TheoretiCS

Directed Acyclic Outerplanar Graphs Have Constant Stack Number

Received Apr 23, 2024
Revised May 30, 2025
Accepted Jul 26, 2025
Published Oct 17, 2025

Key words and phrases
Linear layouts, stack number, directed graphs, outerplanar graphs, 2-trees

Paul Jungeblut [a]
Laura Merker [a]
Torsten Ueckerdt [a]

a Karlsruhe Institute of Technology, Germany

ABSTRACT. The stack number of a directed acyclic graph G is the minimum k for which there is a topological ordering of G and a k-coloring of the edges such that no two edges of the same color cross, i.e., have alternating endpoints along the topological ordering. We prove that the stack number of directed acyclic outerplanar graphs is bounded by a constant, which gives a positive answer to a conjecture by Heath, Pemmaraju and Trenk [SIAM J. Computing, 1999]. As an immediate consequence, this shows that all upward outerplanar graphs have constant stack number, answering a question by Bhore et al. [Eur. J. Comb., 2023] and thereby making significant progress towards the problem for general upward planar graphs originating from Nowakowski and Parker [Order, 1989]. As our main tool we develop the novel technique of directed H-partitions, which might be of independent interest.

We complement the bounded stack number for directed acyclic outerplanar graphs by constructing a family of directed acyclic 2-trees that have unbounded stack number, thereby refuting a conjecture by Nöllenburg and Pupyrev [GD 2023].

1. Introduction

A directed acyclic graph, shortly DAG, is a directed graph with no directed cycles. For an integer $k \geq 1$ and a directed acyclic graph G, a *k-stack layout* of G consists of a topological ordering $\prec$ of the vertices $V(G)$ and a partition of the edges $E(G)$ into k parts such that each part is a *stack*. A part is a stack if no two of its edges cross with respect to $\prec$, where two edges ab and cd *cross*

A preliminary version of this paper is published in the proceedings of the 64th Annual Symposium on Foundations of Computer Science (FOCS) © 2023 IEEE [**43**].

Cite as Paul Jungeblut, Laura Merker, Torsten Ueckerdt. Directed Acyclic Outerplanar Graphs Have Constant Stack Number. TheoretiCS, Volume 4 (2025), Article 25, 1-33.

https://theoretics.episciences.org
DOI 10.46298/theoretics.25.25

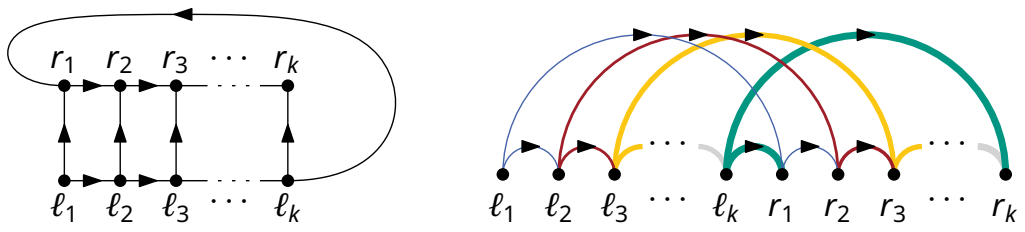


Figure 1. A planar directed acyclic graph G_k of treewidth 3 on $2k$ vertices (left) and its unique topological ordering $\prec$ containing a k-twist (right).

if their endpoints are ordered $a \prec c \prec b \prec d$ or $c \prec a \prec d \prec b$. Now the *stack number*[1] $\mathrm{sn}(G)$ of G is the minimum k such that there exists a k-stack layout of G. It is convenient to interpret the ordering $\prec$ as the vertices of G being laid out from left to right, i.e., for $a \prec b$ we say that a is to the left of b (and b is to the right of a) or that a comes before b in $\prec$ (and b comes after a in $\prec$). Then $\prec$ being a topological ordering of G means that for every edge ab directed from a to b, vertex a must come before vertex b in $\prec$, in other words, every edge is directed from its left to its right endpoint according to $\prec$.

Given a topological ordering $\prec$ of G, a partition of $E(G)$ into k stacks can equivalently be seen as a k-edge coloring, such that each color class is crossing-free. The simplest obstruction to admitting a partition into $k-1$ stacks is a set of k pairwise crossing edges, also called a *k-twist*. For example, consider the $2k$-vertex graph G_k, $k \geq 2$, in the left of Figure 1 consisting of the directed path $P = (\ell_1, \ldots, \ell_k, r_1, \ldots, r_k)$ and the matching $M = \{\ell_i r_i \mid i = 1, \ldots, k\}$. This directed acyclic graph has only one topological ordering $\prec$ in which the vertices are ordered along the directed path P. However with respect to this ordering $\prec$ the edges in M form a k-twist. It follows that $\mathrm{sn}(G_k) \geq k$ for all $k \geq 2$, which is also tight, as we can easily find a partition of $E(G_k)$ into k stacks as indicated in the right of Figure 1.

Evidently, for every k the graph G_k is planar. Moreover, G_k is 2-degenerate and has treewidth[2] at most 3. So this family of DAGs shows that the stack number is not bounded within the class of all planar directed acyclic graphs; not even within those that are 2-degenerate and have treewidth at most 3 (even pathwidth at most 3). On the other hand, as already noted by Nowakowski and Parker [52] as well as Heath, Pemmaraju and Trenk [39], it is easy to verify that if G is a directed forest (equivalently, if G is 1-degenerate, or if G has treewidth 1), then $\mathrm{sn}(G) = 1$. However, determining the largest stack number among the class of DAGs of treewidth 2, in particular in the special case of outerplanar graphs, has remained an intriguing open problem for several decades [39, 23, 51, 9, 17]. In fact there was little progress even for the considerably smaller class of outerplanar DAGs, for which a well-known conjecture of Heath, Pemmaraju and Trenk [39] from 1999 states that the stack number should be bounded.

1 The stack number is also known as *book thickness* or *page number*, especially in the older literature (stacks are called pages then). In recent years the term stack number seems to be preferred over the others as it explicitly expresses the first-in-last-out property of every part, in alignment with related concepts like queue layouts.

2 We do not need the formal definition of treewidth here but we define related relevant concepts like 2-trees in Section 2.

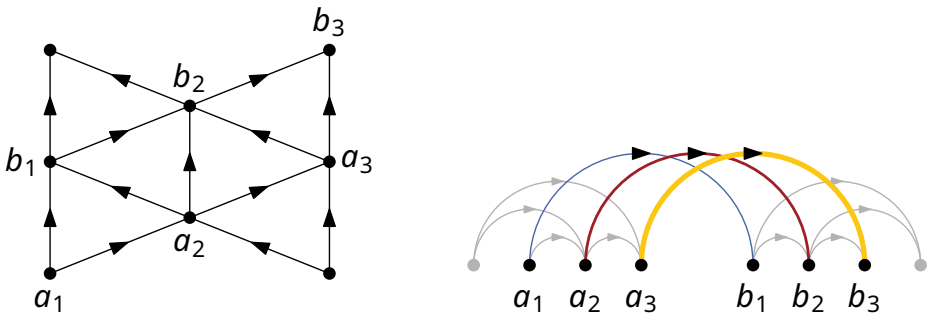


Figure 2. An outerplanar directed acyclic graph on eight vertices (left) and a topological ordering $\prec$ with a_3 to the left of b_1 containing a 3-twist (right).

CONJECTURE 1.1 (Heath, Pemmaraju and Trenk, 1999 [39]). *The stack number of the class of directed acyclic outerplanar graphs is bounded above by a constant.*

For the sake of an example, let us argue that the outerplanar DAG G in Figure 2 has $\mathrm{sn}(G) \geq 3$. In fact, due to symmetry it is enough to consider a topological ordering $\prec$ with a_3 to the left of b_1 and to observe that in $\prec$ the edges a_1b_1, a_2b_2, a_3b_3 form a 3-twist. Conjecture 1.1 concerning outerplanar DAGs, but also the more general question about directed acyclic 2-trees, have received increasing attention over the last years (see Section 1.1 for more details), but it remained open to this day whether either of these classes contains DAGs of arbitrarily large stack number.

In addition to Conjecture 1.1, there is a second major open question in the field of directed linear layouts which we attack. Here, a DAG is called *upward planar* if it can be drawn in the plane such that the edges are crossing-free and y-monotone.

OPEN PROBLEM 1.2 (Nowakowski, Parker, 1989 [52]). *Is the stack number of the class of upward planar graphs bounded above by a constant?*

Our results. In this paper, we answer both long-standing open problems connected to Conjecture 1.1, that is whether or not outerplanar graphs and 2-trees have bounded stack number. We answer the first question in the positive, the second in the negative. Additionally, we attack Open Problem 1.2 and contribute a significant class of upward planar graphs with bounded stack number.

In Section 3, we prove that Conjecture 1.1 is true by showing that outerplanar DAGs have stack number at most 24776. As one of our main tools we introduce *directed H-partitions*, which may be of independent interest for investigations of directed graphs in general. We remark that while another variant of H-partitions, so called *layered H-partitions*, caused a breakthrough in the investigation of queue layouts (a notion closely related to stack layouts) [25], this is – to the best of our knowledge – the first time that H-partitions are successfully adjusted to work with stack layouts.

THEOREM 1.3. *The stack number of outerplanar DAGs is bounded by a constant. Moreover, every outerplanar DAG G has* $\mathrm{sn}(G) \leq 24776$.

Our second main result complements the constant upper bound for outerplanar DAGs by showing that already a just slightly larger subclass of graphs of treewidth 2 has unbounded stack number.[3]

THEOREM 1.4. *The stack number of DAGs of treewidth* 2 *is unbounded. Moreover, for every* $k \geq 1$ *there exists a monotone* 2*-tree* G *with* $\mathrm{sn}(G) \geq k$ *in which at most two vertices are stacked onto each edge.*

We remark that Theorems 1.3 and 1.4 together give a quite good understanding of which 2-trees have bounded stack number as stacking at most one vertex onto each edge yields exactly the maximal outerplanar DAGs. In 2006, Di Giacomo, Didimo, Liotta and Wismath [23] asked whether or not all DAGs with treewidth 2 admit a 2-stack layout. (The DAG in Figure 2 is already a counterexample.) ¡Most recently, Nöllenburg and Pupyrev [51] highlight whether or not all directed acyclic 2-trees have bounded stack number as an important open question. They as well as a Dagstuhl report by Bekos et al. [9] even conjecture that the stack number of all such 2-trees should indeed be bounded. Our Theorem 1.4 refutes this conjecture.

Let us also emphasize that Theorem 1.3 in particular gives that upward outerplanar graphs have bounded stack number. This was known only for specific subclasses before, such as internally triangulated upward outerpaths [17]. As such, Theorem 1.3 provides one of the largest known classes of upward planar graphs with bounded stack number, while it is a famous open problem whether or not actually all upward planar graphs have bounded stack number [52].

Organization of the paper. Before proving Theorem 1.3 in Section 3 and Theorem 1.4 in Section 4, we define in Section 2 all concepts and notions relevant for our proofs. This includes a quick reminder of 2-trees, outerplanar graphs, stack layouts, and twist numbers, but also some specialized notions for directed acyclic 2-trees, such as monotone and transitive vertices, block-monotone DAGs, or transitive subgraphs below monotone vertices. Directed H-partitions are introduced in Section 3.2. But first, let us review related work.

1.1 Related Work

Stack layouts are just one type of so-called *linear layouts* which have been an active field of study over at least the last forty years. In full generality, a linear layout of an (undirected) graph G consists of a total ordering $\prec$ of the vertices $V(G)$ and a partition of the edges $E(G)$ into parts such that each part fulfills certain combinatorial properties. For directed acyclic graphs the vertex ordering $\prec$ must be a topological ordering. The most prominent types of linear layouts are *stack layouts* (no two edges of the same part cross) and *queue layouts* (no two edges of the

3 We remark that every 2-tree can be constructed by iteratively *stacking* vertices onto an edge, i.e., by introducing a new vertex that is connected exactly to the endpoints of an edge. A directed 2-tree is called *monotone* if in each step the new vertex is a source or a sink. See also Section 2.

same part nest, i.e., two edges ab and cd ordered $a \prec c \prec d \prec b$ are forbidden) [40, 36, 25]. These are accompanied by a rich field of further variants like *mixed layouts* (a combination of stack and queue layouts) [40, 55, 6, 2, 21, 29], *track layouts* [26, 8, 54], deque layouts [7, 13, 15], rique layouts [11, 13, 15], *local* variants thereof [50, 49, 28, 6] and more [27, 1].

Let us content ourselves with just briefly summarizing below (some of) those previous results on stack layouts that concern undirected or directed planar graphs.

Stack layouts of undirected planar graphs. Building on the notion of Kainen and Ollmann [44, 53], the stack number for undirected graphs was first investigated by Bernhart and Kainen [16] in 1979. Besides giving bounds for complete and complete bipartite graphs, they conjecture that there are planar graphs with arbitrarily large stack number. This conjecture was refuted in [19, 35] leading eventually to an improved upper bound of 4 [63], which has only recently been shown to be tight [14, 64].

It is well-known that a graph with at least one edge admits a 1-stack layout if and only if it is outerplanar, and that it admits a 2-stack layout if and only if it is planar sub-Hamiltonian (i.e., is a subgraph of a planar graph containing a Hamiltonian cycle) [16]. Testing whether a graph is planar sub-Hamiltonian is NP-complete [61], but there has been significant effort to identify planar Hamiltonian and sub-Hamiltonian graph classes: These include (among others) planar bipartite graphs [22], planar graphs with maximum degree at most 4 [12], planar 4-connected graphs [59], planar 3-connected graphs with maximum degree 5, and 2-trees [56]. Further, three stacks are sufficient for planar 3-trees [35] and planar graphs with maximum degree 5 [32]. Recall that four stacks are always sufficient [63] and sometimes necessary [14, 64] for all planar graphs.

Among the numerous results on non-planar graphs G, there are bounds on $\mathrm{sn}(G)$ depending on the Euler genus [46], the pathwidth [58] and the treewidth [31, 24, 60] of G.

