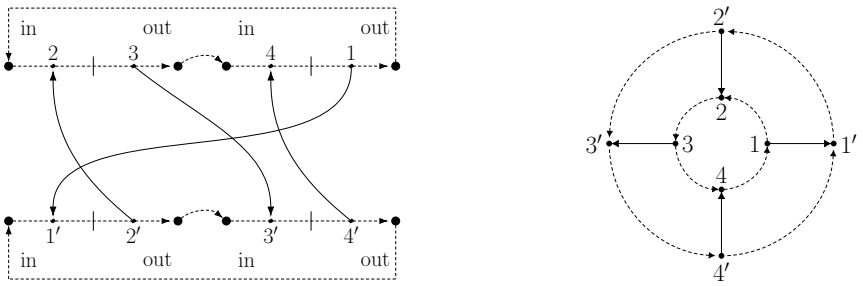

Fig. 12: An example of cylindrical grid of order two in a "well-behaved" path system, where we assume that the paths in $\mathcal{L}$ are pairwise internally vertex-disjoint.

Unfortunately, in general we cannot expect every path system to behave in this way. Hence, the proof of the Directed Grid Theorem by Kawarabayashi and Kreutzer [38] follows a sequence of refinements, as mentioned in item (3) above, each constructing a new structure from the previous one until a cylindrical grid is obtained. This part is represented by the dashed edges in Fig. 2 and Fig. 9 and, although it is not hard to see that the algorithms realizing those constructions are naturally FPT, the constructive proofs are in fact the largest and most involved part of their paper. Namely, they show how to find a *web*[4] or a cylindrical grid from a path system that

[4]The definitions of *webs* and *fences* can be found in the full version of [38].

is sufficiently large. If a web is obtained, then the next step is to find a *fence*[4] in it. Lastly, they prove that we are guaranteed to find a cylindrical grid of order k in any sufficiently large fence.

Fortunately, and as it is the case with path systems, webs and fences are defined around collections of paths satisfying some properties that can be easily verified in polynomial time. Since the number of paths in a ℓ-linked path system of order p depends only on ℓ and p, we can search for a web in a path system by testing the defining properties of webs for every subset of the set of paths in the path system. Thus, in FPT time with parameters ℓ and p we can find a web in a path system. A similar approach is viable to find fences in webs and cylindrical grids in fences and thus Corollary 2.24 follows from Theorem 2.23.

3. Balanced separators and arboreal decompositions. The algorithm for arboreal decompositions given in [35] starts with a trivial decomposition $(\{r\}, \emptyset, \{W_r\})$ whose underlying arborescence contains only one vertex r. Thus, $W_r = V(G)$. Each iteration splits the vertices contained in an excessively large leaf of the current decomposition, if one exists, into a set of new leaves, while guaranteeing that the width of the non-leaf vertices remains bounded from above by a function of k. Although this problem is not explicitly named by the authors, on each of those split operations the algorithm has to decide whether the input digraph admits a (T, r)-balanced separator for a given set T. Formally, on each iteration the need to solve a particular case of the following problem.

Balanced Separator	
Input:	A digraph D, a set $T \subseteq V(D)$ of size k, and two non-negative integers r and s.
Output:	A (T, r)-balanced separator Z with $\lvert Z\rvert \leq s$, if it exists.

The Balanced Separator problem can be naively solved by checking all $\binom{n}{s}$ sets Z of size s in $V(D)$ and enumerating the strong components of $D \setminus Z$. Therefore it is in XP with parameter s. Furthermore, the process of finding balanced separators is the only step of the algorithm given in [35] that is done in XP time. In the next section, we show how to compute (T, r)-balanced separators in FPT time with parameter k. In particular, we show that a set Z is a (T, r)-balanced separator if and only if Z is a solution to a separation problem introduced in [26] that is a particular case of the Multicut problem in digraphs. Then, we use this result to improve the algorithm by Johnson et al. [35] for approximate arboreal decompositions (cf. Proposition 2.13), showing that it can be done in FPT time. Notice that we can assume that $r \leq k-1$ and $s \leq k-r-1$: if $r \geq k$, the empty set is a (T, r)-balanced separator and, if $s \geq k-r$, any choice of s vertices from T form a (T, r)-balanced separator. To avoid repetition, we make these considerations here and refrain from repeating them in the remainder of this article. We refer to instances of Balanced Separator as (D, T, k, r, s).

3.1. Computing (T, r)-balanced separators in FPT time. Given a graph or digraph D and a set of pairs of terminal vertices $\{(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)\}$, the Multicut problem asks to minimize the size of a set $Z \subseteq V(D)$ such that there is no path from s_i to t_i in $D \setminus Z$, for $i \in [k]$. When parameterized by the size of the solution, the problem is FPT in undirected graphs [11, 44]. On the directed case, this problem is FPT in DAGs when parameterized by the size of the solution and the number of pairs of terminals [41], but W[1]-hard in the general case even for fixed $k = 4$ [47].

A variation of MULTICUT is considered in [26]. Namely, in the LINEAR EDGE CUT problem, we are given a digraph D and a collection of sets of vertices $\{S_1, \ldots, S_k\}$, and we want to find a minimum set of edges Z such that there is no path from S_i to S_j in $D \setminus Z$ whenever $j > i$. We remark that the authors in [26] refer to this problem as LINEAR CUT only. This problem is FPT when parameterized by the size of the solution:

PROPOSITION 3.1 (Erbacher et al. [26]). *The* LINEAR EDGE CUT *problem can be solved in time* $\mathcal{O}(4^s \cdot s \cdot n^4)$*, where* s *is the size of the solution.*

We remark that the authors of [26] mention that this result can also be achieved by using a reduction to the SKEW SEPARATOR algorithm given in [15].

In this section, we show how to use the algorithm for the LINEAR EDGE CUT problem to solve the vertex version, and then show how this version can be used to compute (T, r)-balanced separators in FPT time. We formally define the vertex version below.

LINEAR VERTEX CUT

Input: A digraph D, a collection of terminal sets $\mathcal{T}$, with $\mathcal{T} = \{T_1, T_2, \ldots, T_k\}$, where $T_i \subseteq V(D)$ for $i \in [k]$, and an integer $s \geq 0$.

Question: Is there a set of vertices $Z \subseteq V(D)$ with $|Z| \leq s$ such that there are no paths in $D \setminus Z$ from T_i to T_j, for $1 \leq i < j \leq k$?

From an instance $(D, \mathcal{T}, s)$ of LINEAR VERTEX CUT, we construct an equivalent instance of $(D', \mathcal{T}', s)$ of LINEAR EDGE CUT as follows. First, notice that any vertex v occurring in the intersection of two distinct sets in $\mathcal{T}$ must be part of any solution for the instance. Thus we can assume that every vertex of D occurs in at most one set in $\mathcal{T}$. Now, for each vertex $v \in V(D)$, add to D' two vertices v_{in} and v_{out} and an edge e_v from v_{in} to v_{out}. For each edge $e \in E(D)$ with tail u and head v, add to D' a set of $s+1$ parallel edges from u_{out} to v_{in}. Finally, for each $v \in T_i$, for $i \in [k]$, add a new vertex v' to D' together with $s+1$ edges from v' to v_{in} and $s+1$ edges from v_{out} to v'. Let $T'_i = \{v' \mid v \in T_i\}$ and $\mathcal{T}' = \{T'_1, \ldots, T'_k\}$. We have the following easy lemma.

LEMMA 3.2. *An instance* $(D, \mathcal{T}, s)$ *of* LINEAR VERTEX CUT *is positive if and only if the associated instance* $(D', \mathcal{T}', s)$ *of* LINEAR EDGE CUT *is positive.*

Proof. Let $Z \subseteq V(D)$ be a solution for $(D, \mathcal{T}, s)$ and $Z' = \{e_v \mid v \in Z\} \subseteq E(D')$. By contradiction, assume that there is a path P' in $D' \setminus Z'$ from a vertex u' to a vertex v', for $u' \in T'_i$, $v' \in T'_j$, and $j > i$. Then there is a path P from u to v in $D \setminus Z$ with vertex set $\{v \mid e_v \in E(P')\}$. This contradicts our choice of Z and thus the necessity holds.

