Karlsruhe Institute of Technology

Mixed Page Number of Planar Directed Acyclic Graphs

Bachelor Thesis of

Deborah Haun

At the Department of Informatics
Institute of Theoretical Informatics

Reviewers:	PD Dr. Torsten Ueckerdt
	T.T.-Prof. Dr. Thomas Bläsius
Advisors:	Laura Merker, M.Sc.

Time Period: 24th November 2022 – 24th March 2023

Statement of Authorship

Ich versichere wahrheitsgemäß, die Arbeit selbstständig verfasst, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde sowie die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet zu haben.

Karlsruhe, March 24, 2023

Abstract

A *q-queue s-stack layout* consists of a topological ordering of the vertices of a graph and a partition of its edges into q sets of edges, called *queues*, and s sets of edges, called *stacks*, such that any two edges in the same queue do not nest and any two edges in the same stack do not cross. Here, two edges nest if the endpoints of one edge are located between the endpoints of the other edge. Two edges cross if their endpoints alternate in the topological ordering. The minimum number $q+s$ of queues and stacks required for a q-queue s-stack layout is called the *mixed page number*.

We investigate the mixed page number of upward planar graphs and directed acyclic 2-trees. We find a subclass of upward planar bipartite graphs whose mixed page number is bounded by a constant. In addition, we give a lower bound of 3 on the mixed page number of upward planar bipartite graphs and find an upward planar bipartite graph, whose mixed page number is strictly smaller than its stack and queue number. For directed acyclic 2-trees, we show that the mixed page number is unbounded. In contrast, we present a family of directed acyclic 2-trees with bounded mixed page number but unbounded stack and queue number.

Deutsche Zusammenfassung

Ein *q-Queue s-Stack Layout* besteht aus einer topologischen Sortierung der Knoten eines Graphen und einer Partition dessen Kanten in q Kantenmengen, die *Queues* genannt werden, und s Kantenmengen, die *Stacks* genannt werden, sodass je zwei Kanten in der gleichen Queue nicht verschachtelt sind und je zwei Kanten in dem gleichen Stack sich nicht kreuzen. Hierbei sind zwei Kanten verschachtelt, wenn die Endpunkte der einen Kante zwischen den Endpunkten der anderen Kante liegen. Zwei Kanten kreuzen sich, wenn ihre Endpunkte alternieren bezüglich der topologischen Knotenordnung. Die minimale Anzahl $q + s$ an Queues und Stacks, die für ein q-Queue s-Stack Layout benötigt werden, wird *Mixed Page Number* genannt.

Wir untersuchen die Mixed Page Number von upward planaren Graphen und gerichteten azyklischen 2-Bäumen. Wir finden eine Unterklasse von upward planaren bipartiten Graphen, deren Mixed Page Number durch eine Konstante beschränkt ist. Zusätzlich geben wir eine untere Schranke von 3 für die Mixed Page Number von upward planaren bipartiten Graphen an und finden einen upward planaren bipartiten Graphen mit einer Mixed Page Number echt kleiner als dessen Stack und Queue Number. Für gerichtete azyklische 2-Bäume zeigen wir, dass die Mixed Page Number unbeschränkt ist. Im Gegensatz dazu geben wir eine Familie von gerichteten azyklischen 2-Bäumen mit beschränkter Mixed Page Number, aber unbeschränker Stack und Queue Number an.

Contents

1. Introduction

First suggested by Heath and Rosenberg in 1992 [HR92], the mixed page number combines the concept of queue numbers, which they newly introduced, with the concept of stack numbers, which has been studied before since it was introduced by Bernhart and Kainen in 1979 [BK79]. Even though the stack and queue numbers have been investigated extensively over the past decades, works on the mixed page number are still scarce, especially regarding planar directed graphs, on which we focus in this thesis.

Stack, queue, and mixed page numbers are all properties of graphs that relate to linear layouts. A *linear layout* of a graph consists of a total vertex ordering and a partition of the edges into sets with certain properties. If no two edges of such a set cross with respect to the vertex ordering, the set is called a *stack*. Here, two edges cross if their endpoints alternate in the vertex ordering. The edge set is called a *queue*, if no two edges from the set nest with respect to the ordering, i.e., the endpoints of one edge are not located between the endpoints of the other edge. If we only allow the edges to be partitioned into stacks, the minimum number of stacks required is called the *stack number*. If we analogously only allow queues, the minimum number of queues required is called the *queue number*. A *mixed linear layout* of a graph allows each set of edges to form either a stack or a queue. The minimum number of such sets is then called the *mixed page number*. An example of a graph's mixed linear layout is given by Figure 1.1. In this thesis we investigate the mixed page number of special classes of planar directed acyclic graphs, shortly planar DAGs. Before we introduce the basic concepts in more detail, we give a brief overview of the current research and some applications that motivate the topic of this thesis.

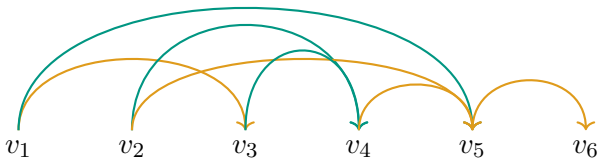


Figure 1.1: A mixed linear layout. The vertex ordering is given from left to right. The edges are partitioned into one queue (orange) and one stack (green).

1.1 Motivation

Stack and queue layouts form an important research topic, which has been studied extensively in the past few decades [FFRV13, FUW21, HPT99, JMU22a, JMU22b, KMU18, NP89, NP21, Pup23]. One intriguing question in this field is whether the stack number of upward planar graphs is bounded by a constant. This question was first raised by Nowakowski and Parker in 1989 [NP89], and although it remains unanswered, there is much work on answering this question [FFRV13, JMU22a, JMU22b, NP21]. But also directed acyclic 2-trees have been studied recently [JMU22b]. Regarding queue layouts, one of the most important problems is to find a bound on the queue number of posets in terms of their width (see Section 1.3). Mixed linear layouts provide a generalisation of stack and queue layouts that allows us to investigate those layouts under a different perspective. Nevertheless, mixed linear layouts are also of great interest in its own right, although they have mainly been studied for undirected graphs so far. For instance, originally, it was conjectured that every undirected planar graph admits a 1-queue 1-stack layout [HR92]. However, this conjecture was refuted [Pup18], and it does not even hold for the subclass of undirected 2-trees [ABKM22].

In contrast to stack and queue layouts, there is not much work on mixed linear layouts of directed graphs. Therefore, we initiate the study of mixed page numbers of planar DAGs. In doing so, we investigate exactly those graph classes that are of great interest in terms of stack layouts, namely upward planar graphs and directed acyclic 2-trees. The expectations are that studying mixed linear layouts could be an intermediate step on the way to answering questions about stack layouts, such as whether the stack number of upward planar graphs is bounded by a constant. For directed acyclic 2-trees, the stack number is known to be unbounded [JMU22b]. However, an unbounded mixed page number would strengthen this proposition.

1.2 Applications

One recurrent issue in computer science is the comparison of stacks and queues regarding their properties and power in several applications. Stack and queue numbers as graph parameters can be used to measure the power of stacks and queues with regard to the given graph [HLR92]. If the stack number is strictly smaller than the queue number, then we can infer that stacks are more powerful in a given context. Since graphs are often used to model practical applications, the stack and queue numbers are not only of graph-theoretical interest, but also tell us something about the power of stacks and queues in these practical applications. Beyond this, there are applications that in total need less stacks and queues when using both mixed together. Here, the mixed page number comes into play.