Stack layouts of planar directed acyclic graphs. For directed acyclic graphs we additionally require the vertex ordering $\prec$ to be a topological ordering. Nowakowski and Parker [52] were the first to study this and consider stack layouts of diagrams of posets. Presenting an example with stack number 3 (which was later improved to 4 by Hung [41] and to 5 by Merker [48]), they ask whether posets with a planar diagram[4] have bounded stack number. Despite significant effort on different subclasses [57, 4, 38, 37, 39, 5, 3], this question still remains open.

A slight generalization, namely whether all upward planar graphs have bounded stack number, is considered to be one of the most important open questions in the field of linear layouts [30, 51, 42]. It is known to hold for upward planar 3-trees [30, 51]. However, this does not imply the same for upward planar DAGs of treewidth at most 3, since upward planar partial

4 In planar diagrams (generally: upward planar drawings) all edges must be drawn y-monotone along their edge direction. For example, the graph in Figure 2 is upward planar, while the graph G_k in Figure 1 admits no upward planar drawing due to the edge from vertex ℓ_k to vertex r_1.

3-trees might not have an upward planar 3-tree as a supergraph. (E.g., Figure 2 depicts such an example.) Superseding previous results [30], Jungeblut, Merker and Ueckerdt [42] recently gave the first sublinear upper bound for all upward planar graphs by showing that every n-vertex upward planar graph G has stack number $\mathrm{sn}(G) \leq O((n \log n)^{2/3}) = o(n)$.

For general planar DAGs (that are not necessarily upward planar), Heath, Pemmaraju and Trenk [39] show that directed trees admit 1-stack layouts and directed unicyclic graphs admit 2-stack layouts. Other classes of DAGs admitting 2-stack layouts include two-terminal series-parallel graphs [23], N-free graphs [47] or planar graphs whose faces have a special structure [18]. Recall from the example in Figure 1 that the stack number can be linear in the number of vertices, already for planar DAGs of treewidth 3. With this in mind, Heath, Pemmaraju and Trenk [39] formulated Conjecture 1.1 in 1999.

Conjecture 1.1 has received considerable attention, especially in recent years. Bhore, Da Lozzo, Montecchiani and Nöllenburg [17] confirm Conjecture 1.1 for several subclasses of outerplanar DAGs, including internally triangulated upward outerpaths or cacti. Subsequently, Nöllenburg and Pupyrev [51] confirm Conjecture 1.1 for single-source outerplanar DAGs, monotone outerplanar DAGs (to be defined in Section 2) and general outerpath DAGs. In a Dagstuhl Report [9], Bekos et al. claim that every n-vertex outerplanar DAG G has $\mathrm{sn}(G) \leq O(\log n)$, while they conjecture that actually $\mathrm{sn}(G) = O(1)$, even for all directed acyclic 2-trees G. Theorem 1.3 in the present paper confirms Conjecture 1.1, while Theorem 1.4 refutes the conjecture of Bekos et al.

Finally, let us mention that the decision problem of whether a given DAG admits a k-stack layout is known to be NP-complete for every $k \geq 2$ [10, 18, 37]. However, there are FPT algorithms parameterized by the branchwidth [18] or the vertex cover number [17].

2. Preliminaries

All graphs considered here are finite, non-empty, and simple, i.e., have neither loops nor parallel edges. For a graph G and an edge e, we define $G + e$, respectively $G - e$, as the graph obtained from G by adding, respectively removing, the edge e if possible. Similarly, vertices, sets of edges or vertices, and subgraphs can be added or removed with the same notation, where incident edges are also removed as necessary.

Outerplanar graphs and 2-trees. We start by considering undirected graphs. A graph G is *outerplanar* if it admits a plane drawing with all vertices incident to the outer face. Further, G is *maximal outerplanar* if no edge e can be added to G such that $G + e$ remains outerplanar. Equivalently, a maximal outerplanar graph is either a single vertex, a single edge, or consists of $n \geq 3$ vertices and admits a plane drawing whose outer face is bounded by a cycle of length n and where every inner face is bounded by a triangle.

Outerplanar graphs are intimately related to graphs of treewidth 2 and so-called 2-trees. A 2-*tree* is inductively defined by the following construction sequence:

— A single edge xy is a 2-tree[5]. This first edge in the process is called the *base edge*.
— If G is a 2-tree and vw an edge of G, then the graph obtained from G by adding a new vertex u and edges uv and uw is again a 2-tree. In this case we say that vertex u is *stacked* onto edge vw.

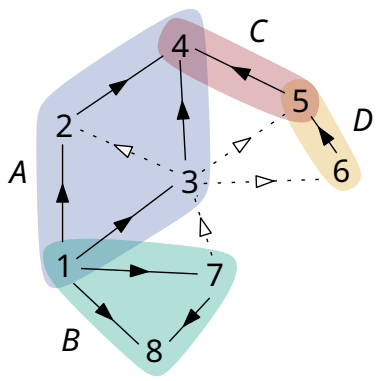
We remark that 2-trees are exactly the edge-maximal graphs of treewidth 2. Note that the construction sequence of a 2-tree G is not unique. In fact, for every 2-tree G and every edge xy of G there is a construction sequence of G with xy as the base edge. (And even for a fixed base edge, there can be exponentially many construction sequences of G.) However, as soon as the base edge xy is fixed, this uniquely determines for each vertex u different from x and y the edge vw that u is stacked onto. In this case we call vw the *parent edge* of u, vertices v and w the *parents* of u, and likewise u a *child* of v and w. Note that for each edge vw in G different from the base edge, its endpoints v and w are in a parent/child relationship.

Maximal outerplanar graphs are exactly those 2-trees in which at most one vertex is stacked onto each edge, except for the base edge onto which up to two vertices can be stacked. In fact, the base edge xy is an inner edge in every outerplanar drawing if two vertices are stacked onto xy, and otherwise xy is an outer edge. In the literature, maximal outerplanar graphs are also known as the *simple* 2-*trees* [45, 62].

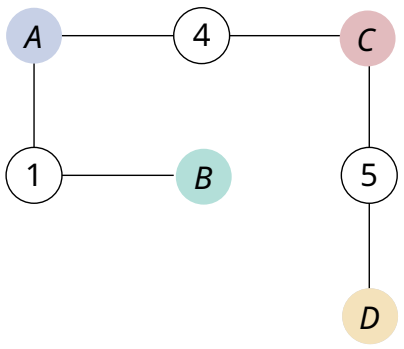
Let G be a connected outerplanar graph, see also Figure 3a for an example which is a subgraph of a simple 2-tree (for now, ignore the edge orientations). A *block* B of G is either a bridge or a maximal 2-connected component, while a vertex v of G is a *cut vertex* if $G - v$ is disconnected. The *block-cut tree* of G has as vertex set all blocks and all cut vertices of G and an edge between block B and cut vertex v if and only if $v \in B$. See Figure 3b for a block-cut tree of the graph in Figure 3a.

Outerplanar DAGs and directed acyclic 2-trees. For the remainder of this paper we will exclusively consider *directed graphs*, i.e., every edge e between two vertices v and w shall have a specified orientation, either from v to w, or from w to v. In the former case we denote $e = vw$ and in the latter case $e = wv$. In general, for every two disjoint vertex sets A and B we refer to the *edges between* A *and* B as those edges with exactly one endpoint in A and one endpoint in B, regardless of their orientation. On the other hand, the *edges from* A *to* B are those oriented from some vertex in A to some vertex in B. Notions like 2-trees and outerplanar graphs are inherited to directed graphs from their underlying undirected graphs. In particular, the *blocks* of a directed graph are exactly the blocks of the underlying undirected graph.

5 Usually this inductive definition of 2-trees starts with a triangle, but for the arguments below it is more convenient to let a single edge be a 2-tree as well.



(a) An outerplanar DAG G.

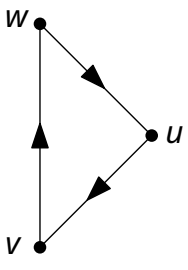


(b) The block-cut tree of G.

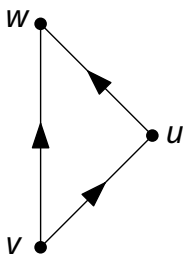
Figure 3. An outerplanar DAG G with its block-cut tree. The construction sequence of G is $1, 2, \ldots, 8$. In particular, the base edge is from 1 to 2. Dotted edges are non-edges of G whose addition to G gives an outerplanar 2-tree.

A *directed acyclic graph (DAG)* is a directed graph with no directed cycle, i.e., with no cycle $C = (v_1, \ldots, v_\ell)$, $\ell \geq 2$, with edges directed from v_i to v_{i+1} for $i = 1, \ldots, \ell - 1$ and from v_ℓ to v_1. Consider a directed 2-tree G with a fixed base edge xy (i.e., oriented from x to y) and a vertex $u \neq x, y$ with parent edge vw (i.e., oriented from v to w). There are four possibilities for the directions of the edges between u and its parents v and w (see Figure 4):

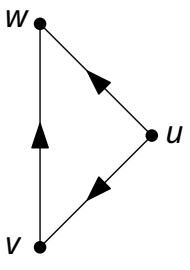
— If the edges wu and uv are in G, then (v, w, u) forms a directed cycle and we call u *cyclic*.
— If the edges vu and uw are in G, we call u *transitive*.
— In the two remaining cases we call u *monotone*. We say that u is a *left child* of vw if uv and uw are in G, and that u is a *right child* if vu and wu are in G. Let us note that a left (right) child is to the left (right) of its parents in every topological ordering.



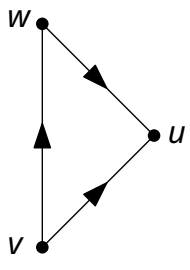
(a) Cyclic.



(b) Transitive.



(c) Monotone (left child).



(d) Monotone (right child).

Figure 4. The four possible stackings of a new vertex u onto a directed edge vw.

Observe that the notions of cyclic, transitive and monotone vertices crucially depend on the choice of the base edge. Further, observe that a directed 2-tree G is a DAG if and only if no vertex is cyclic, independent of the choice of the base edge. So if G is a DAG, then every construction sequence involves only transitive and monotone vertices.

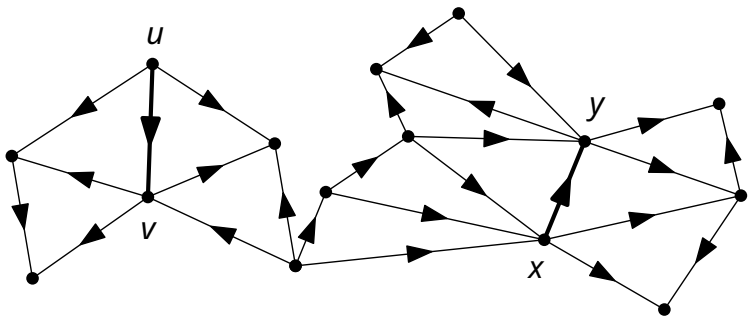


Figure 5. A block-monotone outerplanar DAG G. Each of the two blocks of G contains a base edge (xy and uv) such that all other vertices are monotone.

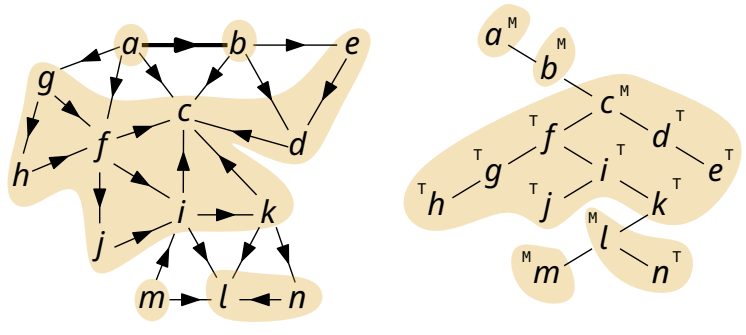


Figure 6. An outerplanar graph with base edge ab and its construction tree. The transitive subgraph below each monotone vertex is highlighted orange.

We say that a directed 2-tree G is *monotone* (respectively *transitive*) if there exists a choice for the base edge xy such that every vertex except for x and y is monotone (respectively transitive). In particular, a monotone (or transitive) directed outerplanar graph is always a maximal directed outerplanar graph. A connected outerplanar (but not necessarily maximal outerplanar) DAG is called *block-monotone* if every block is monotone, see e.g., Figure 5.

Given a maximal outerplanar DAG G with a fixed base edge xy, its *construction tree* with respect to xy is a rooted, undirected (and unordered) binary tree T on the vertices of G with vertex labels `M` (for "monotone") and `T` (for "transitive") such that[6]:

- The tail x of the base edge is the root and has label `M`.
- The head y of the base edge is the unique child of the root and also has label `M`.
- Whenever u is a child of an edge vw or wv of G with w being a child of v, then in T we have that u is a child of w. Moreover, vertex u is labeled `M` in T if u is a monotone vertex and `T` if u is a transitive vertex.

Observe that each vertex in the construction tree has at most two children. For each vertex v, the *transitive subgraph below* v is the subgraph of G induced by v and all its descendants w in T such that the unique v-w-path in T consists solely of vertices labeled `T` (except possibly v itself). See Figure 6 for a construction tree and the transitive subgraphs below monotone vertices.

Stack number versus twist number. Recall that if in a given vertex ordering $\prec$ of G we have a set of k pairwise crossing edges, we call this a k-twist in $\prec$. The maximum k such that there is a k-twist in $\prec$ is called the *twist number* of $\prec$. Then clearly with respect to this vertex

6 We remark that the construction tree (unrooted and without the labels) is exactly the tree of a nice tree-decomposition of width 2 of G, but we do not use this fact here.

ordering, $E(G)$ cannot be partitioned into fewer than k stacks. The minimum twist number over all (topological) vertex orderings $\prec$ of G is called the *twist number* $\mathrm{tn}(G)$ of G. Hence it follows that $\mathrm{sn}(G) \geq \mathrm{tn}(G)$ for every DAG G, i.e., having large twists in every topological vertex ordering is a simple reason for having a large stack number. Somewhat surprisingly, a large twist number is the only reason for a large stack number, up to a polynomial function.