For the sufficiency, let Z' be a minimal solution for $(D', \mathcal{T}', s)$. Notice that all edges in Z' are from a vertex v_{in} to its respective v_{out}, as the budget s for the size of Z' does not allow any other choice. Let $Z = \{v \mid e_v \in Z'\}$ and, by contradiction, let P be a path in $D \setminus Z$ from a vertex u to a vertex v, with $u \in T_i$, $v \in T_j$, and $j > i$. For each edge $e \in E(P)$ with $e = (x, y)$ there is an edge e' with $e' = (x_{\mathsf{out}}, y_{\mathsf{in}})$ in $D' \setminus Z'$. Let F' be the set of such edges of D'. Now, there is a path P' from u_{in} to v_{out} in D' with edge set $\{e_v \mid v \in V(P)\} \cup F'$. Appending to P' the edges from u' to u_{in} and from v_{out} to v' we construct a path from u' to v' in $D \setminus Z'$, contradicting our choice of Z'. Therefore, the sufficiency also holds and the lemma follows. ∎

Combining Proposition 3.1 and Lemma 3.2 we get the following.

COROLLARY 3.3. *There is an* FPT *algorithm for the* LINEAR VERTEX CUT *problem parameterized by the size s of the solution and running in time $\mathcal{O}(4^s \cdot s \cdot n^4)$.*

We now show how to solve BALANCED SEPARATOR using LINEAR VERTEX CUT. Namely, we show that a digraph D admits a (T, r)-balanced separator Z if and only if Z is a solution to some instance $(D, \mathcal{T}, s)$ of LINEAR VERTEX CUT where $\mathcal{T}$ depends of T.

LEMMA 3.4. *Let (D, T, k, r, s) be an instance of* BALANCED SEPARATOR. *A set $Z \subseteq V(D)$ with $|Z| \leq s$ is a (T, r)-balanced separator if and only if there is a partition $\mathcal{T}$ of T into sets $T_1, T_2, \ldots, T_\ell$ such that*

1. *$|T_i| \leq r$, for $i \in [\ell]$, and*
2. *Z is a solution for the instance $(D, \mathcal{T}, s)$ of* LINEAR VERTEX CUT.

Proof. For the necessity, let Z be a (T, r)-balanced separator with $|Z| \leq s$. Let $\mathcal{C}$ be the set of strong components of $D \setminus Z$ and consider an ordering $C_1, \ldots, C_\ell$ of its elements such that there is no path from C_i to C_j in $D \setminus Z$ whenever $j > i$. Notice that this is the reverse of a topological ordering for the elements of $\mathcal{C}$. Let $v_1, \ldots, v_q$ be the vertices in $T \cap Z$, if any exist. For $i \in [\ell]$, choose $T_i = V(C_i) \cap T$ and define $\mathcal{T} = \{T_1, T_2, \ldots, T_\ell\}$ if $T \cap Z \neq \emptyset$ or $\mathcal{T} = \{T_1, T_2, \ldots, T_\ell, \{v_1\}, \ldots, \{v_q\}\}$ otherwise. Notice that it is possible for a set T_i to be empty.

Since Z is a (T, r)-balanced separator, we know that $|T_i| \leq r$ holds for all $i \in [\ell]$. Since the vertices in a non-empty set T_i are contained in exactly one strong component of $D \setminus Z$, any path between different sets in $\mathcal{T}$ must contain a path between distinct strong components of $D \setminus Z$. Thus we conclude that there are no paths from a set T_i to another set T_j with $j > i$, since otherwise we would have a contradiction to our choice for the order of the elements of $\mathcal{C}$, and therefore Z is a solution for the instance $(D, \mathcal{T}, s)$ of LINEAR VERTEX CUT.

For the sufficiency, let $\mathcal{T}$ be as in the statement of the lemma and Z be a solution for the instance $(D, \mathcal{T}, s)$ of LINEAR VERTEX CUT. First, notice that no strong component of $D \setminus Z$ can intersect two distinct sets $T, T' \in \mathcal{T}$. Indeed, if this were the case, then there would be a path in $D \setminus Z$ from a vertex in T to a vertex in T' and vice-versa, contradicting the fact that Z is a solution for $(D, \mathcal{T}, s)$. Thus, if $|V(C) \cap T| \geq r+1$ for some strong component C of $D \setminus Z$, we have a contradiction as C would intersect at least two distinct sets in $\mathcal{T}$. We conclude that Z is a (T, r)-balanced separator and the lemma follows. ∎

The FPT algorithm for BALANCED SEPARATOR follows from Lemma 3.4 and Corollary 3.3. The running time is heavily tied to the number of partitions $\mathcal{T}$ that can be generated from a given set T of an instance (D, T, k, r, s) of BALANCED SEPARATOR. This value is bounded by the k-th *ordered Bell number* [12]. The *Bell number* [4] counts the number of partitions of a set, and its ordered variant also considers the number of possible orderings for each partition. The k-th ordered Bell number is of the form $2^{\mathcal{O}(k \log k)}$. From the previous discussion we get the following theorem.

THEOREM 3.5. *There is an algorithm running in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$ for the* BALANCED SEPARATOR *problem.*

Proof. Let (D, T, k, r, s) be an instance of BALANCED SEPARATOR and $\mathcal{T}^*$ be the set of all ordered partitions $\{T_1, \ldots, T_\ell\}$ of T with $|T_i| \leq r$, for $i \in [\ell]$.

By Corollary 3.3, we can solve instances of LINEAR VERTEX CUT problem in time $\mathcal{O}(4^s \cdot s \cdot n^4)$ for s being the size of the solution. By Lemma 3.4, Z is a (T, r)-balanced separator if and only if there is a $\mathcal{T} \in \mathcal{T}^*$ such that the instance $(D, \mathcal{T}, s)$

of Linear Vertex Cut is positive. Finally, since $|\mathcal{T}^*|$ is at most the k-th ordered Bell number, we can solve Balanced Separator by testing $2^{\mathcal{O}(k \log k)}$ instances of Linear Vertex Cut. As $s \leq k - r$ (since otherwise the instance of Balanced Separator is trivially positive), the bound on the running time follows. ∎

3.2. An FPT algorithm for approximate arboreal decompositions. We are now ready to prove Theorem 2.19. We remark that the proof below follows [35, 3.3] except that we replace the XP procedure of the proof by our FPT algorithm for Balanced Separator. In the following proof, we need to test whether a given set $T \subseteq V(D)$ admits a $(T, k-1)$-balanced separator of size at most $k-1$. Thus we remind the reader of the discussion made in the beginning of Section 3: if $|T| \leq 2k-2$, then the answer is positive since we can pick any $k-1$ vertices of T to form a solution.

Theorem 2.17. *Let D be a digraph and k be a non-negative integer. There is an algorithm running in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$ that either produces a nice arboreal decomposition of D of width at most $3k-2$ or outputs a $(k-1, k-1)$-linked set T with $T = 2k-1$.*

Proof. We begin with a nice arboreal decomposition $(R_0, \mathcal{X}_0, \mathcal{W}_0)$ of D where $\mathcal{X}_0 = \emptyset$, $V(R_0) = \{r\}$, and $\mathcal{W}_0 = \{V(D)\}$. We maintain an arboreal decomposition $(R, \mathcal{X}, \mathcal{W})$ of D for which the following two properties hold:

(P1) $|W_r \cup (\bigcup_{e \sim r} X_e)| \leq 3k-1$ for every $r \in V(R)$ of out-degree at least one, and
(P2) $|X_e| \leq 2k-1$ for every $e \in E(R)$.

Notice that both (P1) and (P2) hold for $(R_0, \mathcal{X}_0, \mathcal{W}_0)$.

If (P1) holds for all $r \in V(R)$, then we have constructed an arboreal decomposition with the desired width. Otherwise, we can assume that $(R, \mathcal{X}, \mathcal{W})$ contains at least one leaf that is *too large*. That is, the width of a vertex r_0 of out-degree zero of R is at least $3k$. If there is an edge $e_0 \in E(R)$ with head r_0, let $T = X_{e_0}$. Otherwise, let $T = \emptyset$. Either way, $|T| \leq 2k-1$ and $|W_{r_0}| \geq 3k - |T| \geq k+1$.

Now, we test whether D contains a $(T, k-1)$-balanced separator of size at most $k-1$ and, by Theorem 3.5, this test can be done in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$. If $|T| \leq 2k-2$ then the answer is positive since we can pick any set of $k-1$ vertices of T to form a solution. Thus, if the answer is negative, we have $|T| = 2k-1$ and we terminate the algorithm outputting T. We may now assume that D contains a $(T, k-1)$-balanced separator Z' with $|Z'| \leq k-1$.

From the bound on the sizes of the sets, there are at least two vertices in $W_{r_0} \setminus Z'$. Choose v to be any of those two vertices, and let $Z = Z' \cup \{v\}$. Now $|Z| \leq k$, $Z \cap W_{r_0} \neq \emptyset$, and $|V(C) \cap T| \leq k-1$ holds for every strong component C of $D \setminus Z$.