One application is permuting elements using parallel stacks and queues. Tarjan [Tar72] asked which permutations are possible to sort depending on the number of stacks or queues used. Given are a source queue, in which an initial permutation of numbers $\pi = (p_1, \ldots, p_n)$ with $\pi(i) = p_i$ is placed, a sink queue, and m additional stacks or queues. In each step, one number is moved from one stack or queue to another with the aim of getting all numbers ordered in the sink queue. By doing so, only the first number that has been inserted can be removed from a queue, and only the last number that has been inserted can be removed from a stack. The problem now asks for the number of stacks or queues required to sort a specific permutation. This question can be modelled using graphs, and in particular stack and queue layouts, which is what Chung, Leighton, and Rosenberg did [CLR87], although only with stacks and not with queues. For this, they constructed a bipartite graph $G_n = (V_n, E_n)$ with vertices $V_n = \{a_1, \ldots, a_n\} \cup \{b_1, \ldots, b_n\}$ and edges $E_n = \{a_i, b_i \mid 1 \leq i \leq n\}$. The problem of sorting the permutation π with m stacks is then equivalent to embedding the graph G_n in a stack layout with m stacks and the fixed vertex

ordering $a_1, \ldots, a_n, b_{\pi(1)}, \ldots, b_{\pi(n)}$. However, the initial question asked by Tarjan [Tar72] also referred to queues. Here, we can apply the same approach as for stacks to queues. Then, sorting π with m queues is equivalent to finding a queue layout with m queues and the same fixed ordering. Additionally, we extend the problem by allowing both stacks and queues for sorting the permutation, which correlates with using a mixed linear layout for G_n.

For linear layouts of directed graphs, scheduling of parallel processors is one important application, as described by Heath, Lenwood, and Rosenberg [HLR92]. In this scenario, we consider a data manager consisting of queues, where again only the first element inserted can be removed. Every process takes its input from these queues when it starts and places its output there when it terminates. Thus, when a process starts, all of its inputs have to be at the heads of the queues. This problem can be transferred to a graph problem, where the vertices represent the processes and the directed edges represent the dependencies between those processes. Then, the queues from the data manager can be abstracted as the queues of a queue layout of the graph, whereby the queue number of the graph is equivalent to the minimum number of queues required in the data manager. Again, it is also conceivable to allow stacks in the data manager, where only the last element inserted can be removed. With only stacks in the data manager, this would correspond to a stack layout of the graph. With both stacks and queues the data manager would correlate to a mixed linear layout.

Another prominent application for linear layouts of graphs is Very Large-Scale Integration (VLSI) design. The Diogenes approach, proposed by Rosenberg [Ros83], is a strategy for designing testable fault-tolerant arrays of processors that is equivalent to our graph layout problem. Thereby, processing elements are arranged on a physical or logical line, which can be modelled as vertices of a graph in a fixed vertex ordering. The processing elements are connected with bundles of wires, which are organised and function as stacks or queues and can be modelled as such in the graph layout. The number of bundles should now be minimised, which is equivalent to finding the mixed page number, i.e., the minimum number of stacks and queues required for the graph layout. The Diogenes approach was the main motivation to investigate queues and mixed linear layouts, and not only stack layouts [HR92].

Furthermore, there are several other applications of linear layouts that are worth mentioning here. For instance, stack layouts are used in computational biology to describe RNA structures [CDD+12], whereby bases are represented by vertices and base pairs as edges. Then, a stack of edges corresponds to a secondary structure. Finally, stack layouts have applications in complexity theory, since the reachability problem over graphs embedded on three pages is NL-complete, while it can be solved with logarithmic space for graphs embedded on two pages [PTV12]. Here, NL denotes the class of decision problems that can be solved by a nondeterministic Turing machine on logarithmic space.

1.3 Related Work

A significant part of the related work refers to undirected graphs. For instance, Yannakakis proved that four stacks suffice for each undirected planar graph [Yan89], which is in fact a tight upper bound [BKK+20, Yan20]. Meanwhile, a bound on the queue number for undirected planar graphs ranges between 4 [ABG+20] and 42 [BGR23]. Since the mixed page number is always at most the stack or queue number, the upper bound of 4 also applies to the mixed page number. Here, it is not known whether this bound is tight. In 1992 Heath and Rosenberg [HR92] even conjectured that every planar graph admits a 1-queue 1-stack layout. However, Pupyrev [Pup18] disproved this conjecture in 2018, but conjectured that the same holds for planar bipartite graphs (definition see Section 2.1). In turn, this conjecture was also refuted [FKM+23]. Nevertheless, the mixed page number of

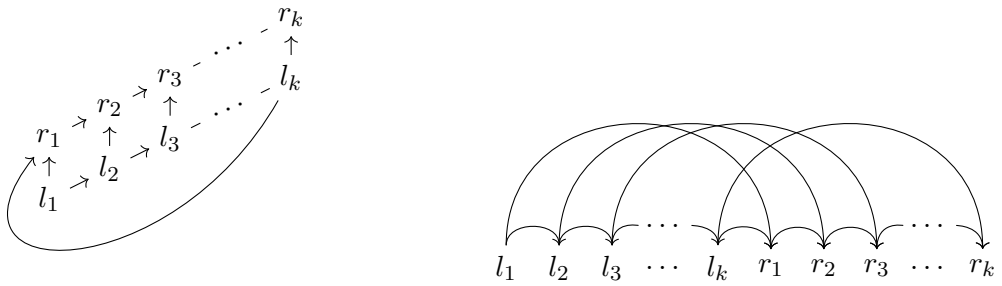


Figure 1.2: A planar DAG G_k on $2k$ vertices (left) with stack number at least k, since there is a k-twist in the unique topological ordering (right). For G_k k stacks are also sufficient.

planar bipartite graphs is at most 2 since this is a tight upper bound on the stack number [dFOdMP95, Ove98]. Again, for the queue number no tight upper bound is known, but it ranges between 2 and 28 [FKM$^+$23]. Furthermore, there are other classes of undirected graphs that are not necessarily planar, for which mixed linear layouts were explicitly studied, such as complete and complete bipartite graphs [ABG$^+$22b].

However, we focus on planar directed acyclic graphs in this thesis. Studying linear layouts of DAGs was first proposed by Nowakowski and Parker in 1989 [NP89]. The main difference from linear layouts of undirected graphs is that the vertex ordering additionally has to be topological (definition see Section 2.3). As a consequence, not every vertex ordering is valid, which makes it harder to partition the edges into few stacks and queues. Hence, if k stacks or queues suffice for a linear layout of a directed graph, they also suffice for the underlying undirected graph, but the converse does not apply. For instance, unlike for undirected planar graphs, the stack number of planar DAGs is unbounded, i.e., there are planar DAGs with n vertices that require $\Theta(n)$ stacks. An example for such a planar DAG was provided by Heath, Pemmaraju, and Trenk [HPT99], and is illustrated in Figure 1.2. However, the stack number is known to be bounded by a constant for some subclasses of planar DAGs. The goal is now to find such subclasses with bounded stack numbers that are as large as possible. Two of those subclasses currently under investigation are upward planar graphs (see Section 2.1) and directed acyclic 2-trees (see Section 2.2), which will also play a major role in this thesis.

One of the most important and long-standing open questions regarding linear layouts is whether the stack number of upward planar graphs is bounded by a constant. The question was first raised by Nowakowski and Parker in 1989 [NP89], and there was not much progress in answering this question until recently, despite some effort on subclasses of upward planar graphs. One simple example of upward planar graphs with bounded stack number are directed trees, or more general, directed forests, whose stack number is 1 [NP89]. Further, upward planar 3-trees have a bounded stack number [FFRV13]. Moreover, it is known that unicyclic DAGs have stack number 2 [HPT99], and that the stack number of outerpath DAGs [NP21] and outerplanar DAGs [JMU22b] is bounded by a constant. By restricting these graph classes to upward planar graphs, we obtain subclasses of upward planar graphs with bounded stack number. Definitions of the graph classes mentioned here are given in the respective paper.