THEOREM 2.1 (Davies, 2022 [20]). *For every vertex ordering $\prec$ of G with twist number k, we can partition $E(G)$ into $2k\log_2(k) + 2k\log_2(\log_2(k)) + 10k$ stacks.*

In fact, Davies [20] gives an upper bound on the chromatic number $\chi(H)$ of a circle graph H in terms of its clique number $\omega(H)$. (Gyárfás [33] was the first to show that circle graphs are χ-bounded, but Davies gives the first asymptotically tight bound.) To obtain Theorem 2.1, simply consider the circle graph H with $V(H) = E(G)$ whose edges correspond to crossing edges in $\prec$. Then $\omega(H)$ is the twist number of $\prec$ and a proper k-coloring of H is a partition of $E(G)$ into k stacks with respect to $\prec$.

3. Outerplanar DAGs have Constant Stack Number

In this section we prove Theorem 1.3, our first main result. One key ingredient is a recent result by Nöllenburg and Pupyrev [51] stating that the stack number of monotone outerplanar graphs is bounded. The idea behind our approach is to partition a given outerplanar DAG G into "transitive parts", such that the contraction of each part into a single vertex yields a block-monotone DAG H. Then the result from [51] can be applied to the blocks of H individually. Two things are left to do: First we show that the many stack layouts for the blocks of H can be combined into a single stack layout of H without requiring too many additional stacks. Then we show that each transitive part can be "decontracted" to yield a stack layout of G, again without requiring too many additional stacks.

We formalize all this by introducing a novel structural tool called *directed H-partitions*.

3.1 Monotone and Block-Monotone Outerplanar DAGs

Nöllenburg and Pupyrev [51] analyzed the stack number of different subclasses of outerplanar DAGs. One of their results is that monotone outerplanar DAGs with at most one vertex stacked on the base edge[7] have bounded twist number (and therefore also bounded stack number).

THEOREM 3.1 (Nöllenburg, Pupyrev [51]). *Every monotone outerplanar DAG G with at most one vertex stacked on the base edge has twist number $\mathrm{tn}(G) \leq 4$.*

COROLLARY 3.2. *Every monotone outerplanar DAG G has stack number $\mathrm{sn}(G) \leq 128$.*

7 The definition of *monotone* in [51] allows only one vertex stacked on the base edge. This is in contrast to our definition where the base edge is (the only edge) allowed to have two vertices stacked onto it.

PROOF. Let $G = (V, E)$ be a monotone outerplanar DAG with base edge e. Recall that the base edge of G may have two children so G is the union of at most two monotone outerplanar DAGs G_1 and G_2 such that both use e as their base edge, both have at most one vertex stacked on e, and their intersection is exactly e. By Theorem 3.1 for $i = 1, 2$ graph G_i admits a topological ordering $\prec_i$ with twist number at most 4. Thus by Theorem 2.1 we have

$$\mathrm{sn}(G_i) \leq 2 \cdot 4 \cdot \log_2(4) + 2 \cdot 4 \cdot \log_2(\log_2(4)) + 10 \cdot 4 = 64.$$

As the endpoints of e appear in the same order in $\prec_1$ and $\prec_2$, the corresponding 64-stack layouts of G_1 and G_2 can be combined into a 128-stack layout of G. ■

The next lemma and the following corollary extend the bound for monotone DAGs to those that are block-monotone. This will be important later, as block-monotonicity plays a crucial role in our proof that outerplanar DAGs have constant stack number.

LEMMA 3.3. *Let G be a DAG and $\mathcal{B}$ be the set of its blocks. Then we have*

$$\mathrm{sn}(G) \leq 2 + 2 \cdot \max_{B \in \mathcal{B}} \mathrm{sn}(B).$$

PROOF. We may assume that G is connected. Let T be the block-cut tree of G rooted at an arbitrary block of G and let $s = \max\{\mathrm{sn}(B) \mid B \in \mathcal{B}\}$ be the maximum stack number among all blocks of G. We incrementally construct a stack layout of G by processing the blocks of G according to their level in T one after another; the level of a block B being the number of cut vertices on the path from B to the root in T. In doing so, we maintain the following two invariants:

(I1) At most $2(s + 1)$ stacks are used in total.

(I2) At most $s + 1$ stacks are used for each block.

We start with the root block, which (like all other blocks) admits an s-stack layout by assumption. This fulfills the invariants (I1) and (I2) trivially.

Now consider a block B in level $\ell \geq 0$. We assume that B is already laid out and insert its children $B_1, \ldots, B_k$ into the layout. For this, repeat the following for every cut vertex v that B shares with blocks $B_1, \ldots, B_k$ in level $\ell + 1$. For $i = 1, \ldots, k$ take an s-stack layout of B_i (which exists by assumption) and let L_i and R_i denote the sets of vertices to the left of v and to the right of v in the s-stack layout for B_i, respectively. We insert the layouts of $B_1, \ldots, B_k$ directly to the left and directly to the right of v such that the vertices appear in the following order (see also Figure 7):

$$L_1 \prec \cdots \prec L_k \prec v \prec R_k \prec \cdots \prec R_1$$

Let E' denote the set of all edges that are in $B_1, \ldots, B_k$. An edge from E' and an edge e from a smaller level can cross only if e is incident to v and therefore belongs to B. As the edges of B use at most $s + 1$ stacks by invariant (I2), there are another $s + 1$ stacks available for the edges

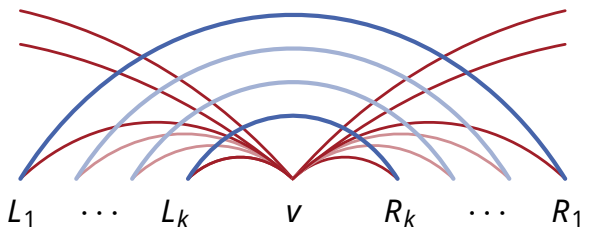


Figure 7. Integration of the stack layouts of blocks $B_1, \ldots, B_k$ around the cut vertex v in the current partial stack layout. Blocks $B_1, \ldots, B_k$ can use the same set of s stacks except for all edges incident to v which are on the $(s+1)$-th stack.

in E'. To assign the edges of E' to stacks we start with the s-stack layouts of each block and then move all edges incident to v to the $(s+1)$-th stack. Observe that edges in E' belonging to different blocks can only cross if exactly one of them is incident to v. As the edges incident to v form a star centered at v, we conclude that all stacks are crossing-free and $s+1$ stacks indeed suffice for E', maintaining (I2).

Finally, obverse that children of different cut vertices are separated in the layout and thus their edges do not cross. Therefore, we have a $(2s+2)$-stack layout for all blocks that are already completed, maintaining (I1). ■

Now Corollary 3.2 and Lemma 3.3 immediately imply the following.

COROLLARY 3.4. *Every block-monotone outerplanar DAG admits a* 258*-stack layout.*

3.2 Directed H-Partitions

We introduce *directed H-partitions* as a new structural tool and explore how they can be applied to reason about stack layouts of DAGs. They are not limited to DAGs nor to treewidth 2, and we believe they may be applicable also for other graph classes in the future. A related tool known as *layered H-partitions* was introduced in [25] to establish that undirected planar graphs have bounded queue number and has since been used widely. Thus, we state and prove the lemmas in this section in a more general form than we actually need them. The construction of a directed H-partition for outerplanar DAGs is deferred to Section 3.3. The definition of directed H-partitions is illustrated in Figure 8.

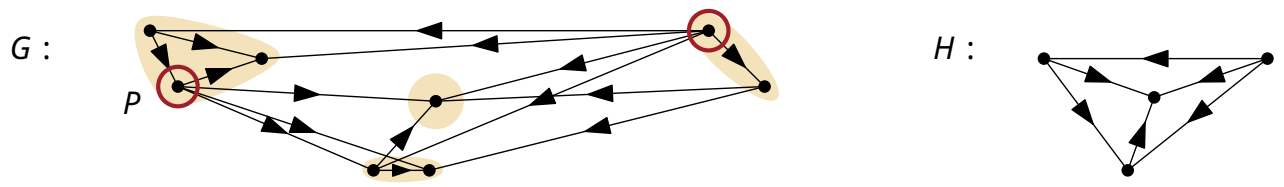


Figure 8. A directed H-partition (orange) of a directed graph G (left) and the quotient H (right). The cut cover number of the part P is 2 as the two vertices marked in red cover all edges with exactly one endpoint in P.

DEFINITION 3.5 (Directed H-Partition, Cut Cover Number). Let G and H be directed graphs. A *directed H-partition* of G is a partition $\mathcal{P}$ of $V(G)$ such that the following holds:

- For every two parts $P, Q \in \mathcal{P}$ the edges of G between P and Q are oriented either all from P to Q or all from Q to P.
- The quotient $G/\mathcal{P}$ is isomorphic to H. Here $G/\mathcal{P}$ is obtained from G by contracting each part $P \in \mathcal{P}$ into a single vertex v_P and directing an edge from v_P to v_Q in H whenever in G there is some edge from P to Q. (This orientation is well-defined by the first property.)

For a part $P \in \mathcal{P}$ its *cut cover number* is the smallest number of vertices of G required to cover (i.e., be incident to) all edges of G with exactly one endpoint in P. The cut cover number of $\mathcal{P}$ is the maximum cut cover number among all its parts.

So each vertex v_P in H corresponds to a part in $P \in \mathcal{P}$ and a subgraph of G, denoted by $G[v_P] = G[P]$, that is induced by the vertices in P. More generally, every induced subgraph B of H corresponds to a subset $\mathcal{P}_B \subseteq \mathcal{P}$ of parts, and we let $G[B]$ denote the corresponding subgraph of G that is induced by all vertices of G contained in parts in $\mathcal{P}_B$.

The definition of directed H-partitions is very similar to the well-known concept of (undirected) H-partitions. The main difference and difficulty is that we need to ensure that the quotient is well-defined, i.e., the orientation of the edges between two parts is consistent. Nevertheless, many useful properties of the undirected version are inherited. In particular, if every part is connected, then the quotient is a minor of the underlying graph, which implies that treewidth and planarity are preserved. Although not used here, we remark that the successful idea of Dujmović et al. [25] to combine H-partitions with layerings to so-called layered H-partitions is also feasible in the directed setting.

Given directed graphs[8] G and H and a directed H-partition $\mathcal{P}$ of G, we say that a vertex ordering $\prec_G$ of G *expands* a vertex ordering $\prec_H$ of H if all vertices of G belonging to the same part of $\mathcal{P}$ appear consecutively in $\prec_G$ and whenever $P \in \mathcal{P}$ lies to the left of $Q \in \mathcal{P}$ in $\prec_G$, then $v_P \in V(H)$ lies to the left of $v_Q \in V(H)$ in $\prec_H$.

LEMMA 3.6. *Let G and H be DAGs and $\mathcal{P}$ be a directed H-partition of G with cut cover number at most w. Further, let $\mathrm{sn}(G[P]) \leq s$ for each $P \in \mathcal{P}$. Then for every h-stack layout $\prec_H$ of H, there is a $(3wh + s)$-stack layout $\prec_G$ of G expanding $\prec_H$.*

In particular, we have $\mathrm{sn}(G) \leq 3w \cdot \mathrm{sn}(H) + s$.

PROOF. We expand a given h-stack layout $\prec_H$ of H to a $(3wh + s)$-stack layout $\prec_G$ of G. For each part $P \in \mathcal{P}$ consider an s-stack layout $\prec_P$ of $G[P]$, which exists by assumption, and replace in $\prec_H$ vertex v_P corresponding to P by the vertices in P ordered as in $\prec_P$. As in the resulting vertex ordering $\prec_G$ of G all vertices from the same part appear consecutively (in other words, $\prec_G$ expands $\prec_H$), it follows that no two edges e_1, e_2 in G belonging to different parts in $\mathcal{P}$ cross. Therefore, we may assign all edges with both endpoints in the same part to the same set of s stacks.

It remains to assign the edges of G with endpoints in two different parts to the remaining $3wh$ stacks. For this we consider each stack S in the h-stack layout of H separately and show that all edges of G corresponding to edges in S can be assigned to $3w$ stacks. (An edge $vw \in E(G)$ with $v \in P$ and $w \in Q$ corresponds to the edge $v_P v_Q \in E(H)$.) First, we note that the edges in S form an outerplanar subgraph of H. As such, it can be partitioned into three star forests [34]. Again, we can treat each star forest separately, and we are left with assigning the edges in G corresponding to the same star forest F in H to at most w stacks.

For this, consider two edges e_1, e_2 of G corresponding to the same star forest F that cross in our chosen vertex ordering $\prec_G$ of G. If their endpoints are in four different parts in $\mathcal{P}$, then their corresponding edges in H cross, which is impossible. Therefore, the endpoints of e_1 and e_2 lie in at most three different parts. Hence, there is one part containing at least two of the endpoints, and we conclude that e_1 and e_2 actually correspond to the same star in F. Thus it suffices to consider the stars of F separately and reuse the same set of w stacks for all stars from F. Now consider all edges of G corresponding to the same star X of F. As the cut cover number of $\mathcal{P}$ is at most w, there is a set $V' \subseteq V(G)$ with $|V'| \leq w$ such that every edge of G corresponding to X is incident to (at least) one vertex in V'. For each vertex $v' \in V'$ its incident edges form a star in G and are therefore non-crossing in $\prec_G$. Thus, we can assign all incident edges at v' corresponding to X to the same stack.

8 Neither Definition 3.5 nor the definition of expanding vertex orderings requires G or H to be acyclic. However, in this paper we shall consider solely constellations where G and H are both DAGs.

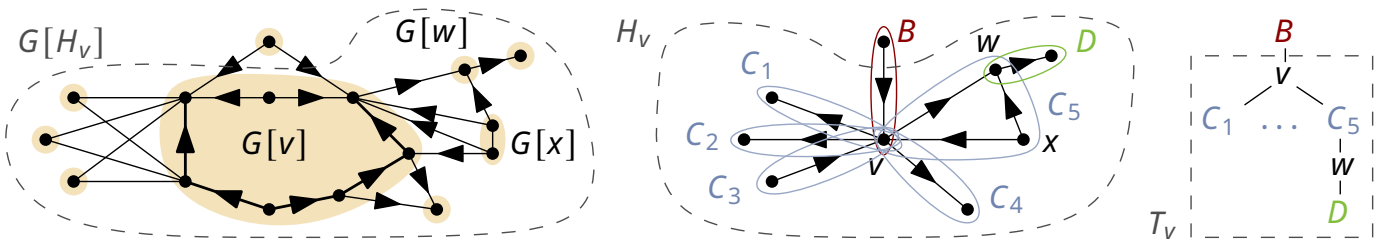


Figure 9. Left: A DAG G with a directed H-partition as in Lemma 3.7 with $p = 2$ and $t = 3$. Some edge directions are omitted for better readability. Middle: H and its blocks. Right: The block-cut tree of H.