Let $C_1, \ldots, C_\ell$ be the strong components of $D \setminus Z$. If B is a strong component of $C_i \setminus T$, for $i \in [\ell]$, then either $V(B) \subseteq W_{r_0}$ or $V(B) \cap W_{r_0} = \emptyset$, for W_{r_0} is T-guarded. Let $B_1, \ldots, B_d$ be all such strong components for which $V(B_j) \subseteq W_{r_0}$ for all $j \in [d]$. Furthermore, let $f : \mathbb{N} \to \mathbb{N}$ be a function assigning an index j to an index i if and only if $B_i \subseteq C_j \setminus T$. Thus, f can be used to tell which set C_j contains a given B_i. Now, $Z \cap W_{r_0}, V(B_1), \ldots, V(B_d)$ is a partition of W_{r_0} into non-empty sets. We show that this partition yields another arboreal decomposition of D.

Let R' be the arborescence obtained from R by adding a vertex r_i and an edge e_i from r_0 to r_i, for $i \in [d]$. Furthermore, let $X'_e = X_e$ for all $e \in E(R)$ and $W'_r = W_r$ for all $r \in V(R) \setminus \{r_0\}$. Also, let $W'_{r_0} = W_{r_0} \cap Z$ and, for $i \in [d]$, let $X'_{e_i} = Z \cup (V(C_{f(i)}) \cap T)$ and $W'_{r_i} = V(B_i)$. Finally, define $\mathcal{X}' = \{X'_e \mid e \in E(R')\}$ and $\mathcal{W}' = \{W'_r \mid r \in V(R')\}$. As the vertices of W_{r_0} have been spread into non-empty sets, we only need to verify that $(R', \mathcal{X}', \mathcal{W}')$ is an arboreal decomposition of D for

which (P1) and (P2) hold; see Fig. 13 for an illustration.

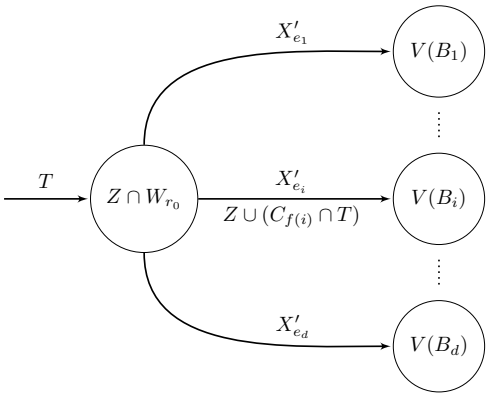

Fig. 13: Spreading the vertices in W_{r_0}.

$\mathcal{W}'$ is indeed a partition of $V(D)$ into non-empty sets, as W_{r_0} is partitioned into non-empty sets. For $i \in [d]$, $W'_{r_i} = V(B_i)$ and B_i is a strong component of $C_{f(i)} \setminus T$. Thus, each new leaf r_i added to R is such that W'_{r_i} is X'_{e_i}-guarded and, for all $e \in E(R')$, $\bigcup\{W'_r : r \in V(R'), r > e\}$ is X'_e-guarded as the property remains unchanged for all $e \in E(R)$.

For $r \in V(R)$, the validity of (P1) remains unchanged. The width of r_0 is bounded from above by $|T| + |Z| \leq 2k - 1 + k = 3k - 1$, as desired, for $W'_{r_0} \subseteq Z$ and $\bigcup_{e \sim r_0} X'_e \subseteq T \cup Z$. (P2) remains true in $(R', \mathcal{X}', \mathcal{W}')$ for all $e \in E(R)$. For e_i, $i \in [d]$, $|X_{e_i}| \leq |Z| + |V(C_{f(i)}) \cap T|$. By the assumption that $(D, T, 2k-1, k-1, k-1)$ is a positive instance of Balanced Separator, $|Z| + |V(C_{f(i)} \cap T| \leq k + k - 1 = 2k - 1$.

Observe that, since each B_i is disjoint from $T \cup Z$, $(R', \mathcal{X}', \mathcal{W}')$ is actually a *nice* arboreal decomposition.

Now, if no leaf of $(R', \mathcal{X}', \mathcal{W}')$ is too large, we end the algorithm returning this arboreal decomposition of D. Otherwise, we repeat the aforementioned procedure with new choices for T and W_{r_0}.

Finally, the running time holds by Theorem 3.5, since $\mathcal{W}$ partitions $V(D)$ into non-empty sets and each iteration decreases the number of vertices in leaves that have width at least $3k$. ∎

The proof of Theorem 2.19 easily follows from Lemma 2.18 and Theorem 2.17.

Theorem 2.19 (First main contribution). *Let D be a digraph and k be a non-negative integer. There is an algorithm running in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$ that correctly states that D admits a haven of order k or produces an arboreal decomposition of D of width at most $3k - 2$.*

Proof. Applying Theorem 2.17 with input D, we either produce an arboreal decomposition of D of width at most $3k - 2$ or find a set $T \subseteq V(D)$ with $|T| = 2k - 1$ such that there is no $(T, k-1)$-balanced separator in D. Now, by Lemma 2.18 applied with inputs D, T, $r = k - 1$, and $s = k - 1$, we conclude that D admits a haven of order k and the result follows. ∎

Next, we show to use Theorem 2.17 to construct a bramble in digraphs of large directed tree-width that is easier to work with than the usual construction that depends on havens (see, for instance, [45, Chapter 6]).

4. Brambles and well-linked systems of paths. Let T be the set constructed by Theorem 2.17 applied to a digraph D with n vertices and $\mathsf{dtw}(D) \geq 3k-1$, and let $\mathcal{H}$ be the haven obtained by applying Lemma 2.18 with input D and T. We remark that from $\mathcal{H}$ it is possible to construct a bramble $\mathcal{B}$ of order $\lfloor k/2 \rfloor$ and size $|V(D)|^{\mathcal{O}(k)}$ (see the discussion in Section 2.4). In this particular case the naive approach yields an XP algorithm to find a hitting set of $\mathcal{B}$ of size k in XP time with parameter k, by checking all $\binom{n}{k}$ subsets X of $V(D)$ with size k and testing whether $X \cap V(B) \neq \emptyset$ for each $B \in \mathcal{B}$, and thus XP algorithms can be extracted from the constructive proofs of Propositions 2.21 and 2.22 assuming that these properties hold for the input brambles. In Section 4.1, we show how to construct from T a bramble $\mathcal{B}_T$ of order k in digraphs with directed tree-width at least $3k-1$ that skips havens and is more efficient in the following two ways.

First, this construction allows us to verify whether an induced subgraph D' of D contains an element of $\mathcal{B}_T$ by looking only at the strong components of D'. This allows us to test if a given set $X \subseteq V(D)$ is a hitting set of $\mathcal{B}_T$ in polynomial time. Second, we show that a set $Y \subseteq V(D)$ is a minimum hitting set of $\mathcal{B}_T$ if and only if Y is a solution for an appropriately defined instance of Balanced Separator. Since we showed that this problem is FPT with parameter $|T|$ (Theorem 3.5), we can compute hitting sets of $\mathcal{B}_T$ in FPT time with parameter $\mathsf{ord}(\mathcal{B}_T)$. Then, in Section 4.2 we use those results to prove stronger versions of Propositions 2.21 and 2.22.

4.1. Brambles in digraphs of large directed tree-width. We now define *T-brambles* and some of its properties when T is the set obtained by applying Theorem 2.17 to a digraph D with $\mathsf{dtw}(D) \geq 3k-1$.

Definition 4.1. *Let D be a digraph and $T \subseteq V(D)$ with $|T| = 2k-1$. The* T-bramble $\mathcal{B}_T$ *of D is defined as*

$$\mathcal{B}_T = \{B \subseteq D \mid B \text{ is induced, strongly connected, and } |V(B) \cap T| \geq k\}.$$

Notice that $\mathcal{B}_T$ is a bramble since, as $|T| = 2k-1$, any two of its element intersect. We remark that, in general, it is possible that $\mathsf{ord}(\mathcal{B}_T)$ is very small: it is in fact zero if, for example, no two vertices of T lay in the same strong component of D. Note also that $\mathcal{B}_T$ may be empty if, for instance, any strong component of D has size strictly smaller than k.

Lemma 4.2. *Let D be a digraph and T be a $(k-1,k-1)$-linked set of size $2k-1$ in D. Then the T-bramble $\mathcal{B}_T$ is a bramble of order k and a set $X \subseteq V(D)$ is a hitting set of $\mathcal{B}_T$ if and only if X is a $(T,k-1)$-balanced separator.*

Proof. Let D, T and $\mathcal{B}_T$ be as in the statement of the lemma. Since $|T| = 2k-1$, any set containing k vertices of T is a hitting set of $\mathcal{B}$. Thus $\mathsf{ord}(\mathcal{B}_T) \leq k$. Let $Z \subseteq V(D)$ with $|Z| \leq k-1$. By definition of $(k-1,k-1)$-linked sets, D does not contain any $(T,k-1)$-balanced separator of size $k-1$, and hence there is a strong component B of $D \setminus Z$ such that $|V(B) \cap T| \geq k$. Since $V(B) \cap Z = \emptyset$ and $B \in \mathcal{B}_T$, we conclude that Z is not a hitting set of $\mathcal{B}_T$ and therefore $\mathsf{ord}(\mathcal{B}_T) = k$.