The first non-trivial, sublinear upper bound on the stack number $\mathrm{sn}(G) \in \mathcal{O}((n \log n)^{2/3}) \subseteq o(n)$ for all upward planar graphs G with n vertices was recently given by Jungeblut, Merker, and Ueckerdt [JMU22a]. However, there is no upward planar graph known that requires more than five stacks [JMU22a], which motivates the search for a better upper bound to close this gap between upper and lower bound. Although a major part of the current research in the field of linear layouts is devoted primarily to the stack number, a

study of mixed linear layouts, to which this thesis is dedicated, has often been proposed [HR92, NP21]. Apart from the variety of applications that require mixed linear layouts, another reason to study them is that this could also make a significant contribution to the investigation of stack and queue layouts. For the upward planar graphs considered here, it is not known whether the mixed page number is bounded, neither is it for the stack number. However, the queue number is known to be unbounded, as we will see in examples given in Section 3.1.

In contrast to upward planar graphs, it is meanwhile indeed known that the stack number of directed acyclic 2-trees is unbounded, as Jungeblut, Merker, and Ueckerdt recently showed [JMU22b]. Simultaneously, they proved that the stack number is bounded by a constant for the subclass of directed acyclic 2-tress, where at most one vertex is stacked onto each edge, except for the base edge, onto which up to two vertices are stacked (definitions see Section 2.2). This graph class is equivalent to the class of directed acyclic maximal outerplanar graphs. From this, it follows that the stack number is bounded for upward outerplanar graphs, as already mentioned above. Outerplanar graphs are graphs that admit a crossing-free embedding into the plane with all vertices incident to the outer face (see Section 2.1). Notably, this is one of the largest upward planar graph classes for which the stack number is known to be bounded. However, the upper bound that they give is 24776, while at the same time there is no known outerplanar DAG that requires more than four stacks [NP21]. Again, this motivates further investigation of outerplanar DAGs and the related class of 2-trees to lower this upper bound to close the gap to the lower bound. Further, it is easy to see that the queue number is unbounded for directed acyclic 2-trees. For this, an example is given in Proposition 5.3. However, for the mixed page number it is only known that there are directed acyclic 2-trees that do not admit a 1-queue 1-stack layout [ABKM22]. Hence, it is still open whether the mixed page number of directed acyclic 2-trees is bounded by a constant.

As already mentioned, for the classes of planar DAGs considered here, namely upward planar graphs and directed acyclic 2-trees, the queue number is unbounded. Nevertheless, there are indeed classes of planar DAGs with bounded queue number. For instance, the queue number of planar partially ordered sets, shortly planar posets, is bounded by a linear function of their width. A *poset* $P = (M, \prec)$ is a set of elements M with a binary, asymmetric, and transitive relation $\prec$. For $a, b \in M$, a relation $a \prec b$ is a *cover* if it is not implied by transitivity. Then, the directed graph $G_P = (M, E)$ with $E = \{(a, b) \mid a \prec b$ is a cover relation in $P\}$ is called the *cover graph* of P. We remark that the cover graph of a planar poset is additionally upward planar. A queue (or stack) layout of a poset P is then a queue (or stack) layout of its cover graph G_p. Two elements $a, b \in M$ are called *comparable* if $a \prec b$ or $b \prec a$, and *incomparable* otherwise. The *width* of a poset denotes the maximum number of pairwise incomparable elements. Then, the queue number of a poset with width w is at most $3w - 2$ [KMU18]. A lower bound on the queue number of planar posets is also given by their width, as there are planar posets with width w and queue number exactly w [KMU18].

Further, also general posets that are not necessarily planar are currently under investigation. Similar as for the planar posets, their queue number is also bounded by a function of the poset's width. Precisely, the queue number of a poset with width w is at most $(w-1)^2+1$ [ABG+22a]. Asymptotically, this matches the best known lower bound of $w^2/8$ [FUW21]. For two-dimensional posets (definition see [Pup23]), the upper bound could even be improved to $w(w+1)/2$ [Pup23]. Note that the queue number of (planar) posets can be unbounded if their width is unbounded.

In addition, other variations such as simultaneous stack-queue layouts and local and union variants are of great interest. A *simultaneous q-queue s-stack layout* consists of a vertex

ordering $\prec$, a partition of the edges into q queues with respect to $\prec$, and a partition of the edges into s stacks with respect to $\prec$. The existence and size of such a simultaneous layout then correlates with some other graph properties. For instance, the class of graphs admitting a simultaneous q-queue s-stack layout with $s, q \in \mathcal{O}(1)$ coincides with the class of graphs with pathwidth in $\mathcal{O}(1)$ [Pup20]. The following local and union variants of stack and queue layouts are also worth mentioning here. Introduced by Merker and Ueckerdt [MU19], a s-stack layout (or a q-queue layout) is called *k-local* if every vertex has incident edges in at most k stacks (or queues). Then, the *local stack (or queue) number* of a graph G is the minimum k, such that G admits a k-local stack (or queue) layout. In this context, the familiar stack and queue layouts are referred to as *global* stack and queue layouts. Clearly, the local stack or queue number is always at most the global stack or queue number. One open problem regarding these local variants is whether there are planar graphs with a local stack number of 4. As already mentioned, there are planar graphs with global stack number 4 [BKK$^+$20, Yan20], but the largest known local stack number of a planar graph is 3 [MU19]. Further, local queue layouts are also under investigation. While the global queue number of planar graphs ranges between 4 [ABG$^+$20] and 42 [BGR23], an upper bound on the local queue number could be localised between 3 and 4 [MU20]. Moreover, the local queue number is known to be bounded in terms of a graph's treewidth [MU20]. Further, also local stack and queue numbers of directed acyclic graphs have been studied recently [Gro21]. Another interesting concept are *k-union stack layouts (or queue layouts)*, also proposed by Merker and Ueckerdt [MU19]. Here, a vertex ordering with a partition of the edges into k stacks (or queues) is searched for, where no two edges of the same connected component cross (or nest) within a stack (or queue).

1.4 Contribution

One of the most important questions in the field of linear layouts is whether the stack number of upward planar graphs is bounded by a constant. An intermediate step on the way to answering this question is to find subclasses of upward planar graphs with a bounded stack number that are as large as possible. In Section 4.1, we give such a subclass of upward planar graphs with a constant upper bound on their stack number. Precisely, we show that upward planar bipartite graphs $G = (A \cup B, E)$ with $E \subseteq A \times B$, i.e., with all edges oriented from a vertex in A to a vertex in B, have stack number at most 56 (see Corollary 4.4). Further, in Proposition 4.2 we show that the queue number is also at most 56.

One approach to answering the question, whether the stack, queue, or mixed page number of a graph class is bounded by a constant, is to fix a vertex ordering, for example "from left to right", and try to use this for all graphs of the given graph class. However, we show in Proposition 4.1 that this does not work for upward planar bipartite graphs, since the mixed page number can become arbitrarily large with respect to the fixed ordering.

Further, also lower bounds are of great interest. After all, the goal is always to close the gap between the smallest known upper bound and the largest known lower bound. Since the mixed linear layouts are widely unexplored, there is no lower bound known for the mixed page number of upward planar graphs. Therefore, we consider very specific graph families in Chapter 3 and Chapter 4, for which we find some linear layouts. Thereby, we give a lower bound of 3 on the mixed page number, firstly for general upward planar graphs (see Proposition 3.10), later even for upward planar bipartite graphs (see Proposition 4.5). In addition, we show for both graph classes that there are graphs with a mixed page number strictly smaller than the stack and queue number (see Corollary 3.9 and Proposition 4.6) This motivates the further investigation of mixed linear layouts in the first place.