This requires at most w stacks for each star X in F, hence also at most w stacks for each star forest F of S. To sum up, we have at most $3w$ stacks for each stack S of the h-stack layout of H. Including the s stacks from the beginning, this yields at most $3wh + s$ stacks in total. ■

Lemma 3.6 gives a good stack layout of G, provided G admits a directed H-partition with small cut cover number for some H with small stack number. The notion of the cut cover number enables us to give the bound on the stack number independently of the size of the parts. We remark that without a bound on the cut cover number, there may be a twist between the vertices of two parts that is as large as the smaller of the two parts. For the next lemma, we loosen the prerequisites by considering the blocks of H separately. First, we require for each block B of H that the corresponding subgraph $G[B]$ has a good stack layout (for example, due to a small cut cover number of the inherited directed H-partition of $G[B]$). And second, the interactions between the blocks of H sharing a common cut vertex v are restricted.

LEMMA 3.7. *Let G be a DAG with a directed H-partition $\mathcal{P}$ such that*

- *for every block B of H the subgraph $G[B]$ of G admits an s-stack layout expanding some vertex ordering of H.*

Moreover, let T be the block-cut tree of H rooted at some block of H, such that for every cut vertex v of H with child blocks $C_1, \dots, C_k$ in T the following holds:

- *For $i = 1, \dots, k$, the intersection of $G[v]$ with the neighborhood of $G[C_i - v]$ consists of a single edge $e_i \in E(G[v])$.*
- *Edges $e_1, \dots, e_k$ can be covered with at most p directed paths in $G[v]$.*
- *For each edge $e \in E(G[v])$ we have $e = e_i$ for at most t indices $i \in \{1, \dots, k\}$.*

Then $\mathrm{sn}(G) \leq 4spt$.

We refer to Figure 9 for an illustration of the situation in Lemma 3.7. In the following proof, we denote the neighborhood of a subgraph G' of some graph G by $N(G')$.

PROOF. Let v be a vertex in H. Recall that $v = v_P$ represents a part $P \in \mathcal{P}$ in the directed H-partition and $G[v] \subseteq G$ is an induced subgraph of G. If v is a cut vertex of H, we associate to v also another subgraph of G by considering everything below v in the block-cut tree T. Formally,

let T_v denote the subtree of T with root v and let H_v denote the subgraph of H that is the union of all blocks in T_v. Then the corresponding subgraph $G[H_v]$ is the subgraph of G induced by the union of all parts $P \in \mathcal{P}$ for which the vertex v_P in H appears in some block B in T_v. See again Figure 9 for an illustration of the notation.

We shall find a $4spt$-stack layout of G whose vertex ordering $\prec_G$ has the following properties:

(I1) For every cut vertex v of H the vertices in $G[H_v]$ appear consecutively in $\prec_G$.
(I2) For every non-cut vertex v of H the vertices in $G[v]$ appear consecutively in $\prec_G$.

Assuming $\prec_G$ satisfies (I1), the following holds:

CLAIM 3.8. *For every two vertex-disjoint blocks B_1, B_2 of H, no edge in $G[B_1]$ crosses an edge in $G[B_2]$ with respect to $\prec_G$.*

Proof. Let T_1 and T_2 be the subtrees of T rooted at B_1 and B_2, respectively. First assume that T_1 and T_2 are disjoint, which in particular means that neither B_1 nor B_2 is the root of T. With v_1, v_2 being the parents of B_1, B_2 in T, respectively, we have $v_1 \neq v_2$ since B_1 and B_2 are vertex-disjoint. Then (I1) gives that vertices of $G[B_1] \subseteq G[H_{v_1}]$ and $G[B_2] \subseteq G[H_{v_2}]$ appear in $\prec_G$ in disjoint intervals. Thus no edge in $G[B_1]$ crosses an edge in $G[B_2]$.

If T_1 and T_2 are not vertex-disjoint, assume without loss of generality that $T_1 \subset T_2$; in particular that B_1 is not the root of T. Then by (I1) the vertices of $G[H_{v_1}]$ form in $\prec_G$ a contiguous interval I. In particular, every edge in $G[B_1] \subseteq G[H_{v_1}]$ has both endpoints in I. As B_1 and B_2 are vertex-disjoint, we have $G[B_2] \cap G[H_{v_1}] = \emptyset$ and every edge in $G[B_2]$ has neither endpoint in I. Thus no edge in $G[B_1]$ crosses an edge in $G[B_2]$. ◆

Claim 3.8 allows us to reuse the same set of stacks for vertex-disjoint blocks. With this in mind, we partition the blocks of H into two sets $\mathcal{B}_{\text{odd}}$ and $\mathcal{B}_{\text{even}}$ containing the blocks with an odd, respectively even number of cut vertices on their path to the root in T. Then it is enough to use a set of $2spt$ stacks for blocks in $\mathcal{B}_{\text{odd}}$ and a set of $2spt$ different stacks for $\mathcal{B}_{\text{even}}$, giving the desired $4spt$ stacks in total. Observe that within the same set of blocks, say $\mathcal{B}_{\text{odd}}$, two blocks are either again vertex-disjoint (and thus non-crossing by Claim 3.8) or have a common parent in T. Thus it is left to consider a single cut vertex and its child blocks.

We now construct the desired $4spt$-stack layout $\prec_G$ of G by processing the block-cut tree T from the root to the leaves. After initializing the root block, in each step we consider a cut vertex v whose parent block B is already processed and process all child blocks of v simultaneously. In doing so, we maintain after each step (I1) and (I2) for the already processed subgraph of G. To initialize, the root of T is a single block of H and admits an s-stack layout expanding some vertex ordering $\prec_H$ of H by assumption. This fulfills (I1) trivially and (I2) since the layout expands $\prec_H$.

Now for a step, consider a cut vertex v whose parent block is already processed and let $C_1, \dots, C_k$ be the child blocks of v in T. By the assumptions of the lemma, for each $i = 1, \dots, k$

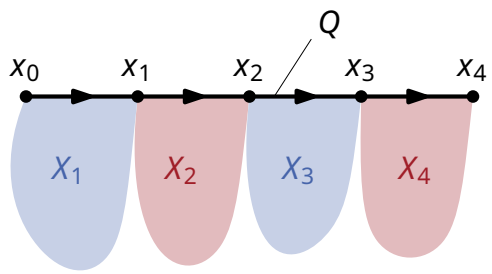


Figure 10. The situation of Lemma 3.7 (see Figure 9, in particular the three blocks C_1, C_2, C_3 attached to the same edge in $G[v]$) simplified in two ways: We only consider one directed path $Q \subseteq G[v]$ and we have exactly one block X_j at each edge $x_{j-1}x_j$, instead of at most t (which is paid for by a factor of t) or none (for which the respective X_j can simply be ignored). That is, each block X_j represents at most one child block C_i of v, where $e_i = x_{j-1}x_j$. Note that the colors match those used in the layout in Figure 11.

the intersection $G[v] \cap N(G[C_i - v])$ consists of a single edge $e_i \in E(G[v])$, the edges $e_1, \ldots, e_k$ are covered with at most p directed paths $Q_1, \ldots, Q_p$ in $G[v]$, and for each edge $e \in E(G[v])$ we have $e = e_i$ for at most t indices $i \in \{1, \ldots, k\}$. Since all of $C_1, \ldots, C_k$ are in $\mathcal{B}_{\text{even}}$ or all in $\mathcal{B}_{\text{odd}}$, we have a set of $2spt$ stacks at our disposal. Reserve p pairwise disjoint sets of stacks of size $2st$, one per path. For a fixed path Q, group these $2st$ stacks further into t disjoint subsets of size $2s$, such that for each subset each edge e of Q corresponds to at most one index $i \in \{1, \ldots, k\}$ with $e = e_i$. It is left to show that we can find a $2s$-stack layout for a fixed path $Q = (x_0, \ldots, x_\ell)$ of length ℓ and a set of blocks $X_1, \ldots, X_\ell$, where $G[v] \cap N(G[X_j - v])$ is exactly the edge $x_{j-1}x_j$ with $j = 1, \ldots, \ell$ (having one block per edge is the most difficult case, having less only makes it easier), refer to Figure 10. Note that each child C_i is dealt with: It is either in $\mathcal{B}_{\text{even}}$ or $\mathcal{B}_{\text{odd}}$, it is attached to one of the p paths, and it is contained in one of the t subsets for the path within which C_i does not share the edge e_i with some other child. That is, each of the X_j's takes care of at most one C_i.

We are now in the situation illustrated in Figure 10 and by the assumptions of the lemma, for each block X_j the corresponding $G[X_j]$ admits an s-stack layout $\prec_j$ expanding some vertex ordering of H. For every vertex w in X_j the vertices of $G[w]$ appear consecutively in $\prec_j$, and this holds in particular for $w = v$. We remove from $\prec_j$ all vertices in $G[v]$ except for x_{j-1} and x_j. As $G[v]$ and $G[X_j - v]$ are connected only via x_{j-1} and x_j, this does not remove any edge of G corresponding to an edge in $X_j - v$.

Let L_j and R_j denote the sets of vertices that are to the left of x_{j-1} respectively to the right of x_j in the remaining $\prec_j$. Recall that there are no vertices between x_{j-1} and x_j as the vertices of $G[v]$ appear consecutively in $\prec_j$. We now insert L_j immediately before x_{j-1} and R_j immediately after x_j in the vertex ordering of the already processed graph. See also Figure 11 for a visualization. We observe that the edges of X_j and $X_{j'}$ do not cross for $|j - j'| > 1$. Thus, $2s$ stacks indeed suffice for all $X_1, \ldots, X_\ell$ by reusing the same s stacks for even indices j and another s stacks for odd indices j.

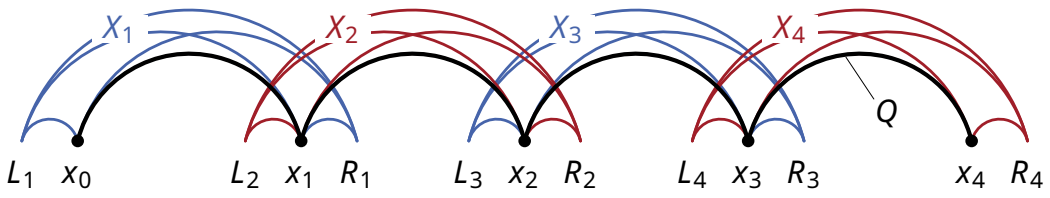


Figure 11. Incorporating the s-stack layouts of $G[X_1], \dots, G[X_4]$ (compare Figure 10) into the interval containing $G[v]$, where $G[v] \cap N(G[X_i - v]) = \{x_{i-1}, x_i\}$ and $Q = (x_0, \dots, x_4)$ is a directed path in $G[v]$. Both, the red and the blue edges represent s-stack layouts of respective $G[X_i]$.

To finish the proof, recall that we maintain (I1) and (I2) after each step for the already processed subgraph of G. For this, note that by processing its child blocks, v is turned from a non-cut vertex to a cut vertex, and thus we may assume (I2) but need to satisfy (I1) for v. As, by (I2), the vertices of $G[v]$ were consecutive in the vertex ordering before, it follows that the vertices of $G[H_v]$ are consecutive in the vertex ordering after those insertions, i.e., (I1) is fulfilled. Moreover, since each vertex ordering $\prec_j$ expands some vertex ordering of H, we conclude that for each newly processed vertex w of H, its corresponding subgraph $G[w]$ lies consecutively inside L_j or inside R_j for some j. Thus also (I2) is fulfilled. ■

REMARK 3.9. The assumption in Lemma 3.7 that the edges $e_1, \dots, e_k$ can be covered with at most p directed paths in $G[v]$ can be easily relaxed. Indeed, if B is the parent block of v in the block-cut tree of H, it is enough that the s-stack layout of $G[v]$ as part of the s-stack layout of $G[B]$ in the statement of the lemma contains a set of at most m non-crossing matchings that cover $e_1, \dots, e_k$. Then the above proof can be easily adapted to show $\mathrm{sn}(G) \leq 2smt$. Having at most p directed paths is clearly enough to have $m \leq 2p$ non-crossing matchings in *every* topological vertex ordering of $G[v]$.

3.3 Directed H-Partitions of Acyclic Outerplanar Graphs

The goal of this section is to construct directed H-partitions $\mathcal{P}$ for every outerplanar DAG G, such that we can apply Lemma 3.6 from the previous section to reason that $\mathrm{sn}(G)$ is bounded by a constant. In particular, we aim for a block-monotone H, with each part $P \in \mathcal{P}$ inducing a relatively simple subgraph in G, as well as small cut cover numbers. Instead of bounding the cut cover number globally, it suffices to have it constant for each block of H locally. Formally, if B is a block of H, then $\mathcal{P}_B = \{P \in \mathcal{P} \mid v_P \in B\}$ is a directed B-partition of the corresponding subgraph $G[B]$ of G, and we want that the cut cover number of $\mathcal{P}_B$ is constant. This is enough to apply Lemma 3.7 from the previous section. These properties of the following lemma are illustrated in Figure 12 and then combined in Figure 13.