For the second part of the lemma, let X be a hitting set of $\mathcal{B}_T$. Then $|V(C) \cap T| \leq k-1$ holds for every strong component C of $D \setminus X$ and, by definition, X is a $(T,k-1)$-balanced separator. Similarly, if X is a $(T,k-1)$-balanced separator then, by definition of $\mathcal{B}_T$, X is a hitting set of $\mathcal{B}_T$ and the result follows. ∎

Note that we can check whether a given set $X \subseteq V(D)$ is a hitting set of $\mathcal{B}_T$ by enumerating the strong components of $D \setminus X$ and, for each such a component C,

checking whether $|V(C) \cap T| \geq k$. This can be done in time $\mathcal{O}(n+m)$. For the remainder of this section, and unless stated otherwise, let T be a $(k-1,k-1)$-linked set with $|T| = 2k-1$. In what follows, we use T-brambles to adapt Proposition 2.22 into an FPT algorithm.

To prove our version of Proposition 2.22, we start with a T-bramble $\mathcal{B}_T$ of order $g(k)$ (the value of $g(k)$ is specified later) in a digraph D with $\mathsf{dtw}(D) \geq 3g(k)-1$, and then we show how to find in polynomial time a path $P(\mathcal{B}_T)$ that is a hitting set of $\mathcal{B}_T$, adapting the proof of Proposition 2.21 shown in [38, Lemma 4.3 of the full version]. Next, we need to show how to split $\mathcal{B}_T$ into brambles of order at least $\lceil k/2 \rceil$ whose elements are intersected by subpaths of $P(\mathcal{B}_T)$. We do this by growing a subpath of P' of $P(\mathcal{B}_T)$ iteratively while checking, on each iteration, whether the set $\mathcal{B}'_T$ of elements of $\mathcal{B}_T$ intersecting $V(P')$ is a bramble of adequate order.

We now show how our choice of $\mathcal{B}_T$ allows us to estimate the order of $\mathcal{B}'_T$ by computing the order of its "complement bramble" $\mathcal{B}_T \setminus \mathcal{B}'_T$, and we show how to do this procedure in FPT time with parameter $\mathsf{ord}(\mathcal{B}_T)$. These ideas are formalized by the following definitions and results.

Definition 4.3. *Let $X \subseteq V(D)$ and $\mathcal{B}$ be a bramble in D. The* restricted bramble $\mathcal{B}(X)$ *contains the elements of $\mathcal{B}$ intersecting X and its* complement bramble $\overline{\mathcal{B}}(X)$ *contains the elements of $\mathcal{B}$ disjoint from X. Formally,*

$$\mathcal{B}(X) = \{B \in \mathcal{B} \mid V(B) \cap X \neq \emptyset\},$$

$$\overline{\mathcal{B}}(X) = \{B \in \mathcal{B} \mid V(B) \cap X = \emptyset\}.$$

Notice that both $\mathcal{B}(X)$ and $\overline{\mathcal{B}}(X)$ are brambles, as both are subsets of a bramble $\mathcal{B}$. Additionally, $\mathcal{B}(X)$ is disjoint from $\overline{\mathcal{B}}(X)$ and the union of a hitting set of the former with a hitting set of the latter is a hitting set of $\mathcal{B}$. From this remark, we have that

$$\mathsf{ord}(\mathcal{B}(X)) + \mathsf{ord}(\overline{\mathcal{B}}(X)) \geq \mathsf{ord}(\mathcal{B}), \tag{4.1}$$

and although in general the order of $\mathcal{B}(X)$ is hard to compute, we can estimate it by knowing the order of its complement bramble $\overline{\mathcal{B}}(X)$ and $\mathsf{ord}(\mathcal{B})$.

Consider now the brambles $\mathcal{B}_T$, $\mathcal{B}_T(X)$, and $\overline{\mathcal{B}_T}(X)$ for some $X \subseteq V(D)$. The following results show that hitting sets of $\overline{\mathcal{B}_T}(X)$ are exactly $(T \setminus X, k-1)$-balanced separators in $D \setminus X$.

Lemma 4.4. *Let $X, Z \subseteq V(D)$ and B be a strongly connected subgraph of D. Then $B \in \overline{\mathcal{B}_T}(X)$ and $V(B) \cap Z = \emptyset$ if and only if B is a strongly connected subgraph of $D \setminus (Z \cup X)$ with $|V(B) \cap T| \geq k$.*

Proof. For the necessity, assume that $B \in \overline{\mathcal{B}_T}$ and $V(B) \cap Z = \emptyset$. Then by the definition of $\overline{\mathcal{B}_T}(X)$, B is a strongly connected subgraph of $D \setminus (Z \cup X)$ intersecting T in at least k vertices.

For the sufficiency, assume that B is a strongly connected subgraph of $D \setminus (Z \cup X)$ containing at least k vertices of T. Then $B \in \overline{\mathcal{B}_T}(X)$ by the definition of $\overline{\mathcal{B}_T}(X)$ and the lemma follows since it is disjoint from $Z \cup X$. ∎

The contrapositive of Lemma 4.4 characterizes hitting sets of $\overline{\mathcal{B}_T}(X)$.

Corollary 4.5. *Let $X, Z \subseteq V(D)$. Z is a hitting set of $\overline{\mathcal{B}_T}(X)$ if and only if Z is a $(T \setminus X, k-1)$-balanced separator in $D \setminus X$.*

Therefore, we can decide whether $\mathsf{ord}(\overline{\mathcal{B}_T}(X)) \leq s$ by testing whether D admits a $(T \setminus X, k-1)$-balanced separator of size s. The following result is a direct consequence of Theorem 3.5 and Corollary 4.5.

COROLLARY 4.6. *For any $X \subseteq V(D)$, there is an algorithm running in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$ that decides whether $\mathsf{ord}(\mathcal{B}_T(X)) \leq s$.*

Next, we show how to find such a path $P(\mathcal{B}_T)$ as described above and a well-linked set A of size roughly $\sqrt{2k}$ that is contained in $V(P(\mathcal{B}_T))$.

4.2. Finding $P(\mathcal{B}_T)$ and A. The proof of the next lemma is an adaptation of the proof of [38, Lemma 4.3 of the full version] to our scenario. We exploit the fact that we can check whether a given set of vertices is a hitting set of $\mathcal{B}_T$ in polynomial time: by Lemma 4.2, a set $X \subseteq V(D)$ is a hitting set of $\mathcal{B}_T$ if and only if X is a $(T, k-1)$-balanced separator, and we can check if a given set X is a $(T, k-1)$-balanced separator by enumerating the strong components of the input digraph.

LEMMA 4.7. *Let D be a digraph, let T be a $(k-1, k-1)$-linked set of size $2k-1$, and consider the T-bramble $\mathcal{B}_T$. There is an algorithm running in time $\mathcal{O}(n(n+m))$ that produces a path P that is a hitting set of $\mathcal{B}_T$.*

Proof. If $\mathsf{ord}(\mathcal{B}_T) \geq 1$, then there is an element $B \in \mathcal{B}_T$ and a strong component C of D such that $V(B) \subseteq V(C)$ and, by the definition of $\mathcal{B}_T$, we know that $D[V(C)] \in \mathcal{B}_T$. Define $B_1 = D[V(C)]$, let v_1 be any vertex of B_1, and define P_1 as the path containing only the vertex v_1 and $V(P_0) = \emptyset$. We proceed to grow a path by iterating from P_1 to $P_{k'}$ where they all start from v_1, each P_i with $i \geq 2$ contains P_{i-1}, and $P_{k'}$ is a hitting set of $\mathcal{B}_T$. Throughout our process, we maintain a collection of elements $B_i \in \mathcal{B}_T$ such that $V(P_i)$ intersects $V(B_i)$ only in the last vertex v_i of P_i. Since $|V(P_1)| = 1$ and $v_1 \in T \subseteq V(B_1)$, this condition trivially holds for P_1. Assume now that i paths have been chosen this way, with $i \geq 1$.