	Stack Number		Queue Number		Mixed Page Number	
	l. b.	u. b.	l. b.	u. b.	l. b.	u. b.
upward planar graphs	5 [1]	?	unbounded [2] [3]		3 [4]	?
upward planar bipartite graphs $G = (A \cup B, E)$ with $E \subseteq A \times B$ [5]	3 [6]	56 [7]	3 [6]	56 [8]	?	56 [7] [8]
upward planar bipartite graphs with fixed vertex ordering	unbounded [9]		unbounded [9]		unbounded [9]	
upward planar bipartite graphs	3 [6] [10]	?	unbounded [3]		3 [10]	?
directed acyclic 2-trees	unbounded [11]		unbounded [12]		unbounded [13]	

Table 1.1: Lower bounds (l.b.) and upper bounds (u.b.) on the stack, queue, and mixed page number of graph classes considered in this thesis.

Another class of planar DAGs considered in this thesis are directed acyclic 2-trees. In contrast to the upward planar graphs, the stack number of directed acyclic 2-trees is known to be unbounded [JMU22b]. However, up until now, it is not known if the same applies to the mixed page number. While investigating the only family of directed acyclic 2-trees, for which the stack number is known to be unbounded [JMU22b], we find out that this graph family has unbounded stack and queue number but bounded mixed page number (see Theorem 5.1). Nevertheless, we show that the mixed page number of general directed acyclic 2-trees is unbounded. The results for these graph classes, upward planar graphs and directed acyclic 2-trees, are summarised in Table 1.1.

However, we do not only investigate the mixed page number of planar directed acyclic graphs, but also of sets of edges with a fixed ordering of their endpoints, called *edge patterns* (see Section 2.3). There, we determine two edge patterns which we call *t-crossing rainbow* and *t-nesting twist* that force the mixed page number to become large. Similar edge patterns, called *t-twist* and *t-rainbow*, are already known for the stack and queue number. Those patterns can then be recognised within a linear layout of a concrete graph. In this case, it follows that the mixed page number of the graph is at least the size of the recognised pattern with respect to the vertex ordering. This can be useful for arguing that the mixed page number of a graph is large, not only concerning planar DAGs but any graph. Further, we ask whether the mixed page number is bounded by a function in the size t of a largest t-crossing rainbow or t-nesting twist, as it is for the stack and queue number with t-twists and t-rainbows, respectively. Although this question remains open, we find an edge pattern that requires a number of pages quadratic in the size of a largest

[1] see [JMU22a]
[2] see Proposition 3.5
[3] see Corollary 3.9
[4] see Proposition 3.10
[5] Note that all edges of $G = (A \cup B, E)$ are oriented from a vertex in A to a vertex in B.
[6] see Proposition 4.6
[7] see Corollary 4.4
[8] see Proposition 4.2
[9] see Proposition 4.1
[10] see Proposition 4.5
[11] see [JMU22b]
[12] see Theorem 5.1
[13] see Theorem 5.4

t-crossing rainbow or t-nesting twist (see Proposition 2.15). It follows that such a function – if it exists – is at least a quadratic function.

1.5 Outline

Before we start investigating graphs, we define and explain notions that are used throughout this thesis. In Chapter 2, we introduce upward planar graphs and directed acyclic 2-trees and describe linear layouts more detailed.

In Chapter 3 we investigate some concrete families of upward planar graphs, and thereby give a lower bound on the mixed page number of upward planar graphs. Further, we observe that there are graphs with a mixed page number strictly smaller than the stack and queue number.

Then, in Chapter 4, we restrict the class of upward planar graphs and consider only those graphs which are additionally bipartite. There we find another subclass of upward planar bipartite graphs with bounded stack, queue, and mixed page number. In contrast to that, we show that the mixed page number is unbounded for upward planar bipartite graphs with fixed vertex ordering. Additionally, we again give a lower bound on the mixed page number of upward planar bipartite graphs and observe that it can be strictly smaller than the stack and queue number for some graphs.

Finally, in Chapter 5 we investigate another class of planar DAGs, namely directed acyclic 2-trees. In this chapter, we even find a 2-tree with unbounded stack and queue number but bounded mixed page number, and show that the mixed page number of directed acyclic 2-trees is unbounded.

2. Preliminaries

The following chapter provides some basic concepts and notations, which we will use throughout this thesis. In particular, we introduce the two main graph classes of this thesis, upward planar graphs and directed acyclic 2-trees, and define stack, queue and mixed page numbers.

If not stated otherwise, we assume graphs to be directed, finite, and simple, i.e., they contain neither loops nor multiple edges. We denote an edge e as $e = (u, v)$ if it is oriented from u to v. We only consider *directed acyclic graphs* (DAGs), which are directed graphs with no directed cycle $C = (v_1, \dots, v_k)$, $k \geq 2$, with $(v_i, v_i + 1) \in E$ for $i = 1, \dots, k-1$ and $(v_k, v_1) \in E$.

2.1 Upward Planar Graphs

To define upward planar graphs, we consider embeddings of graphs $G = (V, E)$ into the Euclidean plane $\mathbb{R}^2$. In such an embedding, vertices $v \in V$ are placed at pairwise distinct points in the Euclidean plane and edges $(u, v) \in E$ are represented by injective, continuous Jordan curves $f_{(u,v)} : [0, 1] \to \mathbb{R}^2$ with $f_{(u,v)}(0) = u$ and $f_{(u,v)}(1) = v$, where $f_{(u,v)}$ does not go through any other points on which another vertex lies. We call an embedding *planar* if no two edges cross in this embedding, i.e., $f_e(x) \neq f_{e'}(x')$ for any two edges $e, e' \in E$ and $0 < x, x' < 1$. If additionally $f_{(u,v)}$ is strictly y-monotone for every edge (u, v), i.e., all edges are oriented upwards, the embedding is called *upward planar*. Therefore, if no edge directions are drawn in figures of upward planar embeddings in this thesis, we assume all edges to go from bottom to top. A graph G is called (*upward*) *planar* if there exists such an (upward) planar embedding of G. Note that all upward planar graphs are DAGs, and in particular acyclic, since there is an embedding with all edges going from bottom to top.

Given an embedding of a planar graph, the connected components of $\mathbb{R}^2$ after removing all edges and vertices, are called *faces*. There is exactly one unbounded face, the area outside the whole graph, which is called the *outer face*. All other faces are called *inner faces*.

A graph $G = (V, E)$ is called *bipartite* if the set of vertices V can be partitioned into two subsets $V = A \dot{\cup} B$, such that there are no edges within the two subsets, i.e., for all edges $(u, v) \in E$ it is either $u \in A$ and $v \in B$ or $u \in B$ and $v \in A$. A graph that is both bipartite and upward planar is called an *upward planar bipartite graph*.

(i) Cyclic. (ii) Transitive. (iii) Monotone, incoming in x. (iv) Monotone, outgoing at x.

Figure 2.1: The four possible options to orient the edges between a child x and its parents u and v with the parent edge (u, v) when constructing a directed 2-tree.

2.2 Directed Acyclic 2-Trees

In the following section we introduce the graph class of directed acyclic 2-trees. Therefore, we first consider undirected 2-trees, whereby an undirected edge between the vertices u and v is denoted as $\{u, v\}$.