LEMMA 3.10. *Let G be a maximal outerplanar DAG with fixed base edge and T be its (rooted) construction tree. Then G admits a directed H-partition $\mathcal{P}$ with the following properties:*

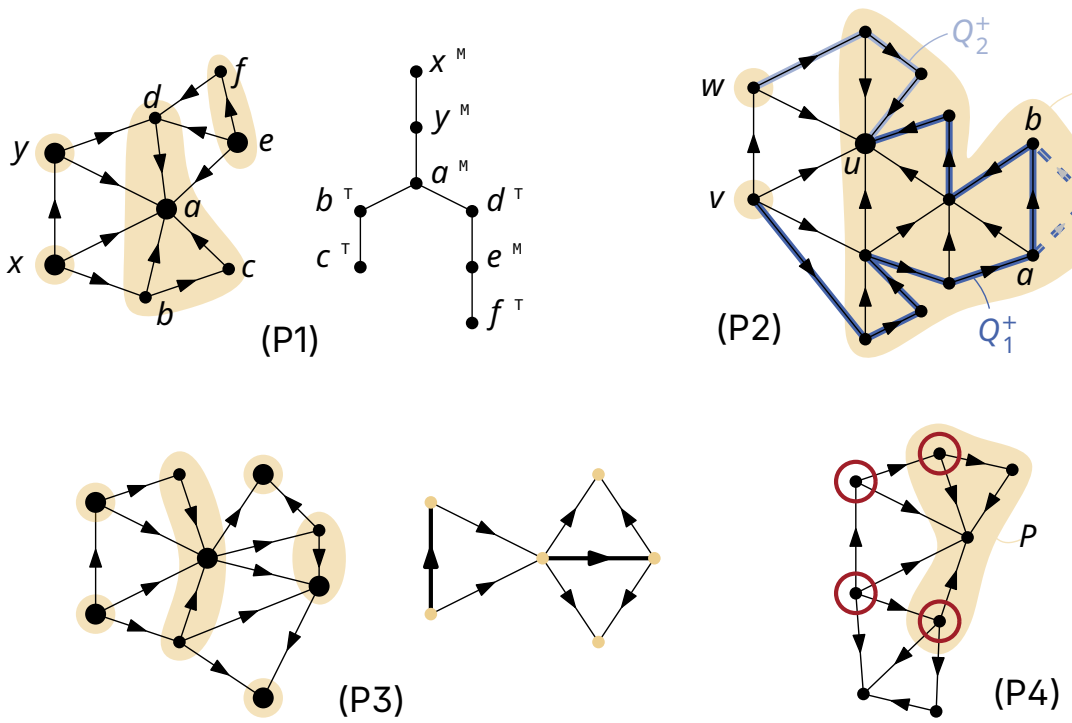


Figure 12. The properties guaranteed by Lemma 3.10. Figure 13 shows a larger example combing all properties. (P1) Each monotone vertex (thick) has its own part (orange) including all transitive vertices until the next monotone vertex, i.e., the transitive subgraph below it (right). (P2) Two paths Q_1^+, Q_2^+ cover all outer edges of part P. Adding another transitive vertex z extends the path (dashed). Removing v and w yields Q_1 and Q_2. (P3) Each block of the quotient (right) consists of a base edge (thick) and monotone vertices corresponding to the monotone vertices in G (left). (P4) Inside each block, the cut cover number is bounded, i.e., for each part P there are four vertices (circled red) that cover all edges leaving P.

(P1) *$\mathcal{P}$ contains exactly one part P for each monotone vertex u of G and P contains exactly the vertices of the transitive subgraph[9] below u.*

(P2) *For each $P \in \mathcal{P}$ the graph $G[P]$ contains two directed paths Q_1, Q_2 such that every vertex of $G - P$ that is stacked onto an edge in $G[P]$ is stacked onto an edge of Q_1 or Q_2.*

(P3) *H is a block-monotone outerplanar DAG.*

(P4) *For each block B of H the directed B-partition $\mathcal{P}_B$ of $G[B]$ has cut cover number at most 4.*

PROOF. We divide the proof into five parts. First we define $\mathcal{P}$ according to property (P1). Second, we analyze the subgraph of G induced by the vertices of each part $P \in \mathcal{P}$ and thereby verify property (P2). Third, we show that $\mathcal{P}$ is indeed a directed H-partition, i.e., the quotient $H := G/\mathcal{P}$ is a well-defined directed graph. Then we show (P3), i.e., that H is block-monotone. At last, we prove that the cut cover number for each block of H is at most 4, verifying property (P4).

9 See Section 2 to recall the definition.

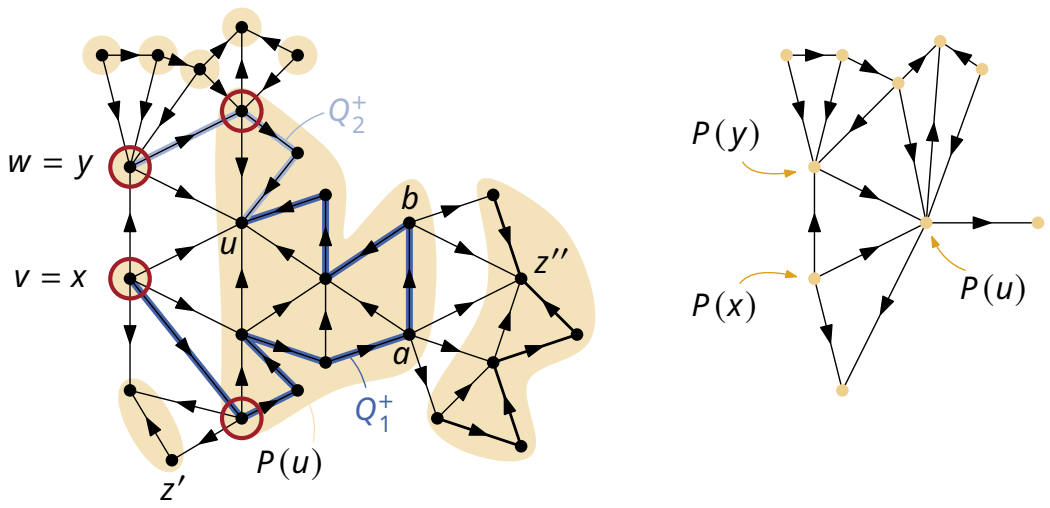


Figure 13. Full example for the proof of Lemma 3.10. For better readability, each step is shown separately in Figure 12. Left: An outerplanar DAG with base edge xy and a directed H-partition $\mathcal{P}$ (orange). The vertex u is stacked onto the edge vw and is the unique monotone vertex in $P(u)$. The paths Q_1^+ (darkblue from v to u) and Q_2^+ (lightblue from w to u) for $P(u)$ are drawn thick. The vertices marked with red circles certify that the cut cover number within the large block (all parts except for the rightmost) of the part $P(u)$ is at most 4. In a later step, z'' is stacked onto ab, introducing a bridge that is a new block in H. Right: The quotient $H = G/\mathcal{P}$, where each part of $\mathcal{P}$ is contracted to a single vertex. The vertices resulting from the parts $P(x), P(y)$, and $P(u)$ are labeled.

Construction of $\mathcal{P}$. For each monotone vertex u of G, let $P(u)$ be the set of vertices in the transitive subgraph below u. By definition, we have $u \in P(u)$ and that u is the only monotone vertex in $P(u)$. Moreover recall that the root of the construction tree T is a monotone vertex and thus every transitive vertex of G lies in $P(u)$ for some monotone u. Hence $\mathcal{P} = \{P(u) \mid u \text{ monotone vertex of } G\}$ is indeed a partition of $V(G)$ satisfying (P1). ◆

For the remainder of the proof it will be convenient to consider a construction sequence of G in which every monotone vertex u is immediately followed by the vertices in the transitive subgraph below u. We consider such a sequence vertex by vertex and argue about intermediate versions of G and $\mathcal{P}$ (and thus of $H = G/\mathcal{P}$). At the beginning we have only the base edge xy directed from x to y, which are both labeled `M`. Then $\mathcal{P} = \{P(x), P(y)\}$ with $P(x) = \{x\}$ and $P(y) = \{y\}$. In each subsequent step, a vertex u is stacked onto an edge vw, where v, w are the parents of u. Recall, however, that the *parent of u in the construction tree T* is the younger among v, w.[10] If u is a transitive vertex (i.e., labeled `T` in T), then u is simply added to the part P in $\mathcal{P}$ that contains the parent of u in T. Otherwise, if u is a monotone vertex (i.e., labeled `M` in T), then a new part $P(u)$ is added to $\mathcal{P}$ consisting of only u. Figures 12 and 13 show an example of the resulting partition. Note that this iterative process indeed eventually results in the partition $\mathcal{P} = \{P(u) \mid u \text{ monotone vertex of } G\}$ of G as described above.

10 We remark that, unless stated otherwise, parent/child refers to the construction sequence.

$\mathcal{P}$ fulfills (P2). We shall argue that each part $P \in \mathcal{P}$ fulfills (P2) by showing this for the moment when P is created in the construction sequence and maintaining (P2) for P whenever P is augmented with a new vertex thereafter. So fix a monotone vertex u and consider the step in the construction sequence when the part $P = P(u)$ is created. As P is created with just the single vertex u, property (P2) holds vacuously immediately after its creation. Moreover, if $u = x$, i.e., the tail of the base edge xy, then $P = P(u)$ will never be augmented with a new vertex and (P2) holds throughout. To show that (P2) is maintained for $P = P(u)$ for $u \neq x$ with each step of the construction sequence, we maintain two directed paths Q_1^+, Q_2^+ associated to P and containing all edges of the subgraph induced by P onto which a vertex can be stacked (see again Figures 12 and 13). If $u = y$, we initialize both Q_1^+ and Q_2^+ with (x, y). If $u \neq x, y$ and the monotone vertex u is stacked onto edge vw, we initialize $Q_1^+ := (v, u)$ and $Q_2^+ := (w, u)$. In any case, we have a u-v-path Q_1^+ and a u-w-path Q_2^+ (taking $v = w = x$ if $u = y$), where in G both paths are either consistently oriented towards u or both consistently oriented away from u. It holds that the next vertex in the construction sequence with a parent in P is stacked onto an edge of Q_1^+ or an edge of Q_2^+. Moreover, the subpaths $Q_1 := Q_1^+ - \{v\}$ and $Q_2 := Q_2^+ - \{w\}$ are contained in $G[P]$ and trivially fulfill (P2) for $P = P(u)$.

Now consider the next step with a transitive vertex z added to $P = P(u)$. (If the next vertex is monotone, then P is final and we are done.) Then z is stacked onto an edge ab of Q_1^+ or Q_2^+ (gray/dashed in Figure 12). We replace edge ab in Q_1^+ (or Q_2^+) by the path (a, z, b). This way, Q_1^+ and Q_2^+ are still oriented in G consistently towards u or away from u. Since G is outerplanar, no future vertex is also stacked onto ab. Hence, the next vertex with a parent in P is again stacked onto an edge of Q_1^+ or an edge of Q_2^+. It follows that the subpaths $Q_1 := Q_1^+ - \{v\}$ and $Q_2 := Q_2^+ - \{w\}$ are contained in $G[P]$ and again fulfill (P2) for $P = P(u)$. ◆

$\mathcal{P}$ is a directed H-partition. We show that in each step of the construction sequence $\mathcal{P}$ is indeed a directed H-partition, i.e., all edges between any two parts $P(u_1), P(u_2) \in \mathcal{P}$ are oriented in the same direction. Assume without loss of generality that monotone vertex u_1 appears before monotone vertex u_2 in the construction sequence. Thus at the time u_2 is stacked onto some edge, $P(u_1)$ is already in $\mathcal{P}$ and contains at least one parent of u_2. If u_2 is a right (left) child, we show that all edges are oriented from $P(u_1)$ to $P(u_2)$, respectively from $P(u_2)$ to $P(u_1)$. This clearly holds immediately after the construction step for u_2.

By symmetry, assume that u_2 is a right child. Then the paths Q_1^+ and Q_2^+ for $P(u_2)$ are both in G directed towards u_2 and away from the parents of u_2. Now consider the next step with a transitive vertex z added to $P(u_2)$ such that z also has one of its parents in $P(u_1)$. Then z is stacked onto the first edge of Q_1^+ or the first edge of Q_2^+. As z is transitive, the edge between $P(u_1)$ and z is oriented towards $z \in P(u_2)$, as desired. ◆

H is a block-monotone DAG. To show that (P3) holds, we start by observing that $H = G/\mathcal{P}$ is a DAG. By the previous paragraph, H is a well-defined directed graph, but we need to argue that

it is acyclic[11]. Since all edges between two parts of $\mathcal{P}$ are oriented consistently, it is sufficient to verify that H remains acyclic whenever a new part P is created. So we have $P = P(u)$ for some monotone vertex u and either both edges incident to u are oriented towards u or both edges are oriented away from u. Thus, v_P is not on any cycle in H.

Further observe that each part in $\mathcal{P}$ induces a connected subgraph of G. Thus, the quotient graph $H = G/\mathcal{P}$ is a minor of G. In particular, H is outerplanar and every block of H is maximal outerplanar. We refer to Figures 12 and 13 for examples of quotients.

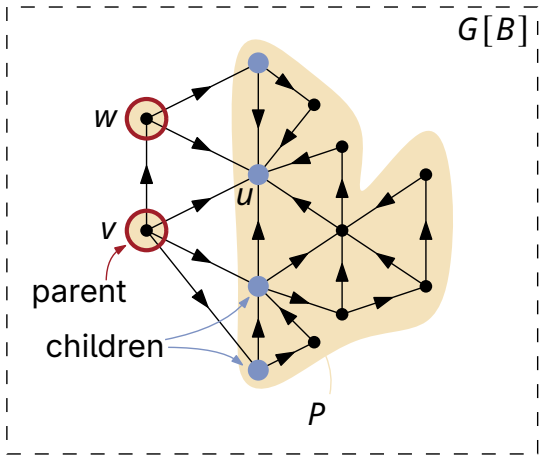
It remains to show for (P3) that H is block-monotone. Again we do this using the construction sequence of G. Initially, there is only the base edge xy, and we have $\mathcal{P} = \{P(x), P(y)\}$ with $P(x) = \{x\}$ and $P(y) = \{y\}$. We declare the currently only edge of H to be the base edge for the currently only block of H.