Consider the last vertex v_i of the path P_i and the element B_i of $\mathcal{B}_T$ with $V(P_i) \cap V(B_i) = \{v_i\}$. By Lemma 4.2, $V(P_i)$ is a hitting set of $\mathcal{B}_T$ if and only if $V(P_i)$ is a $(T, k-1)$-balanced separator, and this can be tested in time $\mathcal{O}(n+m)$ by enumerating all strong components of $D \setminus V(P_i)$. If $V(P_i)$ is a hitting set of $\mathcal{B}_T$, then we terminate the algorithm returning P_i. Otherwise, $V(P_i)$ is not a $(T, k-1)$-balanced separator and thus there is a strong component F of $D \setminus V(P_i)$ with $|V(F) \cap T| \geq k$. Therefore, $D[V(F)]$ is an element of $\mathcal{B}_T$ whose vertices are disjoint from $V(P_i)$ and we choose $B_{i+1} = D[V(F)]$.

Since $\mathcal{B}_T$ is a bramble, we can find a path P' from $v_i \in V(P_i) \cap V(B_i)$ to a vertex $v_{i+1} \in B_{i+1}$ in $D[V(B_i) \cup V(B_{i+1})]$ such that $V(P') \cap V(B_{i+1}) = \{v_{i+1}\}$. Moreover, v_i is the only vertex of P_i in B_i and thus the path P' does not contain any vertex in $V(P_i) \setminus \{v_i\}$. Now, let P_{i+1} be the path obtained from P_i by appending P'. By our choice of P', we know that only the last vertex v_{i+1} of P_{i+1} is in $V(B_{i+1})$, as desired, and $V(P_{i+1})$ hits strictly more elements of $\mathcal{B}_T$ than $V(P_i)$. We repeat the aforementioned procedure now considering the vertex v_{i+1}, the path P_{i+1}, and the element B_{i+1} of $\mathcal{B}_T$.

Since we can enumerate the strong components of a subgraph of D in time $\mathcal{O}(n+m)$ (see, for instance, [10, Chapter 6]), at the i-th iteration we can find B_{i+1}, the path P_{i+1}, and the vertex v_{i+1} in time $\mathcal{O}(n+m)$. Finally, the procedure eventually terminates as $|V(P)| \leq n$ and thus the bound on the running time follows. ∎

For the remainder of this section, we assume that $g(k) = (k+1)(\lfloor k/2 \rfloor + 1) - 1$, that D is a digraph containing a $(g(k)-1, g(k)-1)$-linked set T of size $2g(k)-1$, consider the T-bramble $\mathcal{B}_T$, and fix P to be the path received by applying Lemma 4.7 with inputs D, T, and $\mathcal{B}_T$. To prove Theorem 2.23, we use the following definition.

DEFINITION 4.8 ((i)-split). *An (i)-split $\mathcal{S}$ of P is a collection formed by a set*

$\{Q_j \mid j \in [i]\}$ of subpaths of P, a subpath P_i of P, a set of brambles $\{\mathcal{B}_j \mid j \in [i]\}$, a set of vertices $\{a_j \mid j \in [i]\}$, and a set of vertices X_i such that

1. *for $j \in [i]$, vertex a_j is the successor in P of the last vertex of Q_j, and, if $j \leq i-1$, the first vertex of Q_{j+1} is the successor in P of vertex a_j,*
2. *for $j \in [i]$, $\mathsf{ord}(\mathcal{B}_j) \geq \lfloor k/2 \rfloor$,*
3. *for $j \in [i]$, $\mathcal{B}_j \subseteq \mathcal{B}_T$ and $V(Q_j)$ is a hitting set of $\mathcal{B}_j$,*
4. *P_i is the subpath of P from the successor in P of the last vertex of Q_i to the last vertex of P, and*
5. *$X_i = \bigcup_{j \in [i]} (V(P_j) \cup \{a_j\})$, and*
6. *the order of $\overline{\mathcal{B}_T}(X_i)$ satisfies*

$$\mathsf{ord}(\overline{\mathcal{B}_T}(X_i)) \geq g(k) - i\left(\left\lfloor \frac{k}{2} \right\rfloor + 1\right).$$

See Fig. 14 for an example of a (2)-split. We remark that a (0)-split for P consists only of the path P_0 with $P_0 = P$ and the empty set X_0.

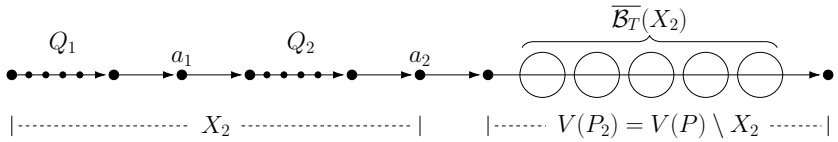

Fig. 14: Illustration of a (2)-split of P. A circle represents an element of the bramble $\overline{\mathcal{B}}(X_2)$.

Now, the proof of Theorem 2.23 follows three steps. First, Lemma 4.9 states that the set of vertices $\{a_1, \ldots, a_i\}$ of an (i)-split of P is well-linked when $i \leq k$. Thus our goal is to construct a (k)-split of P. Then, Lemma 4.10 states that, for $i \geq 0$, we can construct an $(i+1)$-split of P from an (i)-split of P in FPT time if $\mathsf{ord}(\overline{\mathcal{B}}(X_i))$ is large enough. Finally, the proof of Theorem 2.23 starts from a (0)-split of P and iterates Lemma 4.10 until a (k)-split is constructed.

LEMMA 4.9. *Let $\mathcal{S}_i$ be an (i)-split of P with $i \in [k]$. Then the set A with $A = \{a_1, \ldots, a_i\}$ is well-linked in D.*

Proof. Let X and Y be disjoint subsets of A such that $|X| = |Y| = r$ for some $r \in [i]$. Suppose, by contradiction, that there is no set of r pairwise internally disjoint paths from X to Y in D. Then, by Menger's Theorem, there is an (X, Y)-separator $S \subseteq V(D)$ such that $|S| \leq r - 1$.

Let $Q_{i+1} = P_i$ and $\mathcal{B}_{i+1} = \overline{\mathcal{B}}(X_i)$. By the definition of (i)-splits and our choice of Q_{i+1}, we know that for every $a_j \in A$ with $j \in [i]$, Q_j is a path ending on the vertex occurring in P before a_j, and Q_{j+1} is a path starting on the first vertex occurring in P after a_j (see Figure 14 for an example when $i = 2$). Moreover, we have

$$\mathsf{ord}(\mathcal{B}_{i+1}) \geq g(k) - i\left(\left\lfloor \frac{k}{2} \right\rfloor + 1\right)$$

which implies that $\mathsf{ord}(\mathcal{B}_{i+1}) \geq \lfloor k/2 \rfloor$ since $i \leq k$.

As $|S| \leq r - 1 \leq \lfloor k/2 \rfloor - 1$ there is a $j \in [i-1]$ such that $a_j \in X \setminus S$ and $S \cap V(Q_{j+1}) = \emptyset$. Furthermore, since S is not large enough to be a hitting set of

$\mathcal{B}_{j+1}$, there must be $B \in \mathcal{B}_{j+1}$ such that $S \cap V(B) = \emptyset$. Similarly, there are $a_\ell \in Y \setminus S$ and $B' \in \mathcal{B}_\ell$ such that $S \cap V(Q_\ell) = \emptyset$ and $S \cap V(B') = \emptyset$.

By choice, clearly $V(Q_{j+1})$ is a hitting set of $\mathcal{B}_{j+1}$. Since S is disjoint from $V(Q_{j+1}) \cup V(B)$ and $V(B)$ induces a strongly connected subgraph of D, we conclude that there is in $D \setminus S$ a path from a_j to any vertex in $V(B)$. Similarly, there is a path from any vertex in $V(B')$ to a_ℓ in $D \setminus S$. Finally, since every pair of elements in $\mathcal{B}_T$ intersect, we conclude that there is a path in $D \setminus S$ from a_j to a_ℓ using the path Q_{j+1}, the vertices in $V(B) \cup V(B)'$, and the path Q_ℓ. This contradicts our choice of S, and thus we conclude that every (X, Y)-separator in D must have size at least r, and the result follows by Menger's Theorem. ∎

LEMMA 4.10. *Let $\mathcal{S}_i$ be an (i)-split of P with $i \leq k-1$. Then in time $2^{\mathcal{O}(k^2 \log k)} \cdot n^{\mathcal{O}(1)}$ we can construct an $(i+1)$-split of P.*

Proof. For a digraph F, for the sake of notational simplicity we abbreviate –recall Definition 4.3– $\mathcal{B}(V(F))$ and $\overline{\mathcal{B}}(V(F))$ as $\mathcal{B}(F)$ and $\overline{\mathcal{B}}(F)$, respectively, and write $\mathcal{B}(v)$ and $\overline{\mathcal{B}}(v)$ (omitting the braces) for $v \in V(F)$. Let $\mathcal{B}' = \overline{\mathcal{B}}(X_i)$.