Definition 2.1. *An* undirected 2-tree *is defined inductively as follows:*

- *A single edge is a 2-tree. This first edge in the construction sequence of a 2-tree is called the* base edge.
- *If $G = (V, E)$ is a 2-tree and $\{a, b\} \in E$ is an edge in G, then $G' = (V', E')$ with $V' = V \cup \{x\}$ and $E' = E \cup \{\{a, x\}, \{b, x\}\}$ is a 2-tree. Then, we say x is* stacked *onto the edge (a, b).*

Note that there are various construction sequences for the same 2-tree. In fact, there is a construction sequence for each edge e, such that e is the base edge. Further, even with a fixed base edge, there are different construction sequences for the same 2-tree. However, if we fix a base edge, we determine uniquely for each vertex $x \in V$ onto which edge $\{u, v\} \in E$ it is stacked (apart from the endpoints of the base edge). Then, $\{u, v\}$ is called the *parent edge* of x, the vertices u and v are called the *parents* of x and x is called a *child* of u and v. In some cases, we distinguish the two parents u and v by calling u the *left parent* and v the *right parent* of x. A vertex y is called a *descendant* of the edge (u, v) or of the vertices u and v if y is a child of u and v or if y is a child of y' and y' is a descendant of (u, v). Conversely, if y is a descendant of (u, v), then u and v are called *ancestors* of y and (u, v) is called *ancestor edge*. The subgraph induced by an edge e and its descendants is again a 2-tree, called a *subtree*, with base edge e. Note that every construction sequence of a 2-tree G contains only one unique base edge of the graph G, but every edge is a base edge for the subtree induced by itself and its descendants. We now consider directed 2-trees with a fixed base edge. Then, there are four possibilities to orient the edges between a child x and its parents u and v with the parent edge (u, v) directed from u to v (see Figure 2.1):

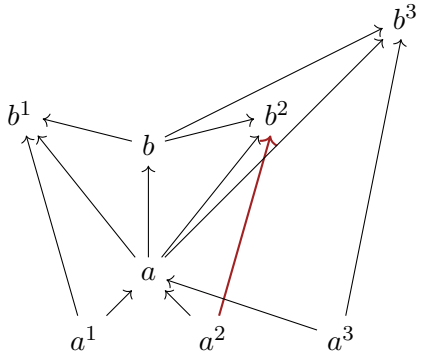
(i) If $(x, u), (v, x) \in E$, then (u, v, x) forms a directed cycle and x is called *cyclic*.

(ii) If $(u, x), (x, v) \in E$, x is called *transitive*.

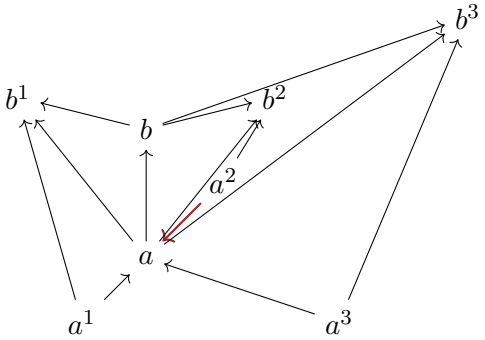
(iii) If $(u, x), (v, x) \in E$, then both edges are incoming in x and x is called *monotone*.

(iv) If $(x, u), (x, v) \in E$, then both edges are outgoing at x and x is also called *monotone*.

Since a cyclic vertex forms a directed cycle with its parent, a graph containing a cyclic vertex is not acyclic. Moreover, a directed 2-tree is acyclic, and hence a DAG, if and only if no vertex is cyclic, independent from the choice of the base edge. According to this, we define a *directed acyclic 2-tree* as a directed 2-tree, where every construction sequence only uses the options (ii), (iii) and (iv), and thus, where every vertex, except for the base edge's endpoints, is either transitive or monotone. We remark that every directed acyclic 2-tree is



(a) This embedding is not even planar.



(b) In this embedding not all edges are oriented in an upward way.

Figure 2.2: Two options to embed the same directed acyclic 2-tree, which is not upward planar.

also a planar graph, but not necessarily upward planar. Figure 2.2 illustrates a directed acyclic 2-tree G which is not upward planar. At least two of the three vertices b^1, b^2 and b^3 are on the same side of the base edge (a, b), here b^2 and b^3. Now, consider the vertex a^2 that is adjacent to b^2, which is here the vertex of the two that is closer to the base edge. There are only two faces of the subgraph $G - a^2$ (G without a^2 and all edges incident to a^2), into which a^2 can be embedded in a planar way: between (a, b) and (a, b^2) or between (a, b^2) and (a, b^3). In both cases, the edge (a^2, a) that is incoming in a is located between two edges outgoing from a. This is a contradiction to the bimodality property of upward planar embeddings that states that the sets of incoming and outgoing edges are separated from each other in the cyclic ordering of the edges at each vertex. Therefore, G cannot be embedded in an upward planar way.

Furthermore, a directed acyclic 2-tree is called *transitive* if there is a choice for the base edge (a, b), so that every vertex except a and b is transitive. Analogously, if there is a choice for the base edge (a, b), such that every vertex except a and b is monotone, we call the 2-tree *monotone*. In this case, we call x a *right child* of u and v, if the edges are directed from u and v to x as in (iii). Otherwise, if the edges are directed from x to its parents u and v as in (iv), we call x a *left child* of u and v. That is, because x is to the right or respectively left of its parents in any topological vertex ordering (see Section 2.3).

2.3 Linear Layouts

In this section we formalise the concepts of linear layouts, particularly, stack, queue, and mixed layouts. A *linear layout* of a graph consists of a total vertex ordering $\prec$ and a partition of the edges into sets, here into stacks and queues. We say, u is to the *left* of v and v is to the *right* of v if $u \prec v$. If there is no vertex ordering specified within a set of vertices A, we use the notations $u \prec A$ or $A \prec u$ if $u \prec v$ or $v \prec u$, respectively, for each $v \in A$. Additionally, for a directed graph we require that the vertex ordering is *topological*. That is, for every edge (u, v) oriented from u to v, it holds that $u \prec v$. Therefore, if no edge directions are drawn in figures of linear layouts of directed graphs in this thesis, we assume all edges to be oriented from left to right. Note that a directed graph G admits a topological vertex ordering if and only if G is acyclic, which is why we only consider directed acyclic graphs in this thesis.

(i) Two crossing edges. (ii) Two nesting edges.

Figure 2.3: Forbidden edge constellations in (a) stacks and (b) queues.

Definition 2.2. *Given a fixed vertex ordering $\prec$, a* stack *S is a set of edges, such that the endpoints of any two edges $(u_1, v_1), (u_2, v_2) \in S$ do not alternate with respect to $\prec$, i.e., it is neither $u_1 \prec u_2 \prec v_1 \prec v_2$ nor $u_2 \prec u_1 \prec v_2 \prec v_1$.*

Definition 2.3. *An s-*stack layout *of a graph G is a linear layout of G where the edges are partitioned into s stacks.*

Definition 2.4. *The* stack number $\mathrm{sn}(G)$ *of a graph G is the minimum number s, such that G admits an s-stack layout.*

Originally, stack layouts were referred to as *book embeddings*, where the vertices are imagined to lie orderly on the *spine* of a book and the edges are drawn on the *pages* of the book, such that no two edges on the same page cross [BK79]. A page is thereby a half-plane with the spine as boundary. As illustrated in Figure 2.3, two edges cross if and only if their endpoints alternate with respect to the vertex ordering. Thus, stacks and pages are equivalent. Therefore, in this thesis, we also say that two edges $(u_1, v_1), (u_2, v_2)$ *cross* with respect to the ordering $\prec$, if and only if $u_1 \prec u_2 \prec v_1 \prec v_2$ or $u_2 \prec u_1 \prec v_2 \prec v_1$. Apart from that, we prefer the terminologies associated with stacks as a counterpart to the queues, which we also consider in this thesis and define in the following.