Now assume again that vertex u is stacked onto an edge vw in G. If u is a transitive vertex, nothing needs to be done as H does not change. If u is a monotone vertex and v, w are in different parts in $\mathcal{P}$, then some block B of H is extended by a new vertex v_P for $P = P(u)$. Since the stacking is monotone, the enlarged block B of H remains monotone with respect to the same base edge. Lastly, if u is a monotone vertex and v, w are in the same part $P' \in \mathcal{P}$, then H gets extended by a bridge e between $v_{P'}$ and the new vertex v_P for $P = P(u)$. This bridge e forms a new block of H, which is monotone with base edge e. ◆

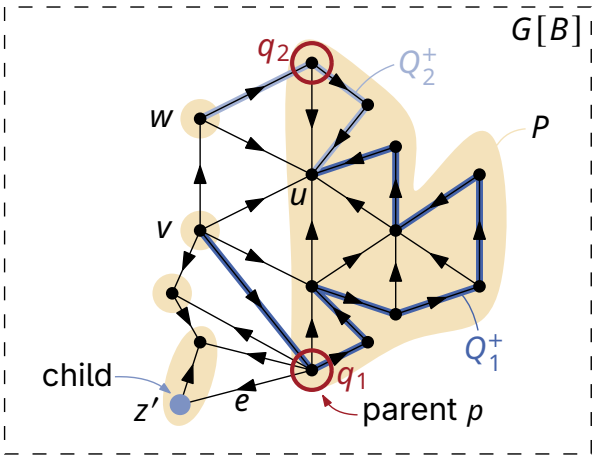
Each block has cut cover number at most 4. Let B be a block of H and $G[B] \subseteq G$ be the corresponding subgraph of G. Further, let P be a part of $\mathcal{P}_B$, i.e., such that v_P lies in B. The goal is to show that part P has cut cover number at most 4 within the block B, i.e., to find a set S of at most four vertices in $G[B]$ that cover all edges of $G[B]$ with one endpoint in P and the other endpoint in B but in another part of $\mathcal{P}_B$. Recall that the endpoints of every edge in G are in a parent/child relation by construction of the outerplanar graph, where each child has two parents independent of the edge directions. In particular, we shall consider edges $e \in E(G[B])$ whose parent-endpoint lies in P while the child-endpoint does not, and edges whose child-endpoint is in P while the parent-endpoint is not.

Let u be the monotone vertex of G with $P = P(u)$. Clearly, if $u = x$, then $P(u) = \{u\}$ and it is enough to take $S = \{u\}$. If $u \neq x$, let Q_1^+ and Q_2^+ be the paths associated to $P(u)$ as constructed above. By symmetry assume that Q_1^+ is a directed v-to-u-path and Q_2^+ is a directed w-to-u-path in G (either because $u = y$ or $u \neq y$ is some right child). Let us first assume that $G[B]$ contains v and w. In this case let S be the set consisting of v, w, the neighbor q_1 of v in Q_1^+, and the neighbor q_2 of w in Q_2^+, see Figures 13 and 14. (Several of these vertices might coincide.) Then $|S| \leq 4$ and we claim that S covers every edge in $G[B]$ with exactly one endpoint in P. Indeed, every edge in $G[B]$ with child-endpoint in P but parent-endpoint outside P has as parent-endpoint

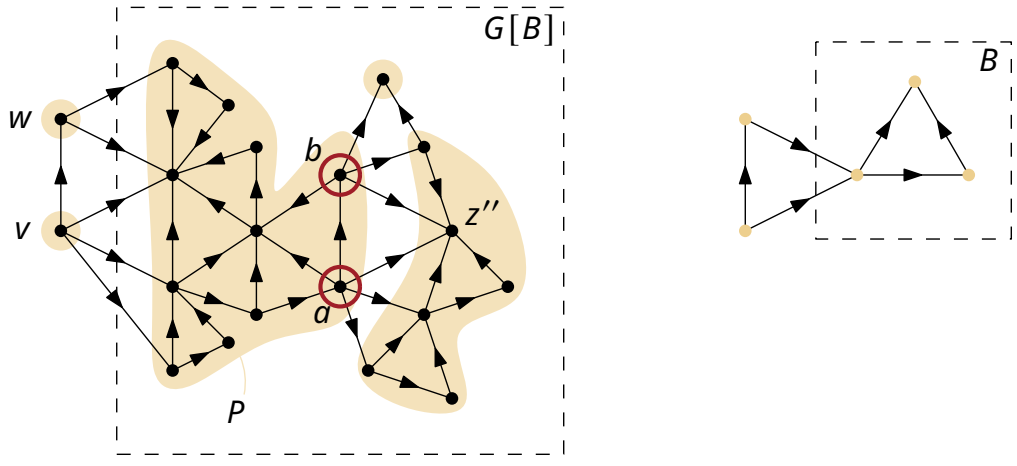
11 The quotient of an outerplanar DAG obtained by contracting some edges might be cyclic. For example consider a 6-cycle with alternating edge orientations. Contracting a maximum matching results in a directed 3-cycle.



(a) All edges with the child-endpoint in P but the parent-endpoint not are incident to v or w.



(b) All edges with the parent-endpoint in P but the child-endpoint not are incident to q_1 or q_2. Note that stacking onto any edge with both endpoints in P yields a new block and thus is not considered here, while stacking onto an edge with both endpoints not in P creates only edges that do not intersect P.



(c) Left: If $G[B]$ does not contain v, w, we choose $S = \{a, b\}$. Again, recall that stacking onto any edge in part P opens a new block. Right: The quotient with block B

Figure 14. Bounding the cut cover number in the proof of Lemma 3.10. Each subfigure refines Figure 13 to show the details in the different cases of the proof. Recall that the cut cover number is bounded for each block separately.

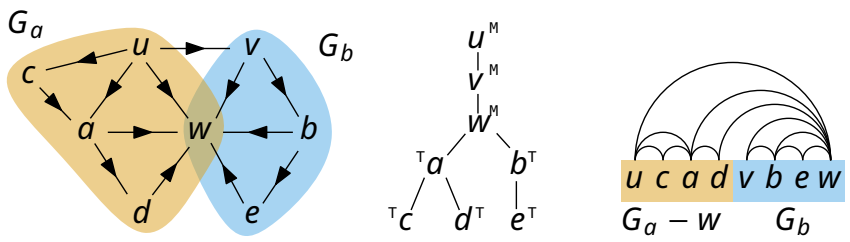


Figure 15. An outerplanar DAG with its construction tree and a 1-stack layout of the transitive subgraph below w

either v or w (Figure 14a). So let e be an edge with parent-endpoint p in P but child-endpoint z' in $G[B] - P$, see Figure 14b. Hence the part P' in $\mathcal{P}_B$ containing z' was created later in the construction sequence than P. To show that p is either q_1 or q_2 and therefore in S, we make two observations: First, recall that every monotone vertex that is stacked onto an edge with both endpoints in P creates a new block. And second, observe that immediately after finishing P in the construction sequence, the only two edges with exactly one endpoint in P are vq_1 and wq_2, i.e., the first edge in each of Q_1^+ and Q_2^+. Together, it follows that whenever a new monotone vertex is introduced in the construction sequence, either it has both parents in P and creates a new block, or it has exactly one parent in P, namely q_1 or q_2, or it has no parent in P. Since we only consider the block B, we conclude that all vertices introduced after finishing P that have at least one parent in P actually have exactly one parent in P, namely q_1 or q_2.

Finally, assume that $G[B]$ does not contain v and w (e.g., Figure 14c or if B is the bridge in Figure 13). Then v_P is a cut vertex of H and B is a child block of v_P in the block-cut tree of H. In particular, P is the first part in $G[B]$ according to the construction sequence, and thus every edge in $G[B]$ with exactly one endpoint in P has the parent-endpoint in P and the child-endpoint in $G[B] - P$. Recall that block B was initialized as a bridge when a monotone vertex z'' was stacked onto an edge ab of Q_1 or Q_2. In this case let $S = \{a, b\}$. Then $|S| = 2$ and S clearly contains the parent-endpoint of every edge in $G[B]$ with exactly one vertex in P. Since we only need to bound the cut cover number of the partition restricted to some block B, this finishes (P4). ◆

We conclude that $\mathcal{P}$ is indeed a directed H-partition satisfying all four properties, which concludes the proof. ■

3.4 A Constant Upper Bound

By Property (P1) of Lemma 3.10, each part of the constructed directed H-partition induces the transitive subgraph below a monotone vertex. Bounding the stack number of these subgraphs is the last missing piece before we prove Theorem 1.3.

LEMMA 3.11. *Let G be a maximal outerplanar DAG with a fixed base edge and let T be its (rooted) construction tree. Then the transitive subgraph below every monotone vertex w admits a 1-stack layout.*

PROOF. Let uv denote the parent edge of w. By symmetry, we assume that w is a right child. First recall that w has at most two children a and b in T, corresponding to the children of uw, respectively vw, in G. Let G_a (G_b) denote the subgraph of G induced by u (v), w, and the transitive subgraph below a (b), see Figure 15. Note that the union of G_a and G_b contains the transitive subgraph below w. Now observe that G_a and G_b are transitive outerplanar DAGs, i.e., obtained from uw, respectively vw, by repeatedly stacking transitive children. Therefore, they have a unique topological ordering which coincides with the ordering of the vertices around the outer face, starting with the tail of their base edge and ending with its head. It is well known that one stack suffices for outerplanar graphs with this vertex ordering. A 1-stack layout for $G_a \cup G_b$ is now obtained by concatenating the layouts of $G_a - w$ and G_b, which in particular gives a 1-stack layout for the transitive subgraph below w. ■

Finally, we are ready to prove the first main result of this paper. This includes verifying the premises of Lemma 3.7, which we restate here for convenience.

LEMMA 3.7. (Restated) *Let G be a DAG with a directed H-partition $\mathcal{P}$ such that*

- *for every block B of H the subgraph $G[B]$ of G admits an s-stack layout expanding some vertex ordering of H.*

Moreover, let T be the block-cut tree of H rooted at some block of H, such that for every cut vertex v of H with child blocks $C_1, \dots, C_k$ in T the following holds:

- *For $i = 1, \dots, k$, the intersection of $G[v]$ with the neighborhood of $G[C_i - v]$ consists of a single edge $e_i \in E(G[v])$.*
- *Edges $e_1, \dots, e_k$ can be covered with at most p directed paths in $G[v]$.*
- *For each edge $e \in E(G[v])$ we have $e = e_i$ for at most t indices $i \in \{1, \dots, k\}$.*

Then $\mathrm{sn}(G) \leq 4spt$.

THEOREM 1.3. (Restated) *The stack number of outerplanar DAGs is bounded by a constant. Moreover, every outerplanar DAG G has* $\mathrm{sn}(G) \leq 24776$.

PROOF. Without loss of generality we assume that G is a maximal outerplanar DAG. This is justified as the stack number is monotone under taking subgraphs and because a non-maximal outerplanar DAG G can easily be extended into a maximal one: Add (undirected) edges to G as long as the underlying undirected graph remains outerplanar. Then take any topological ordering $\prec$ of G and orient each added edge from its left endpoint in $\prec$ to its right endpoint in $\prec$.

Fix a base edge for G and hence the corresponding construction tree. By Lemma 3.10 there is a directed H-partition $\mathcal{P}$ of G satisfying properties (P1)–(P4). In particular, by property (P3) the DAG H is block-monotone. Thus by Corollary 3.4 there is an h-stack layout $\prec_H$ of H with $h \leq 258$. Further, by property (P1), every part of $\mathcal{P}$ induces a transitive subgraph below some monotone vertex of G and as such admits a 1-stack layout by Lemma 3.11.

Now we seek to apply Lemma 3.7, for which we have to check its premises:

— By property (P4), for every block B of H the directed B-partition $\mathcal{P}_B$ of $G[B]$ has cut cover number at most $w = 4$. Thus, by Lemma 3.6, graph $G[B]$ admits an H-expanding stack layout using at most $s \leq 3wh + 1 = 3 \cdot 4 \cdot 258 + 1 = 3097$ stacks.

— By property (P2), in each part $P \in \mathcal{P}$ there are $p = 2$ directed paths Q_1, Q_2 in $G[P]$ such that all vertices stacked onto edges of $G[P]$ are stacked onto an edge of Q_1 or Q_2. As G is outerplanar, we additionally get that at most $t = 1$ vertex is stacked onto each of those edges.

Seeing all premises fulfilled, Lemma 3.7 yields that the stack number of G is at most $\mathrm{sn}(G) \leq 4spt \leq 4 \cdot 3097 \cdot 2 \cdot 1 = 24776$. ■

4. Directed Acyclic 2-Trees have Unbounded Stack Number

We construct a directed acyclic 2-tree G with arbitrarily large twist number (hence arbitrarily large stack number) in every topological vertex ordering $\prec$. Somewhat surprisingly, we first consider rainbows, which can be seen as the counterpart to twists and are defined as follows. A *k-rainbow*, $k \geq 1$, is a set of k edges that pairwise nest with respect to $\prec$. While in general, vertex orderings with small stack number (hence small twist number) are allowed to have arbitrarily large rainbows, we first argue that there is a very large rainbow in $\prec$ for our constructed 2-tree G, and then use that rainbow as a lever to slowly find larger and larger twists in $\prec$.

We start with a straightforward auxiliary lemma. For this consider a triangle with vertices $u \prec v \prec w$ with vertex ordering $\prec$. Then we call u the *left vertex*, v the *middle vertex*, and w the right vertex. We further call a set of pairwise vertex-disjoint triangles *well-interleaved* with respect to some vertex ordering if we first have all left vertices, then all middle vertices, and finally all right vertices. Note that the ordering of the vertices within each group is not determined.

For the proof, we use the Erdős-Szekeres theorem which states that every sequence of length $p \cdot q$ consisting of pairwise distinct integers contains a monotonically increasing sequence of length p or a monotonically decreasing sequence of length q.

LEMMA 4.1. *If k^3 triangles are well-interleaved with respect to a vertex ordering $\prec$, then there is a k-twist.*

PROOF. Consider well-interleaved triangles $T_i = (u_i, v_i, w_i)$ for $i = 1, \ldots, k^3$. Without loss of generality we have $u_1 \prec \cdots \prec u_{k^3} \prec v_1, \ldots, v_{k^3} \prec w_1, \ldots, w_{k^3}$, that is, only the ordering within the v-vertices and within the w-vertices is unknown. Among the v-vertices, the Erdős-Szekeres theorem with $p = k$ and $q = k^2$ yields an increasing sequence of k indices, i.e., a k-twist between u- and v-vertices with which we are done, or a decreasing sequence of k^2 indices with which we continue. From now on, we only consider these k^2 indices. Again, by Erdős-Szekeres, there

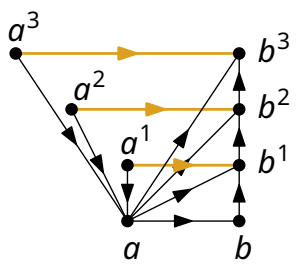


Figure 16. Construction of $T(ab)$ with the edge set $E(ab)$ (orange)

either is an increasing sequence of length k among the k^2 considered w-vertices yielding a k-twist with the corresponding u-vertices. Or there is a decreasing sequence of length k giving a k-twist between the v- and w-vertices. ■

THEOREM 1.4. (Restated) *The stack number of DAGs of treewidth* 2 *is unbounded. Moreover, for every* $k \geq 1$ *there exists a monotone* 2*-tree* G *with* $\mathrm{sn}(G) \geq k$ *in which at most two vertices are stacked onto each edge.*

We remark that we actually prove a slightly stronger statement, namely that the twist number is at least k, which in turn is a lower bound on the stack number.