The goal is to construct a subpath Q_{i+1} of P starting on the first vertex of P appearing after the vertex a_i (or simply the first vertex of P if $i = 0$) such that

$$\mathsf{ord}(\mathcal{B}'(Q_{i+1})) \geq \left\lfloor \frac{k}{2} \right\rfloor.$$

That is, the order of the bramble containing the elements of $\mathcal{B}$ which are disjoint from X_i while intersecting $V(Q_{i+1})$ is at least $\lfloor k/2 \rfloor$. We start with $V(Q_{i+1}) = \emptyset$. By Inequality 4.1, we have that

$$\mathsf{ord}(\mathcal{B}'(Q_{i+1})) \geq \mathsf{ord}(\mathcal{B}') - \mathsf{ord}(\overline{\mathcal{B}'}(Q_{i+1}))$$

at any point of the procedure. Now, we iteratively grow Q_{i+1}, adding one vertex at a time while testing, at each newly added vertex, whether

$$\mathsf{ord}(\overline{\mathcal{B}'}(Q_{i+1})) \leq g(k) - i\left(\left\lfloor \frac{k}{2} \right\rfloor + 1\right) - 1 - \left\lfloor \frac{k}{2} \right\rfloor.$$

Observe that, when $V(Q_{i+1}) = \emptyset$, we have $\overline{\mathcal{B}'}(Q_{i+1}) = \mathcal{B}'$ and thus

$$\mathsf{ord}(\overline{\mathcal{B}'}(Q_{i+1})) \geq g(k) - i\left(\left\lfloor \frac{k}{2} \right\rfloor + 1\right) > g(k) - i\left(\left\lfloor \frac{k}{2} \right\rfloor + 1\right) - \left\lfloor \frac{k}{2} \right\rfloor.$$

As $\mathcal{B}' = \overline{\mathcal{B}}(X_i)$, we have $\overline{\mathcal{B}'}(Q_{i+1}) = \overline{\mathcal{B}}(X_i \cup V(Q_{i+1}))$ and thus, by Corollary 4.6, we can test whether $\mathsf{ord}(\overline{\mathcal{B}'}(Q_{i+1})) \leq s$ in time $2^{\mathcal{O}(k^2 \log k)} \cdot n^{\mathcal{O}(1)}$ for any $s \in [g(k)]$ since $g(k) = \mathcal{O}(k^2)$.

On a negative answer, we add to Q_{i+1} the first vertex of P not contained in $V(Q_{i+1}) \cup X_i$ and repeat the test. On the first time we obtain a positive answer to this test, we set $\mathcal{B}_{i+1} = \mathcal{B}'(Q_{i+1})$, define a_{i+1} to be the first vertex appearing in P after the last vertex of Q_{i+1}, and stop the procedure. In this case, we have that $\mathsf{ord}(\mathcal{B}_{i+1}) \geq \lfloor k/2 \rfloor$ and since $\mathsf{ord}\left(\overline{\mathcal{B}'}(Q_{i+1})\right)$ can decrease by at most one each time we increase by one the size of $V(Q_{i+1})$, this procedure actually ends with $\mathsf{ord}(\overline{\mathcal{B}'}(Q_{i+1})) = g(k) - i(\lfloor k/2 \rfloor + 1) - \lfloor k/2 \rfloor$.

Now, we define $X_{i+1} = X_i \cup V(Q_{i+1}) \cup \{a_{i+1}\}$ and P_{i+1} to be the subpath of P with $V(P_{i+1}) = V(P) \setminus X_{i+1}$. Finally, let $\mathcal{B}^* = \overline{\mathcal{B}'}(Q_{i+1})$. Then by Inequality 4.1,

$$\mathsf{ord}(\mathcal{B}^*(P_{i+1})) \geq \mathsf{ord}(\mathcal{B}^*) - \mathsf{ord}(\overline{\mathcal{B}^*}(P_{i+1}))$$

and observing that $\mathcal{B}^*(P_{i+1}) = \overline{\mathcal{B}}(X_{i+1})$, we conclude that

$$\mathsf{ord}(\overline{\mathcal{B}}(X_{i+1})) \geq g(k) - i\left(\left\lfloor \frac{k}{2} \right\rfloor + 1\right) - \left\lfloor \frac{k}{2} \right\rfloor - 1 = g(k) - (i+1)\left(\left\lfloor \frac{k}{2} \right\rfloor + 1\right),$$

as required, since $\overline{\mathcal{B}^*}(P_{i+1}) = \overline{\mathcal{B}^*}(a_{i+1})$ and thus $\mathsf{ord}(\overline{\mathcal{B}^*}(P_{i+1})) \leq 1$. Then, we output the $(i+1)$-split $\mathcal{S}_{i+1}$ of P_i formed by the sequence of paths $Q_1, \ldots, Q_{i+1}$, the path P_{i+1}, the sequence of brambles $\mathcal{B}_1, \ldots, \mathcal{B}_{i+1}$, the set of vertices $\{a_1, \ldots, a_{i+1}\}$, and the set of vertices X_{i+1}. ∎

We remark that the bramble $\overline{\mathcal{B}'}(Q_{i+1})$ is used only in the proof of Lemma 4.9 and thus we do not need to maintain it during the algorithm. However, if we want to store this information, it suffices to maintain the set T, the set X_i, and the path Q_{i+1} since the bramble $\overline{\mathcal{B}'}(Q_{i+1})$ is equal to the bramble $\mathcal{B}(Q_{i+1})$ in the digraph $D \setminus X_i$. We are now ready to prove Theorem 2.23.

THEOREM 2.23 (Second main contribution). *Let $g(k) = (k+1)(\lfloor k/2 \rfloor + 1) - 1$, D be a digraph and T be a $(g(k)-1, g(k)-1)$-linked set in D with $|T| = 2g(k)-1$. There is an algorithm running in time $2^{\mathcal{O}(k^2 \log k)} \cdot n^{\mathcal{O}(1)}$ that finds in D a bramble $\mathcal{B}$ of order $g(k)$, a path P that is a hitting set of $\mathcal{B}$, and a well-linked set A of order k such that $A \subseteq V(P)$.*

Proof. By Lemma 4.2, the T-bramble $\mathcal{B}_T$ has order $g(k)$ and, by Lemma 4.7, we can find a path P that is a hitting set of $\mathcal{B}_T$ in polynomial time. We start with a trivial (0)-split $\mathcal{S}_0$ of P where $P_0 = P$ and $X_0 = \emptyset$.

For $i \in \{0, \ldots, k-1\}$, we apply Lemma 4.10 with input $\mathcal{S}_i$ to obtain an $(i+1)$-split $\mathcal{S}_{i+1}$ of P in time $2^{\mathcal{O}(k^2 \log k)} \cdot n^{\mathcal{O}(1)}$. After the last iteration, we obtained a (k)-split $\mathcal{S}_k$ of P and, by Lemma 4.9, the set of vertices $\{a_1, \ldots, a_k\}$ of $\mathcal{S}_k$ is well-linked in D and all such vertices are in $V(P)$, as desired. ∎

By following the remainder of the proof of the Directed Grid Theorem [38], which yields FPT algorithms for all the remaining steps (see Section 2.5), we can validate Corollary 2.24.

5. Concluding remarks. The main consequence of our results is an FPT algorithm with parameter k that either produces an arboreal decomposition of width at most $f(k)$ for a digraph D or constructs a cylindrical grid of order k as a butterfly minor of D, for some computable function $f(k)$. This is achieved by adapting some of the steps used in the proof of the Directed Grid Theorem from Kawarabayashi and Kreutzer [38].

For the first possible output of this algorithm, we improve on a result from [35] by providing an FPT algorithm with parameter k that either produces an arboreal decomposition of a digraph D with width at most $3k-2$, or concludes that D has a haven of order k. As a tool to prove this result, we consider a generalization of the problem of finding balanced separators in digraphs (we remind the reader that our definition of balanced separators extends the classical definition that can be found, for example, in [45]) and show how to solve it in FPT time with parameter $|T|$. Since in the undirected case balanced separators are strongly related to the tree-width of undirected graphs, and the only result for balanced separators in the directed case considered only a relaxed version of the problem (see [45, Chapter 6]), we consider this result to be of its own interest.

Although it is possible to construct a bramble $\mathcal{B}$ of order $\lfloor k/2 \rfloor$ from a haven of order k, this construction is not *efficient* in general, in the sense that we must

go through all elements of $\mathcal{B}$ to verify whether a given set X is a hitting set of $\mathcal{B}$. Motivated by this, we consider a definition of brambles, which we call T-brambles, which naturally occur in digraphs of large directed tree-width that are better to work with in a number of ways. For instance, by reducing to the problem of computing (T, r)-balanced separators for T, we show how to compute hitting sets of T-brambles in FPT time when parameterized by $|T|$.