Definition 2.5. *Given a fixed vertex ordering $\prec$, a* queue *Q is a set of edges such that for any two edges $(u_1, v_1), (u_2, v_2) \in Q$ it is neither $u_1 \prec u_2 \prec v_2 \prec v_1$ nor $u_2 \prec u_1 \prec v_1 \prec v_2$.*

Definition 2.6. *A q-*queue layout *of a graph G is a linear layout of G where the edges are partitioned into q queues.*

Definition 2.7. *The* queue number $\mathrm{qn}(G)$ *of a graph G is the minimum number q, such that G admits a q-queue layout.*

It is also possible to imagine a queue layout as a book embedding, where no two edges on the same page nest. Then, a page is equivalent to a queue (see Figure 2.3). Thus, it became increasingly reasonable to distinguish the two concepts by the different terms stack and queue layout with introduction of the queue number [HR92]. However, we also use the expression that two edges nest in this thesis. Precisely, we say that two edges $(u_1, v_1), (u_2, v_2)$ *nest*, if and only if $u_1 \prec u_2 \prec v_2 \prec v_1$ or $u_2 \prec u_1 \prec v_1 \prec v_2$. Moreover, the notions of books and pages are well suited to talk about mixed linear layouts, which we define below. In this thesis the term *page* denotes both, stacks and queues, if it is not exactly specified which of the two is meant.

Definition 2.8. *A q-*queue *s-*stack layout *is a linear layout of G where the edges are partitioned into q queues and s stacks.*

Definition 2.9. *The* mixed page number $\mathrm{mpn}(G)$ *of a graph G is the minimum number $s + q$ of stacks and queues, such that G admits a q-queue s-stack layout.*

(a) A t-twist. (b) A t-rainbow.

Figure 2.4: (a) requires at least t stacks. (b) requires at least t queues.

Note that $q = 0$ or $s = 0$ is possible, such that the mixed linear layout becomes a stack or queue layout, respectively. Therefore, a q-queue layout is also a q-queue 0-stack layout, and a s-stack layout is a 0-queue s-stack layout. It follows that $\mathrm{mpn}(G) \leq \mathrm{sn}(G), \mathrm{qn}(G)$ for each graph G. For $s, q \neq 0$, a q-queue s-stack layout is also simply described as a *mixed linear layout*.

Given a vertex ordering, we observe that t pairwise crossing edges, which we call a *t-twist* (see Figure 2.4), require at least t stacks in a pure stack layout, and t pairwise nesting edges, called t-rainbow (see Figure 2.4), require at least t queues in a queue layout. Moreover, a large t-twist (or t-rainbow) is indeed necessary for a large stack number (or queue number). As Davies showed, the stack number is bounded by a function of the size t of the largest twist [Dav22].

Theorem 2.10 (Davies, 2022 [Dav22])**.** *Given a vertex ordering with a largest twist of size t, the edges can be partitioned into*

$$2t \log_2(t) + 2t \log_2(\log_2(t)) + 10t$$

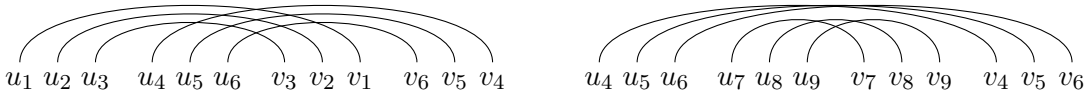
stacks.

As Heath and Rosenberg proved [HR92], for the queue number a large t-rainbow is indeed the only reason to be large, since the queue number is bounded by the size t of the largest rainbow.

Theorem 2.11 (Heath, Rosenberg, 1992 [HR92])**.** *Given a vertex ordering with a largest rainbow of size t, the edges can be partitioned into t queues.*

It follows that $\mathrm{sn}(G) \in \Theta(t \log(t))$ and $\mathrm{qn}(G) \in \Theta(t)$ for each graph G with a largest twist or respectively rainbow of size t in a linear layout. We call such an edge set E with the respective endpoints and a fixed vertex ordering $\prec$, like a t-twist or a t-rainbow, an *edge pattern* $P = (V, E, \prec)$. A q-queue s-stack layout of an edge pattern is a partition of the edges into q queues and s stacks with the same vertex ordering as in the pattern. The mixed page number $\mathrm{mpn}(P)$ of an edge pattern P is the minimum number of pages $q + s$ required for a q-queue s-stack layout of the pattern.

By combining t-twists and t-rainbows, we identify two edge patterns that cause the mixed page number to become large. To define these two patterns, we say that *two rainbows r_1 and r_2 cross*, when all edges from r_1 cross all edges from r_2 (see Figure 2.5). Analogously, we say that *two twists t_1 and t_2 nest*, when all edges from t_1 nest with all edges from t_2 (see Figure 2.5). Then, one of the two patterns with large mixed page number consists of t rainbows of the size t that cross with each other. We call such an edge pattern a *t-crossing rainbow* (see Figure 2.6). For a t-crossing rainbow at least t pages are required in a mixed linear layout, since any queue contains at most one edge of each of the t rainbows and any stack contains at most one whole rainbow, but no two edges from different rainbows. Thus, each page contains at most t edges, but there are t^2 edges in total, so at least t pages are



Two 3-rainbows that cross. Two 3-twists that nest.

Figure 2.5: The edge patterns t-twist and t-rainbow combined to crossing rainbows and nesting twists.

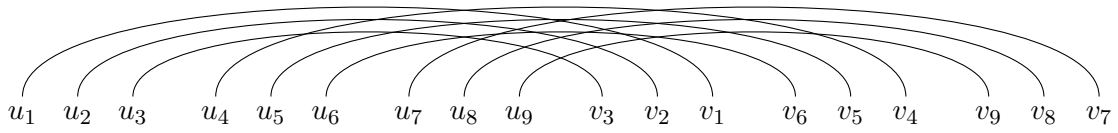


Figure 2.6: A 3-crossing rainbow, for which at least 3 pages are required.

required. Further, t pages are sufficient for a t-crossing rainbow as we use one stack per rainbow or put one edge of each rainbow into a queue. Hence, a t-crossing rainbow admits a t-stack layout and a t-queue layout.

On the other hand, for an edge pattern consisting of t twists of the size t that pairwise nest, called *t-nesting twist*, also at least t pages are required (see Figure 2.7), since any queue contains at most one whole twist, but no two edges from different twists, and any stack contains at most one edge of each of the t twists. Hence, there are at most t edges in each page, but t^2 edges in the t-nesting twist, so that in total t pages are necessary. Again, a t-nesting twist admits a t-stack layout and a t-queue layout, since we can put one edge of each twist into a stack or we can use one queue for each twist. As we show below, for t-crossing rainbows and for t-nesting twists, t pages suffice indeed only in a stack or queue layout, but not in a mixed linear layout, where we require at least one stack and one queue each.

Observation 2.12. *For each q-queue s-stack layout of a t-crossing rainbow or a t-nesting twist it holds that $q \geq t$ or $s \geq t$.*

Proof. First, consider a q-queue s-stack layout of a t-crossing rainbow with $q < t$. We show that $s \geq t$ follows. Each queue contains at most one edge per rainbow. Since the rainbows have size t and there are less than t queues, there is at least one edge left from each rainbow. The edges of different rainbows pairwise cross, and therefore the remaining edges form a t-twist, for which at least t stacks are required.

Now, let $s < t$, and we show that $q \geq t$ follows. Each stack contains at most one rainbow but no two edges from different rainbows. Since there are t rainbows but less than t stacks, one complete t-rainbow remains. For this, at least t queues are required.