PROOF. Let $k \geq 1$ be fixed. Below we construct a 2-tree G with twist number $\mathrm{tn}(G) \geq k$. The proof is split into two parts. First we construct the 2-tree G before proving that every topological vertex ordering contains a k-twist.

Construction of G. We define the desired 2-tree G via a sequence of 2-trees $G_0 \subset G_1 \subset \cdots \subset G_k$ with $G_k = G$. For each $t = 0, \ldots, k$ we shall have a matching $E_t \subset E(G_t)$ such that in G_t no vertex is stacked onto any edge in E_t, and $E_0, \ldots, E_k$ are pairwise disjoint.

We start with G_0 being a single edge ab oriented from a to b, and $E_0 = \{ab\}$. Having defined G_t and E_t for some $0 \leq t < k$, we define G_{t+1} and E_{t+1} as follows: We use each edge $ab \in E_t$ (directed from a to b) as the base edge for a particular 2-tree that we denote by $T(ab)$ and that is constructed as follows: Let N be a large enough integer (to be specified below).

- Add a sequence $b^1, \ldots, b^N$ of vertices, where b^j is stacked as a right child onto the edge ab^{j-1} (putting $b^0 = b$).
- Add a sequence $a^1, \ldots, a^N$ of vertices, where a^j is stacked as a left child onto the edge ab^j.
- Denote by $E(ab)$ the matching $E(ab) = \{a^j b^j \mid j \in \{1, \ldots, N\}\}$.

See Figure 16 for an illustration. Observe that $T(ab)$ involves no transitive stackings, that at most two vertices are stacked onto each edge of $T(ab)$, and that in $T(ab)$ no vertex is stacked onto an edge in $E(ab)$. Further observe that in every vertex ordering of $T(ab)$ the vertices $b^1, \ldots, b^N$ come (actually in that order) to the right of a and b, while the vertices $a^1, \ldots, a^N$ come (not necessarily in that order) to the left of a and b. In particular, a and b are consecutive in every vertex ordering of $T(ab)$.

Now let G_{t+1} be the 2-tree obtained from G_t by adding the 2-tree $T(ab)$ onto all edges $ab \in E_t$. Further let E_{t+1} be the union of the matchings $E(ab)$ for all edges $ab \in E_t$. Apart from the exact value of N, this completes the definition of G_{t+1} on the basis of G_t, and hence the definition of $G = G_k$.

For our proof below to work, we require that N is an enormous (but constant) number in terms of k. We set $N = r_1 k$, where $r_1, \dots, r_k$ is a sequence of integers defined recursively by

$$r_k = 1 \qquad \text{and}$$
$$r_t = 2 \cdot k^3 (2k^3)^{1+r_{t+1}k} \qquad \text{for } t = k-1, \dots, 1.$$

◆

Twist number of G. Let $\prec$ be an arbitrary vertex ordering of G. We give an inductive proof of a slightly stronger statement than the existence of a k-twist, for which we need the following definition. For positive integers r, t we call a matching a *t-twist with an r-thick edge* if it is obtained from a t-twist by replacing one edge by an r-rainbow. In particular, the r-rainbow is vertex-disjoint from the remaining $(t-1)$-twist and each rainbow-edge crosses each twist-edge. Our long-term goal is to find such a t-twist with a thick edge in G_t. However, we may also find a k-twist along the way, in which case we can stop immediately. Thus, throughout the proof we always assume that we do not find a k-twist. Under this assumption, we now give our stronger statement that we prove by induction: For every $t = 1, \dots, k$, the subgraph G_t of G contains a t-twist with an r_t-thick edge, where the r_t-rainbow consists of edges in E_t. We start with a huge rainbow, which rapidly decreases while increasing the size of the twist by 1 in each step until we obtain a k-twist for $t = k$. ◆

For $t = 1$, a 1-twist with an r_1-thick edge is simply an r_1-rainbow. Recall that $G_1 = T(ab)$ for the only edge $ab \in E_0 = E(G_0)$. As mentioned above, we have $a^1, \dots, a^N \prec a \prec b \prec b^1 \prec b^2 \prec \cdots \prec b^N$. By the Erdős-Szekeres theorem, the ordering of N such a-vertices according to $\prec$ contains a k-element subsequence with monotonically increasing indices or an N/k-element subsequence with monotonically decreasing indices. The former case gives a k-twist from the corresponding a-vertices to the b-vertices, so we can stop. Otherwise, the latter case gives a rainbow formed by $N/k = r_1$ edges of E_1, as desired.

Now, for $t \geq 1$, assume that we have a t-twist T with an r_t-thick edge, where $R \subseteq E_t$ denotes the r_t-rainbow. We aim to find an entirely new rainbow $R' \subseteq E_{t+1}$ of size r_{t+1} and an edge e' crossing all edges of R', where all these edges start in the region spanned by the starting points of R and end in the region spanned by the endpoints of R. Together with the $t-1$ edges of $T - R$, this forms a $(t+1)$-twist with an r_{t+1}-thick edge, see Figure 17. For this, recall that in the construction of G, the 2-tree $T(ab)$ is added to each edge $ab \in R \subseteq E_t$. To find R' and e', we follow two steps: First, we analyze the edges $ab \in R$ and their right children. Here, we either find a large subset of edges having their children far to the right (Figure 18 left), which yields well-interleaved triangles and thus a k-twist by Lemma 4.1. Or we have the other extreme that

Figure 17. Left: A 4-twist with a 3-thick edge. Right: A 2-rainbow R' crossed by an edge e' resulting in a 5-twist with a 2-thick edge.

Figure 18. Left: Three consecutive edges of the rainbow R with their first right children (blue vertices) skipping sufficiently many vertices of R such that three well-interleaved triangles are formed. (Note that for simplification of the proof, *all* children are required to skip at least k^3 vertices of R, although this is not necessary and not shown here for better readability, e.g., the child of the outermost edge.) Right: Two edges of the form $ab \in S_s \subseteq R$ with the right children of $T(ab)$ immediately following b.

many edges of R have their children close to their right endpoint (Figure 18 right). In the second step, we find R' and e' in the 2-tree $T(ab)$ of such an edge ab.

For the first part, consider an edge $ab \in R$ and the first right child b^1 in $T(ab)$, and count the number of vertices of R that are *skipped*, i.e., the number of endpoints of edges of R between b and b^1. If there are k^3 (along the nesting order) consecutive edges ab in R such that each b^1 skips at least k^3 vertices of R, then we obtain k^3 well-interleaved triangles (Figure 18 left) and therefore a k-twist by Lemma 4.1. As we are done in this case, we assume the other case in which we have a set $S \subseteq R$ with the following two properties: First, S consists of at least r_t/k^3 edges whose first right child skips less than k^3 vertices of R. And second, among each k^3 consecutive edges of R at least one belongs to S. It follows that among the edges of S, there is a subset $S_1 \subseteq S$ of $r_t/(2k^3)$ edges ab such that the right child b^1 does not skip any vertex of S_1; choose one in each block of $2k^3$ consecutive edges in R (Figure 19).

Next, consider the second right children, i.e., the vertex b^2 for edges $ab \in S_1$. With the same argument as above, we have one of the two extremes in Figure 18: Either k^3 consecutive

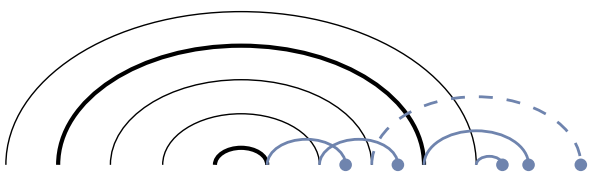


Figure 19. Five consecutive edges of R with their first right children, edges to the left parent are omitted for readability. Edges to children skipping more than one vertex of R are drawn dashed. Each block of two consecutive edges of R contains an edge whose child skips at most one vertex. The chosen subrainbow S_1 is highlighted thick.

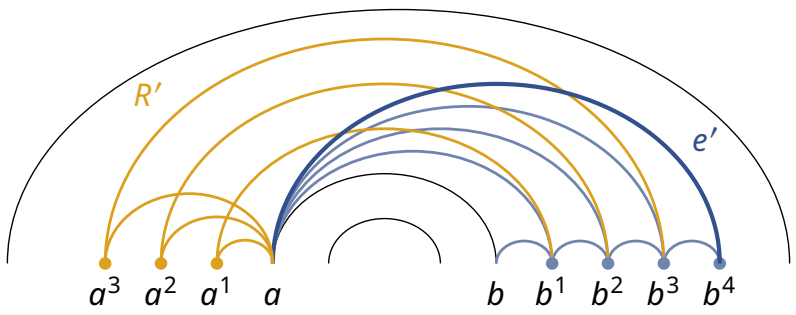


Figure 20. The new rainbow R' and the edge $e' = ab^s$ crossing R'

edges of S_2 have their second right children far to the right forming well-interleaved triangles. Or there is a subrainbow $S_2 \subseteq S_1 \subseteq S \subseteq R$ of size $r_t/(2k^3)^2$ such that for every edge $ab \in S_2$, its second right child b^2 does not skip any vertex of S_2. Repeating the argument s times, we obtain a subrainbow of $S_s \subseteq \cdots \subseteq S_2 \subseteq S_1 \subseteq R$ such that each edge ab has many right children $b^1, b^2, \ldots, b^s$ not skipping any vertex of S_s. Note that the size of the subrainbow shrinks by a factor of $2k^3$ in each step. Since $r_t = k^3(2k^3)^s$ for $s = 1 + r_{t+1}k$, after s steps we are left with a set $S_s \subseteq R$ of k^3 edges of the form ab such that $b^1 \prec \cdots \prec b^s$ immediately follow b, i.e., no other vertex of S_s or the considered right children is between b and b^s, which concludes the first part.

The second part considers the left children of the 2-trees $T(ab)$ to find R' and e'. Observe that if all k^3 edges of $S_s \subseteq R$ have a left child outside the region spanned by R, then these children together with their parents form k^3 well-interleaved triangles. As this yields a k-twist by Lemma 4.1, we may assume that there is an edge $ab \in S_s$ with the children $a^1, \ldots, a^{s-1}$ below the outermost edge of R, where again $s = 1 + r_{t+1}k$. Among these children, we find the starting points of the new rainbow R' using the Erdős-Szekeres theorem: Either we find a sequence of k increasing indices, then the respective a^ib^i-edges form a k-twist and we are done. Or we find a sequence of r_{t+1} decreasing indices, then the respective a^ib^i-edges form our desired r_{t+1}-rainbow R'. Finally, we choose $e' = ab^s$ as an edge that crosses all edges of R', see Figure 20. Combing the $(t-1)$-twist $T - R$ with the edge e' and the rainbow $R' \subseteq E_{t+1}$, and recalling that the 2-tree $T(ab)$ including R' and e' is below the outermost edge of R, we obtain a $(t+1)$-twist with an r_{t+1}-thick edge. ∎

5. Conclusion and Open Problems

We proved that outerplanar DAGs have bounded stack number (Theorem 1.3) and that monotone 2-trees with at most two vertices stacked on every edge have unbounded stack number (Theorem 1.4). In both cases we solved long-standing open problems or conjectures. In doing so we got pretty close to pinpointing the boundary between bounded and unbounded stack number of directed acyclic 2-trees, hence DAGs of treewidth 2. However, several interesting questions worth to be considered remain open.

Our first open problem is about the exact upper bound for the stack number of outerplanar DAGs. We are certain that the number of 24776 stacks required by our approach can be lowered. The best known lower bound is an outerplanar DAG (that is even upward planar) presented by Nöllenburg and Pupyrev [51] that requires four stacks.

OPEN PROBLEM 5.1. *What is the largest stack number of outerplanar DAGs exactly?*

Our family of 2-trees constructed to prove Theorem 1.4 is not upward planar. This motivates the following open problem to further narrow the gap between bounded and unbounded stack number.

OPEN PROBLEM 5.2. *Is the stack number of upward planar 2-trees bounded?*

In fact, this is just a special case of the same question for general upward planar graphs. Here the best lower bound is an upward planar graph requiring five stacks compared to an $O((n \log n)^{2/3})$ upper bound, where n is the number of vertices [42].