We use our results for T-brambles in digraphs of large tree-width to show how to find, in FPT time with parameter k, a path that is a hitting set of a T-bramble $\mathcal{B}_T$ of order $(k+1)(\lfloor k/2 \rfloor + 1)$ and a well-linked set of size k that is contained in this path. This is the second step that we change in the proof of the Directed Grid Theorem [38]. From this point forward, the remaining steps in the proof yield FPT algorithms.

Kreutzer and Ordyniak [42] and Ganian et al. [30] showed that many important problems in digraphs remain hard when restricted to digraphs of bounded directed tree-width. In particular, Kreutzer and Ordyniak [42] showed that the Directed Feedback Vertex Set (DFVS) problem is NP-complete even when restricted to digraphs of directed tree-width at most five. However, some open problems in digraphs may benefit from an approach resembling Bidimensionality using our FPT algorithm for the Directed Grid Theorem. For example, Bezáková et al. [7] asked whether the Longest Detour problem in digraphs could be solved by using the Directed Grid Theorem. To provide more potential applicability of our results, we briefly discuss the parameterized tractability of the DFVS problem.

Chen et al. [15] provided an algorithm running in time $2^{\mathcal{O}(k \log k)} \cdot n^{\mathcal{O}(1)}$ for the DFVS problem, where k is the size of the solution. Bonamy et al. [9] showed that, when parameterized by the tree-width t of the underlying graph, DFVS is solvable in time $2^{\mathcal{O}(t \log t)} \cdot n^{\mathcal{O}(1)}$ in general digraphs and the dependency on the parameter is improved to $2^{\mathcal{O}(t)}$ when restricted to planar digraphs. When parameterized by the feedback vertex set number of the underlying graph, Bergougnoux et al. [5] showed that DFVS admits a polynomial kernel in general digraphs, and a linear kernel in digraphs that are embeddable on surfaces of bounded genus.

On the one hand, DFVS remains hard even when restricted to digraphs of directed tree-width at most five [42], but on the other hand both of the aforementioned parameters related to the underlying graph are individually stronger than the directed tree-width of the input digraph and, by the Directed Grid Theorem [38], every positive instance of DFVS parameterized by the size k of the solution occurs in a digraph of bounded directed tree-width: since a cylindrical grid of order r contains a set of r vertex-disjoint cycles and butterfly contractions do not generate new paths, the minimum size of a feedback vertex set of a digraph D is at least the order of the largest cylindrical grid that is as a butterfly minor of D. Now, by Corollary 2.24, in FPT time with parameter k we can either find a certificate that the considered instance of DFVS is negative (a cylindrical grid of order $k+1$ that is a butterfly minor of the input digraph), or produce an arboreal decomposition of the input digraph of width at most $f(k)$, for some computable function $f : \mathbb{N} \to \mathbb{N}$.

Thus, it is sensible to ask whether similar or improved results for DFVS (when parameterized by the tree-width or the feedback vertex set number of the underlying graph, as previously mentioned) can be proved if we consider that the input digraph has bounded directed tree-width, since by the above discussion we can restrict instances of DFVS to this class of digraphs.

One could also consider the tractability of hard problems in digraphs of bounded directed tree-width under stronger parameterizations. For example, Lopes and Sau [43] recently showed that a relaxation for the Directed Disjoint Paths problem,

a notoriously hard problem in digraphs, admits a kernelization algorithm for some choices of parameters. In this spirit, it seems plausible that combining directed treewidth with other parameters may lead to FPT algorithms for hard problems, and in this context the FPT algorithm presented in this paper may become handy.

It is worth mentioning that Giannopoulou et al. [31] recently provided an analogous version of the Flat Wall Theorem [50] for directed graphs, which may have interesting algorithmic applications when combined with our results.

Finally, the attempts to obtain a Bidimensionality theory for directed graphs, such as the one presented by Dorn et al. [23], are so far less satisfying that the undirected version, from the point of view of generality and efficiency of the obtained algorithms. We hope that our FPT version of the Directed Grid Theorem will have a relevant role in an eventual Bidimensionality theory for directed graphs.

REFERENCES

[1] S. A. Amiri, K. Kawarabayashi, S. Kreutzer, and P. Wollan, *The Erdős-Pósa property for directed graphs*, 2016, https://arxiv.org/abs/1603.02504v3.

[2] S. Arnborg, D. G. Corneil, and A. Proskurowski, *Complexity of finding embeddings in a k-tree*, SIAM Journal on Algebraic Discrete Methods, 8 (1987), pp. 277–284, https://doi.org/10.1137/0608024.

[3] J. Bang-Jensen and G. Gregory, *Classes of Directed Graphs*, Springer Monographs in Mathematics, 2018, https://doi.org/10.1007/978-3-319-71840-8.

[4] E. T. Bell, *Exponential polynomials*, Annals of Mathematics, 35 (1934), pp. 258–277, http://www.jstor.org/stable/1968431.

[5] B. Bergougnoux, E. Eiben, R. Ganian, S. Ordyniak, and M. S. Ramanujan, *Towards a Polynomial Kernel for Directed Feedback Vertex Set*, in Proc. of the 42nd International Symposium on Mathematical Foundations of Computer Science (MFCS), vol. 83 of LIPIcs, 2017, pp. 36:1–36:15, https://doi.org/10.4230/LIPIcs.MFCS.2017.36.

[6] U. Bertele and F. Brioschi, *Nonserial Dynamic Programming*, Academic Press, Inc., Orlando, FL, USA, 1972.

[7] I. Bezáková, R. Curticapean, H. Dell, and F. Fomin, *Finding detours is fixed-parameter tractable*, SIAM Journal on Discrete Mathematics, 33 (2016), pp. 2326–2345, https://doi.org/10.1137/17M1148566.

[8] H. L. Bodlaender, *Treewidth: characterizations, applications, and computations*, in Proc. of the 32nd International Workshop on Graph-Theoretic Concepts in Computer Science (WG), vol. 4271 of LNCS, 2006, pp. 1–14, https://doi.org/10.1007/11917496_1.

[9] M. Bonamy, L. Kowalik, J. Nederlof, M. Pilipczuk, A. Socala, and M. Wrochna, *On directed feedback vertex set parameterized by treewidth*, in Proc. of the 44th Graph-Theoretic Concepts in Computer Science (WG), vol. 11159 of LNCS, 2018, pp. 65–78, https://doi.org/10.1007/978-3-030-00256-5_6.

[10] A. Bondy and M. R. Murty, *Graph Theory*, Springer-Verlag London, 2008.

[11] N. Bousquet, J. Daligault, and S. Thomassé, *Multicut is FPT*, SIAM Journal on Computing, 47 (2018), pp. 166–207, https://doi.org/10.1137/140961808.

[12] A. Cayley, *On the analytical forms called trees. Second part*, vol. 4 of Cambridge Library Collection - Mathematics, Cambridge University Press, 2009, p. 112–115, https://doi.org/10.1017/CBO9780511703706.026.

[13] C. Chekuri and J. Chuzhoy, *Polynomial bounds for the grid-minor theorem*, Journal of the ACM, 63 (2016), pp. 40:1–40:65, https://doi.org/10.1145/2820609.

[14] C. Chekuri, A. Ene, and M. Pilipczuk, *Constant congestion routing of symmetric demands in planar directed graphs*, in Proc. of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP), vol. 55 of LIPIcs, 2016, pp. 7:1–7:14, https://doi.org/10.4230/LIPIcs.ICALP.2016.7.

[15] J. Chen, Y. Liu, S. Lu, B. O'Sullivan, and I. Razgon, *A fixed-parameter algorithm for the directed feedback vertex set problem*, Journal of the ACM, 55 (2008), pp. 21:1–21:19, https://doi.org/10.1145/1374376.1374404.

[16] J. Chuzhoy and Z. Tan, *Towards Tight(er) Bounds for the Excluded Grid Theorem*, in Proc. of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2019, pp. 1445–1464, https://doi.org/10.1137/1.9781611975482.88.