Next, we consider a q-queue s-stack layout of a t-nesting twist with $q < t$. Similar to the previous cases, each queue contains at most one twist but no two edges from different twists,

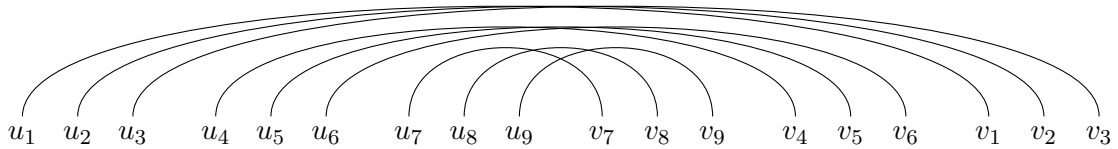


Figure 2.7: A 3-nesting twist, for which at least 3 pages are required.

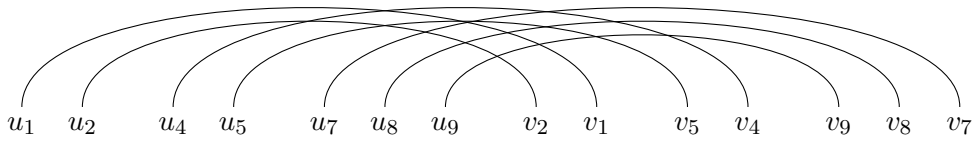


Figure 2.8: A 2-crossing rainbow, whose edges all cross every edge of a single 3-rainbow. For this edge pattern 3 pages are already required.

since they nest. Again, there are t twists but less than t queues, so that one complete t-twist remains, which requires at least t stacks.

Lastly, we assume $s < t$ and show that $q \geq t$ follows. Analogously to above, each stack contains at most one edge per twist. Since there are t twists but less than t stacks, at least one edge per twist remains. The edges of different twists pairwise nest, and thus form a t-rainbow altogether. Therefore, at least t queues are additionally necessary. □

It follows that even if we have $t-1$ queues, t additional stacks would be necessary for a t-crossing rainbow or a t-nesting twist, although t stacks alone are already sufficient. Thus, there is no reason to use the $t-1$ queues, or any queue at all. Conversely, using $t-1$ or less stacks is not reasonable, since then t queues are still required, even though t queues alone would already suffice. Therefore, whenever a t-crossing rainbow or a t-nesting twist occurs, we may assume its edges to be partitioned into t stacks or into t queues.

The question that now arises is, whether the mixed page number only depends on the size of the largest t-crossing rainbow or t-nesting twist, similarly as for the stack and queue number with t-rainbows and t-twists. One first observation that we make is that a slightly smaller edge subset of t-crossing rainbows or t-nesting twists already requires at least t stacks.

Observation 2.13. *A $(t-1)$-crossing rainbow, whose edges all cross every edge of a single t-rainbow, requires at least t pages.*

Proof. Consider such a $(t-1)$-crossing rainbow that crosses a single t-rainbow (see Figure 2.8). Assume that $t-1$ pages suffice. We observe that the pattern consists of $(t-1)(t-1)+t$ edges, namely $(t-1)(t-1)$ edges from the $(t-1)$-crossing rainbow and t edges from the additional t-rainbow. By pigeonhole principle, one of the $t-1$ pages contains at least

$$\left\lceil \frac{(t-1)(t-1)+t}{t-1} \right\rceil = \left\lceil t-1+\frac{t}{t-1} \right\rceil = t+1$$

edges. However, a stack contains at most one rainbow but no two edges from two distinct rainbows, as they cross each other. Since the largest rainbow has size t, there is no stack with $t+1$ edges. Similarly, a queue contains at most one edge per rainbow, and thus not more than t edges in total. This is a contradiction to the assumption that $t-1$ pages suffice. □

Similar arguments also apply to nesting twists, which leads to the following observation. The edge pattern considered in Observation 2.14 is illustrated in Figure 2.9.

Observation 2.14. *A $(t-1)$-nesting twist, whose edges all nest with every edge of a single t-twist, requires at least t pages.*

To get even closer to answering the question, we aim to find a function f in the size t of the largest t-crossing rainbow or t-nesting twist in a pattern P, such that the mixed

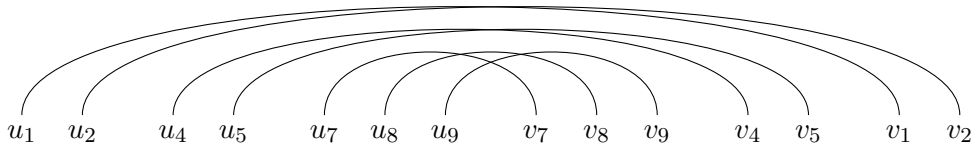


Figure 2.9: A 2-nesting twist, whose edges all nest with every edge of a single 3-twist. For this edge pattern 3 pages are already required.

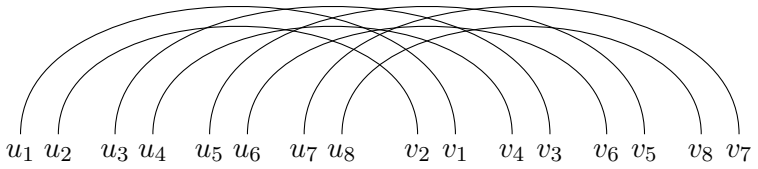


Figure 2.10: A 4-crossing 2-rainbow consisting of four 2-rainbows that pairwise cross.

page number is bounded by this function, i.e., $\mathrm{mpn}(P) \leq f(t)$. In the following, we give a lower bound on this function, if it exists, by proving that $f(t) \in \Omega(t^2)$. To do this, we first generalise the edge patterns t-crossing rainbow and t-nesting twist by allowing that the size t of the rainbows or twists may differ from the number of t-rainbows or t-twists in the edge pattern, respectively. Precisely, k pairwise crossing t-rainbows are called a *k-crossing t-rainbow* (see Figure 2.10) and analogously, k pairwise nesting t-twists are called a *k-nesting t-twist* (see Figure 2.11). We observe that for these two patterns $\min\{k, t\}$ pages still suffice. For instance, a k-crossing t-rainbow with $k < t$ admits a k-stack layout, where each stack contains one of the t-rainbows. Conversely, with $t < k$ the pattern admits a t-queue layout, where every queue contains one edge per t-rainbow. Analogous arguments also apply to k-nesting t-twists. For $k = t$ the two patterns are simply t-crossing rainbows or t-nesting twists, for which we know that t pages suffice. In particular, it follows that a k-crossing t-rainbow or a k-nesting t-twist does not contain a $(t+1)$-crossing rainbow or a $(t+1)$-nesting twist for each $k \in \mathbb{N}$ because then at least $t+1$ pages would be required. We now use these patterns to construct an even larger pattern that still contains neither a $(t+1)$-crossing rainbow nor a $(t+1)$-nesting twist, but requires at least t^2 pages.

Proposition 2.15. *For each $t \geq 1$ there is an edge pattern that contains neither a $(t+1)$-crossing rainbow nor a $(t+1)$-nesting twist, and requires at least t^2 pages.*

Proof. For $t = 1$, the proposition is true, since a single edge already requires one page. For $t \geq 2$, consider t pairwise crossing t^2-nesting t-twists (see Figure 2.12). We number the t^2-nesting t-twists, which we call *subpatterns*, from left to right and label these subpatterns nt_i for $i = 1, \ldots, t$. Before we prove that this pattern requires at least t^2 pages, we first show that it does neither contain a $(t+1)$-crossing rainbow nor a $(t+1)$-nesting twist. For

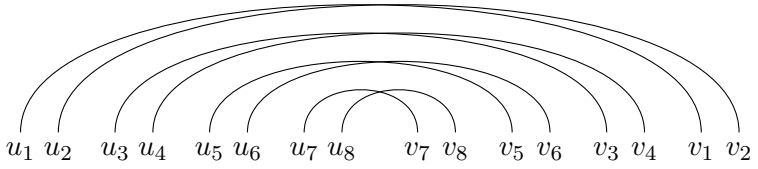


Figure 2.11: A 4-nesting 2-twist consisting of four 2-twists that pairwise nest.