OPEN PROBLEM 5.3. *Is the stack number of upward planar graphs bounded?*

References

[1] **Md. Jawaherul Alam**, **Michael A. Bekos**, **Vida Dujmović**, **Martin Gronemann**, **Michael Kaufmann**, **and Sergey Pupyrev**. On dispersable book embeddings. *Theoretical Computer Science*, 861:1–22, 2021. DOI (5)

[2] **Md. Jawaherul Alam**, **Michael A. Bekos**, **Martin Gronemann**, **Michael Kaufmann**, **and Sergey Pupyrev**. The mixed page number of graphs. *Theoretical Computer Science*, 931:131–141, 2022. DOI (5)

[3] **Mustafa Alhashem**, **Guy-Vincent Jourdan**, **and Nejib Zaguia**. On The Book Embedding Of Ordered Sets. *Ars Combinatoria*, 119:47–64, 2015. (5)

[4] **Mohammad Alzohairi and Ivan Rival**. Series-Parallel Planar Ordered Sets Have Pagenumber Two. *Graph Drawing (GD 1996)*, volume 1190 of *Lecture Notes in Computer Science*, pages 11–24, 1997. DOI (5)

[5] **Mohammad Alzohairi**, **Ivan Rival**, **and Alexandr Kostochka**. The Pagenumber of Spherical Lattices is Unbounded. *Arab Journal of Mathematical Sciences*, 7(1):79–82, 2001. (5)

[6] **Patrizio Angelini**, **Michael A. Bekos**, **Philipp Kindermann**, **and Tamara Mchedlidze**. On mixed linear layouts of series-parallel graphs. *Theoretical Computer Science*, 936:129–138, 2022. DOI (5)

[7] **Christopher Auer**, **Christian Bachmaier**, **Franz Josef Brandenburg**, **Wolfgang Brunner**, **and Andreas Gleißner**. Plane drawings of queue and deque graphs. *Graph Drawing (GD 2010)*, pages 68–79, Berlin, Heidelberg. Springer Berlin Heidelberg, 2011. DOI (5)

[8] **Michael J. Bannister**, **William E. Devanny**, **Vida Dujmović**, **David Eppstein**, **and David R. Wood**. Track layouts, layered path decompositions, and leveled planarity. *Algorithmica*, 81:1561–1583, 2019. DOI (5)

[9] **Michael A. Bekos**, **Giordano Da Lozzo**, **Vida Dujmović**, **Fabrizio Frati**, **Martin Gronemann**, **Tamara Mchedlidze**, **Fabrizio Montecchiani**, **Martin Nöllenburg**, **Sergey Pupyrev**, **and Chrysanthi N. Raftopoulou**. On Linear Layouts of Planar and k-Planar Graphs. *Beyond-Planar Graphs: Combinatorics, Models and Algorithms (Dagstuhl Seminar 19092)*, volume 9(2) of *Dagstuhl Reports*, pages 144–148, 2019. DOI (2, 4, 6)

[10] **Michael A. Bekos**, **Giordano Da Lozzo**, **Fabrizio Frati**, **Martin Gronemann**, **Tamara Mchedlidze**, **and Chrysanthi N. Raftopoulou**. Recognizing DAGs with page-number 2 is NP-complete. *Theoretical Computer Science*, 946:113689, 2023. DOI (6)

[11] **Michael A. Bekos**, **Stefan Felsner**, **Philipp Kindermann**, **Stephen Kobourov**, **Jan Kratochvíl**, **and Ignaz Rutter**. The rique-number of graphs. *Graph Drawing and Network Visualization (GD 2022)*, pages 371–386, Cham. Springer International Publishing, 2023. DOI (5)

[12] **Michael A. Bekos**, **Martin Gronemann**, **and Chrysanthi N. Raftopoulou**. Two-Page Book Embeddings of 4-Planar Graphs. *Algorithmica*, 75:158–185, 2016. DOI (5)

[13] **Michael A. Bekos**, **Mirco Haug**, **Michael Kaufmann**, **and Julia Männecke**. An online framework to interact and efficiently compute linear layouts of graphs, 2023. arXiv:2003.09642. DOI (5)

[14] **Michael A. Bekos**, **Michael Kaufmann**, **Fabian Klute**, **Sergey Pupyrev**, **Chrysanthi N. Raftopoulou**, **and Torsten Ueckerdt**. Four Pages are Indeed Necessary for Planar Graphs. *Journal of Computational Geometry*, 11(1):332–353, 2020. DOI (5)

[15] **Michael A. Bekos**, **Michael Kaufmann**, **Maria Eleni Pavlidi**, **and Xenia Rieger**. On the deque and rique numbers of complete and complete bipartite graphs. *Proceedings of the 35th Canadian Conference on Computational Geometry, CCCG 2023, Concordia University, Montreal, Quebec, Canada, July 31 - August 4, 2023*, pages 89–95, 2023. URL (5)

[16] **Frank Bernhart and Paul C. Kainen**. The Book Thickness of a Graph. *Journal of Combinatorial Theory, Series B*, 27(3):320–331, 1979. DOI (5)

[17] **Sujoy Bhore**, **Giordano Da Lozzo**, **Fabrizio Montecchiani**, **and Martin Nöllenburg**. On the upward book thickness problem: combinatorial and complexity results. *European Journal of Combinatorics*, 110:103662, 2023. DOI (2, 4, 6)

[18] **Carla Binucci**, **Giordano Da Lozzo**, **Emilio Di Giacomo**, **Walter Didimo**, **Tamara Mchedlidze**, **and Maurizio Patrignani**. Upward book embeddability of st-graphs: complexity and algorithms. *Algorithmica*, 85(12):3521–3571, 2023. DOI (6)

[19] **Jonathan F. Buss and Peter W. Shor**. On the Pagenumber of Planar Graphs. *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing (STOC 1984)*, pages 98–100, 1984. DOI (5)

[20] **James Davies**. Improved bounds for colouring circle graphs. *Proceedings of the American Mathematical Society*, 150(12):5121–5135, 2022. DOI (10)

[21] **Philipp de Col**, **Fabian Klute**, **and Martin Nöllenburg**. Mixed linear layouts: complexity, heuristics, and experiments. *Graph Drawing and Network Visualization (GD 2019)*, pages 460–467, Cham. Springer International Publishing, 2019. DOI (5)

[22] **Hubert de Fraysseix**, **Patrice O. de Mendez**, **and János Pach**. A Left-First Search Algorithm for Planar Graphs. *Discrete & Computational Geometry*, 13(3–4):459–468, 1995. DOI (5)

[23] **Emilio Di Giacomo**, **Walter Didimo**, **Giuseppe Liotta**, **and Stephen K. Wismath**. Book Embeddability of Series–Parallel Digraphs. *Algorithmica*, 45:531–547, 2006. DOI (2, 4, 6)

[24] **Vida Dujmovic and David R. Wood**. Graph treewidth and geometric thickness parameters. *Discret. Comput. Geom.* 37(4):641–670, 2007. DOI (5)

[25] **Vida Dujmović**, **Gwenaël Joret**, **Piotr Micek**, **Pat Morin**, **Torsten Ueckerdt**, **and David R. Wood**. Planar Graphs Have Bounded Queue-Number. *Journal of the ACM*, 67(4):1–38, 2020. DOI (3, 5, 12, 13)

[26] **Vida Dujmović**, **Pat Morin**, **and David R. Wood**. Layout of Graphs with Bounded Tree-Width. *SIAM Journal on Computing*, 34(3):553–579, 2005. DOI (5)

[27] **Vida Dujmović and David R. Wood**. On linear layouts of graphs. *Discrete Mathematics & Theoretical Computer Science*, Vol. 6 no. 2, January 2004. DOI (5)

[28] **Stefan Felsner**, **Laura Merker**, **Torsten Ueckerdt**, **and Pavel Valtr**. Linear Layouts of Complete Graphs. *Graph Drawing and Network Visualization (GD 2021)*, volume 12868 of *Lecture Notes in Computer Science*, pages 257–270, 2021. DOI (5)

[29] **Henry Förster**, **Michael Kaufmann**, **Laura Merker**, **Sergey Pupyrev**, **and Chrysanthi Raftopoulou**. Linear layouts of bipartite planar graphs. *Algorithms and Data Structures (WADS 2023)*, pages 444–459, Cham. Springer Nature Switzerland, 2023. DOI (5)

[30] **Fabrizio Frati**, **Radoslav Fulek**, **and Andres J. Ruiz-Vargas**. On the Page Number of Upward Planar Directed Acyclic Graphs. *Journal of Graph Algorithms and Applications*, 17(3):221–244, 2013. DOI (5, 6)

[31] **Joseph L. Ganley and Lenwood S. Heath**. The pagenumber of k-trees is $O(k)$. *Discrete Applied Mathematics*, 109(3):215–221, 2001. DOI (5)

[32] **Xiaxia Guan and Weihua Yang**. Embedding planar 5-graphs in three pages. *Discrete Applied Mathematics*, 282:108–121, 2020. DOI (5)

[33] **András Gyárfás**. On the Chromatic Number of Multiple Interval Graphs and Overlap Graphs. *Discrete Mathematics*, 55(2):161–166, 1985. DOI (10)

[34] **Seifollah L. Hakimi**, **John Mitchem**, **and Edward F. Schmeichel**. Star arboricity of graphs. *Discrete Mathematics*, 149(1–3):93–98, 1996. DOI (14)

[35] **Lenwood S. Heath**. Embedding Planar Graphs In Seven Pages. *25th Annual Symposium on Foundations of Computer Science (FOCS 1984)*, pages 74–83, 1984. DOI (5)

[36] **Lenwood S. Heath**, **Frank Thomson Leighton**, **and Arnold L. Rosenberg**. Comparing Queues and Stacks As Machines for Laying Out Graphs. *SIAM Journal on Discrete Mathematics*, 5(3):398–412, 1992. DOI (5)

[37] **Lenwood S. Heath and Sriram V. Pemmaraju**. Stack and Queue Layouts of Directed Acyclic Graphs: Part II. *SIAM Journal on Computing*, 28(5):1588–1626, 1999. DOI (5, 6)

[38] **Lenwood S. Heath and Sriram V. Pemmaraju**. Stack and Queue Layouts of Posets. *SIAM Journal on Discrete Mathematics*, 10(4):599–625, 1997. DOI (5)

[39] **Lenwood S. Heath**, **Sriram V. Pemmaraju**, **and Ann N. Trenk**. Stack and Queue Layouts of Directed Acyclic Graphs: Part I. *SIAM Journal on Computing*, 28(4):1510–1539, 1999. DOI (2, 3, 5, 6)

[40] **Lenwood S. Heath and Arnold L. Rosenberg**. Laying Out Graphs Using Queues. *SIAM Journal on Computing*, 21(5):927–958, 1992. DOI (5)

[41] **Le Tu Quoc Hung**. A Planar Poset which Requires 4 Pages. *Ars Combinatoria*, 35:291–302, 1993. (5)

[42] **Paul Jungeblut**, **Laura Merker**, **and Torsten Ueckerdt**. A sublinear bound on the page number of upward planar graphs. *SIAM J. Discret. Math.* 37(4):2312–2331, 2023. DOI (5, 6, 31)

[43] **Paul Jungeblut**, **Laura Merker**, **and Torsten Ueckerdt**. Directed acyclic outerplanar graphs have constant stack number. *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1937–1952, 2023. DOI (1)

[44] **Paul C. Kainen**. Some recent results in topological graph theory. *Graphs and Combinatorics*, pages 76–108, Berlin, Heidelberg. Springer Berlin Heidelberg, 1974. (5)

[45] **Kolja Knauer and Torsten Ueckerdt**. Simple Treewidth. *Midsummer Combinatorial Workshop Prague*, pages 21–23, 2012. URL (7)

[46] **Seth M. Malitz**. Genus g Graphs Have Pagenumber $O(\sqrt{g})$. *Journal of Algorithms*, 17(1):85–109, 1994. DOI (5)

[47] **Tamara Mchedlidze and Antonios Symvonis**. Crossing-Free Acyclic Hamiltonian Path Completion for Planar st-Digraphs. *Algorithms and Computation (ISAAC 2009)*, volume 5878 of *Lecture Notes in Computer Science*, pages 882–891, 2009. DOI (6)

[48] **Laura Merker**. Ordered Covering Numbers. Master's thesis, Karlsruhe Institute of Technology, 2020. URL (5)

[49] **Laura Merker and Torsten Ueckerdt**. The Local Queue Number of Graphs with Bounded Treewidth. *Graph Drawing and Network Visualization (GD 2020)*, volume 12590 of *Lecture Notes in Computer Science*, pages 26–39, 2020. DOI (5)

[50] **Laura Merker and Tortsen Ueckerdt**. Local and Union Page Numbers. *Graph Drawing and Network Visualization (GD 2019)*, volume 11904 of *Lecture Notes in Computer Science*, pages 447–459, 2019. DOI (5)

[51] **Martin Nöllenburg and Sergey Pupyrev**. On Families of Planar DAGs with Constant Stack Number. *Graph Drawing and Network Visualization (GD 2023)*, pages 135–151, Cham. Springer Nature Switzerland, 2023. DOI (2, 4–6, 10, 31)

[52] **Richard Nowakowski and Andrew Parker**. Ordered Sets, Pagenumbers and Planarity. *Order*, 6:209–218, 1989. DOI (2–5)

[53] **L. Taylor Ollmann**. On the Book Thicknesses of Various Graphs. *Proc. 4th Southeastern Conference on Combinatorics, Graph Theory and Computing*, volume 8, 1973. (5)

[54] **Sergey Pupyrev**. Improved bounds for track numbers of planar graphs. *Journal of Graph Algorithms and Applications*, 24(3):323–341, 2020. DOI (5)

[55] **Sergey Pupyrev**. Mixed linear layouts of planar graphs. *Graph Drawing and Network Visualization (GD 2017)*, pages 197–209, Cham. Springer International Publishing, 2018. DOI (5)

[56] **S. Rengarajan and C. E. Veni Madhavan**. Stack and Queue Number of 2-Trees. *Computing and Combinatorics (COCOON 1995)*, volume 959 of *Lecture Notes in Computer Science*, pages 203–212, 1995. DOI (5)

[57] **Maciej M. Sysło**. Bounds to the Page Number of Partially Ordered Sets. *Graph-Theoretic Concepts in Computer Science (WG 1989)*, volume 411 of *Lecture Notes in Computer Science*, pages 181–195, 1990. DOI (5)

[58] **Mitsunori Togasaki and Koichi Yamazaki**. Pagenumber of pathwidth-k graphs and strong pathwidth-k graphs. *Discrete Mathematics*, 259(1–3):361–368, 2002. DOI (5)

[59] **William T. Tutte**. A Theorem on Planar Graphs. *Transactions of the American Mathematical Society*, 82(1):99–116, 1956. DOI (5)

[60] **Jennifer Vandenbussche**, **Douglas B. West**, **and Gexin Yu**. On the pagenumber of k-trees. *SIAM Journal on Discrete Mathematics*, 23(3):1455–1464, 2009. DOI (5)

[61] **Avi Wigderson**. The Complexity of the Hamiltonian Circuit Problem for Maximal Planar Graphs. Technical report, Electrical Engineering and Computer Science Department, Princeton University, 1982. URL (5)

[62] **Lasse Wulf**. Stacked Treewidth and the Colin de Verdière Number. Bachelor's thesis, Karlsruhe Institute of Technology, 2016. URL (7)

[63] **Mihalis Yannakakis**. Embedding Planar Graphs in Four Pages. *Journal of Computer and System Sciences*, 38(1):36–67, 1989. DOI (5)

[64] **Mihalis Yannakakis**. Planar Graphs that Need Four Pages. *Journal of Combinatorial Theory, Series B*, 145:241–263, 2020. DOI (5)