[17] W. Cook and P. D. Seymour, *Tour merging via branch-decompositions*, INFORMS Journal on Computing, 15 (2003), pp. 233–248, https://doi.org/10.1287/ijoc.15.3.233.16078.
[18] B. Courcelle, *The monadic second-order logic of graphs. I. Recognizable sets of finite graphs*, Information and Computation, 85 (1990), pp. 12–75, https://doi.org/10.1016/0890-5401(90)90043-H.
[19] M. Cygan, F. V. Fomin, L. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh, *Parameterized Algorithms*, Springer, 2015.
[20] M. Cygan, D. Marx, M. Pilipczuk, and M. Pilipczuk, *The Planar Directed k-Vertex-Disjoint Paths Problem is Fixed-Parameter Tractable*, in Proc. of the IEEE 54th Annual Symposium on Foundations of Computer Science (FOCS), 2013, pp. 197–206, https://doi.org/10.1109/FOCS.2013.29.
[21] M. de Oliveira Oliveira, *An algorithmic metatheorem for directed treewidth*, Discrete Applied Mathematics, 204 (2016), pp. 49–76, https://doi.org/10.1016/j.dam.2015.10.020.
[22] D. Demaine, V. Fomin, M. Hajiaghayi, and D. M. Thilikos, *Subexponential parameterized algorithms on bounded-genus graphs and H-minor-free graphs*, Journal of the ACM, 52 (2005), pp. 866–893, https://doi.org/10.1145/1101821.1101823.
[23] F. Dorn, F. V. Fomin, D. Lokshtanov, V. Raman, and S. Saurabh, *Beyond bidimensionality: Parameterized subexponential algorithms on directed graphs*, Information and Computation, 233 (2013), pp. 60–70, https://doi.org/10.1016/j.ic.2013.11.006.
[24] R. G. Downey and M. R. Fellows, *Fundamentals of Parameterized Complexity*, Texts in Computer Science, Springer, 2013.
[25] K. Edwards, I. Muzi, and P. Wollan, *Half-integral linkages in highly connected directed graphs*, in Proc. of the 25th Annual European Symposium on Algorithms (ESA), vol. 87 of LIPIcs, 2017, pp. 36:1–36:12, https://doi.org/10.4230/LIPIcs.ESA.2017.36.
[26] R. F. Erbacher, T. Jaeger, N. Talele, and J. Teutsch, *Directed multicut with linearly ordered terminals*, July 2014, https://arxiv.org/abs/1407.7498v1.
[27] J. Flum and M. Grohe, *Parameterized Complexity Theory*, Springer, 2006.
[28] F. V. Fomin, E. D. Demaine, M. T. Hajiaghayi, and D. M. Thilikos, *Bidimensionality*, in Encyclopedia of Algorithms, 2016, pp. 203–207, https://doi.org/10.1007/978-1-4939-2864-4_47.
[29] S. Fortune, J. Hopcroft, and J. Wyllie, *The directed subgraph homeomorphism problem*, Theoretical Computer Science, 10 (1980), pp. 111–121, https://doi.org/10.1016/0304-3975(80)90009-2.
[30] R. Ganian, P. Hliněný, J. Kneis, A. Langer, J. Obdržálek, and P. Rossmanith, *Digraph width measures in parameterized algorithmics*, Discrete Applied Mathematics, 168 (2014), pp. 88–107, https://doi.org/10.1016/j.dam.2013.10.038.
[31] A. C. Giannopoulou, K. Kawarabayashi, S. Kreutzer, and O. Kwon, *The Directed Flat Wall Theorem*, in Proc. of the 13st ACM-SIAM Symposium on Discrete Algorithms (SODA), 2020, pp. 239–258, https://doi.org/10.1137/1.9781611975994.15.
[32] M. Grohe, K.-i. Kawarabayashi, D. Marx, and P. Wollan, *Finding topological subgraphs is fixed-parameter tractable*, in Proc. of the 43rd Annual ACM Symposium on Theory of Computing (STOC), 2011, pp. 479–488, https://doi.org/10.1145/1993636.1993700.
[33] R. Halin, *S-functions for graphs*, Journal of Geometry, 8 (1976), pp. 171–186, https://doi.org/10.1007/BF01917434.
[34] M. Hatzel, K. Kawarabayashi, and S. Kreutzer, *Polynomial Planar Directed Grid Theorem*, in Proc. of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2019, pp. 1465–1484, https://doi.org/10.1137/1.9781611975482.89.
[35] T. Johnson, N. Robertson, P. D. Seymour, and R. Thomas, *Directed tree-width*, Journal of Combinatorial Theory, Series B, 82 (2001), pp. 138–154, https://doi.org/10.1006/jctb.2000.2031.
[36] T. Johnson, N. Robertson, P. D. Seymour, and R. Thomas, *Excluding a grid minor in planar digraphs*, Oct. 2015, https://arxiv.org/abs/1510.00473v1.
[37] K. Kawarabayashi and S. Kreutzer, *An Excluded Grid Theorem for Digraphs with Forbidden Minors*, in Proc. of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), 2014, pp. 72–81, https://doi.org/10.1137/1.9781611973402.6.
[38] K.-i. Kawarabayashi and S. Kreutzer, *The Directed Grid Theorem*, in Proc. of the 47th Annual ACM Symposium on Theory of Computing (STOC), 2015, pp. 655–664, https://doi.org/10.1145/2746539.2746586.
[39] J. M. Kleinberg, *Decision algorithms for unsplittable flow and the half-disjoint paths problem*, in Proc. of the 30th Annual ACM Symposium on Theory of Computing (STOC), 1998, pp. 530–539, https://doi.org/10.1145/276698.276867.
[40] A. Koster, S. van Hoesel, and A. Kolen, *Solving frequency assignment problems via tree-*

decomposition, Electronic Notes in Discrete Mathematics, 3 (1999), pp. 102–105, https://doi.org/10.1016/S1571-0653(05)80034-4.

[41] S. Kratsch, M. Pilipczuk, M. Pilipczuk, and M. Wahlström, *Fixed-parameter tractability of multicut in directed acyclic graphs*, SIAM Journal on Discrete Mathematics, 29 (2015), pp. 122–144, https://doi.org/10.1137/120904202.

[42] S. Kreutzer and S. Ordyniak, *Digraph decompositions and monotonicity in digraph searching*, Theoretical Computer Science, 412 (2011), pp. 4688–4703, https://doi.org/https://doi.org/10.1016/j.tcs.2011.05.003.

[43] R. Lopes and I. Sau, *A relaxation of the Directed Disjoint Paths problem: a global congestion metric helps*, in Proc. of the 45th International Symposium on Mathematical Foundations of Computer Science (MFCS), vol. 170 of LIPIcs, 2020, pp. 66:1–66:15, https://doi.org/10.4230/LIPIcs.MFCS.2020.68.

[44] D. Marx and I. Razgon, *Fixed-parameter tractability of multicut parameterized by the size of the cutset*, SIAM Journal on Computing, 43 (2014), pp. 355–388, https://doi.org/10.1137/110855247.

[45] D. Matthias and F. Emmert-Streib, *Quantitative Graph Theory: Mathematical Foundations and Applications*, Discrete Mathematics and its Applications, Chapman and Hall/CRC, 2014.

[46] K. Menger, *Zur allgemeinen kurventheorie*, Fundamenta Mathematicae, 10 (1927), pp. 96–115, http://eudml.org/doc/211191.

[47] M. Pilipczuk and M. Wahlström, *Directed Multicut is W[1]-hard, Even for Four Terminal Pairs*, ACM Transactions on Computation Theory, 10 (2018), pp. 13:1–13:18, http://doi.acm.org/10.1145/3201775.

[48] B. Reed, *Introducing directed tree-width*, Electronic Notes in Discrete Mathematics, 3 (1999), pp. 222–229, https://doi.org/10.1016/S1571-0653(05)80061-7.

[49] N. Robertson and P. D. Seymour, *Graph minors. V. Excluding a planar graph*, Journal of Combinatorial Theory, Series B, 41 (1986), pp. 92–114, https://doi.org/10.1016/0095-8956(86)90030-4.

[50] N. Robertson and P. D. Seymour, *Graph minors. XIII. The disjoint paths problem*, Journal of Combinatorial Theory, Series B, 63 (1995), pp. 65–110, https://doi.org/10.1006/jctb.1995.1006.

[51] N. Robertson and P. D. Seymour, *Graph minors. XXI. Graphs with unique linkages*, Journal of Combinatorial Theory, Series B, 99 (2009), pp. 583–616, https://doi.org/10.1016/j.jctb.2008.08.003.

[52] N. Robertson and P. D. Seymour, *Graph Minors. XXII. Irrelevant vertices in linkage problems*, Journal of Combinatorial Theory, Series B, 102 (2012), pp. 530–563, https://doi.org/10.1016/j.jctb.2007.12.007.

[53] P. D. Seymour and R. Thomas, *Graph searching and a min-max theorem for tree-width*, Journal of Combinatorial Theory, Series B, 58 (1993), pp. 22–33, https://doi.org/10.1006/jctb.1993.1027.

[54] A. Slivkins, *Parameterized tractability of edge-disjoint paths on directed acyclic graphs*, SIAM Journal on Discrete Mathematics, 24 (2010), pp. 146–157, https://doi.org/10.1137/070697781.