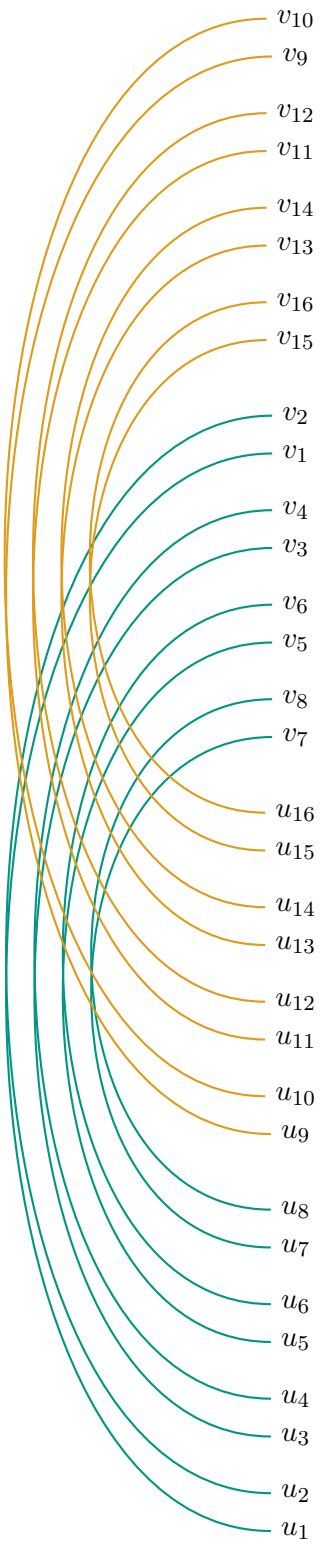


Figure 2.12: Two pairwise crossing 4-nesting 2-twists that contain neither a 3-crossing rainbow nor a 3-nesting twist but require at least 4 pages. The edge pattern can be partitioned into the two subpatterns nt_1 (green) and nt_2 (orange), which are the two 4-nesting 2-twists.

this, we first observe that two edges from different subpatterns $\mathrm{nt}_i, \mathrm{nt}_j$ with $i \neq j$ do not nest, since they all pairwise cross.

Now, we show that the pattern does not contain a $(t+1)$-crossing rainbow. To do this, we consider two disjoint rainbows r_1, r_2. Since edges from two different subpatterns do not nest, we may assume that r_1 and r_2 are each completely contained in one subpattern. Thus, let r_1 be contained in nt_i and r_2 in nt_j. If they are contained in the same subpattern, i.e., $i = j$, then they do not cross, since the subpatterns are nesting twists and no two rainbows cross within a nesting twist. Hence, only rainbows from different subpatterns $\mathrm{nt}_i, \mathrm{nt}_j$ with $i \neq j$ can cross. Since there are only t of these subpatterns nt_i $(i = 1, \ldots, t)$, there are at most t disjoint pairwise crossing rainbows. Thus, the pattern does not contain a $(t+1)$-crossing rainbow.

Next, we show that the pattern does not contain a $(t+1)$-nesting twist. A $(t+1)$-nesting twist can also be described as a $(t+1)$-rainbow, whose edges are each crossed by a t-twist that does not cross any other edges from the $(t+1)$-rainbow. Each edge from the $(t+1)$-rainbow forms a $(t+1)$-twist with the t-twist, by which it is crossed (see Figure 2.7). Therefore, we consider a $(t+1)$-rainbow r, and show that the edges in r are not crossed by t-twists in a way that they form a $(t+1)$-nesting rainbow. Let r be contained in the subpattern nt_i, and let e be one edge from the rainbow r. We now investigate the edges that cross e. Edges from a different subpattern that cross e also cross all other edges from r. Only edges from the same subpattern nt_i cross e and no other edges from r. Since nt_i is a t^2-nesting t-twist, it does not contain a twist of size larger than t. Therefore, there are only $t-1$ edges from the same subpattern as e that cross e and no other edges from r. It follows that r does not form a $(t+1)$-nesting twist with any t-twists.

Finally, we show that t pairwise crossing t^2-nesting t-twists require at least t^2 pages. First, we observe that each t-twist contains t edges. Thus, each t^2-nesting t-twist contains t^3 edges. Since there are t of them in our pattern, we have t^4 edges in total. Now, we assume that $t^2 - 1$ pages suffice. It follows after pigeonhole principle that one of those pages contains at least

$$\left\lceil \frac{t^4}{t^2 - 1} \right\rceil > t^2$$

edges. A stack contains only edges from one subpattern because the subpatterns pairwise cross. Furthermore, within a subpattern a stack contains only one edge per t-twist, and thus at most t^2 edges in total. On the other hand, a queue contains at most one t-twist from each subpattern, and hence also at most t^2 edges. This is a contradiction to the assumption that $t^2 - 1$ pages suffice. □

We remark that t^2 pages are indeed sufficient for this pattern, since it admits a t^2-stack layout. Here, we use t stacks for each of the t subpatterns, which are t^2-nesting t-twists. Each of these stacks then contains one edge per t-twist. Hence, it remains open if there are patterns that require more than t^2 pages with t the size of a largest t-crossing rainbow or t-nesting twist in the pattern. Moreover, it is still open whether the mixed page number even depends only on the size of a largest t-crossing rainbow or t-nesting twist.

3. Upward Planar Graphs

In the following chapter we observe a few s-stack q-queue layouts and thereby give some bounds on the stack, queue, and mixed page number of special upward planar graph families. For instance, cycles, grids, and N-grids are considered. Furthermore, we find a lower bound on the mixed page number of upward planar graphs and observe that the mixed page number of some graphs is strictly smaller than the stack and queue number. The results of this chapter are summarized in Table 3.1.

3.1 Cycles and Kelly graphs

Many graphs contain a cycle as a subgraph. Therefore, we first give a lower bound on the stack number for cycles, which we use in the following for other graphs.

Proposition 3.1. *For each cycle with an even number of vertices and with alternating incoming and outgoing edges at the vertices, at least two stacks are necessary.*

Proof. Let $C = [v_1, v_2, \ldots, v_n, v_1]$ be a cycle with an even number of vertices and alternating incoming and outgoing edges. Now, we try to find a topological ordering where all edges fit into one stack. Without loss of generality we assume that v_1 has two outgoing edges, i.e., v_2 and v_n each have two incoming edges, one of which comes from v_1. Thus, in any topological ordering $\prec$ of the vertices $v_1 \prec v_2$ and $v_1 \prec v_n$ holds. Due to symmetry, we can assume that $v_1 \prec v_2 \prec v_n$. As v_3v_2 and v_1v_n may not cross, the order is $v_1 \prec v_3 \prec v_2 \prec v_n$. Analogous arguments lead to the vertex ordering $v_1 \prec v_3 \prec \cdots \prec v_{n-1} \prec v_{n-2} \prec \cdots \prec v_2 \prec v_n$, where $v_{n-1}v_n$ and v_1v_2 cross (see Figure 3.1). □

	Stack Number	Queue Number	Mixed Page Number
Cycle	≥ 2		
Kelly Graph G_k	$= 2$	unbounded	$= 2$
Grid $Grid_{m,n}$	≥ 3	≤ 2	≤ 2
N-Grid $N_{m,n}$	≥ 5	≤ 4	≤ 4
$N_{m,n} + R_k$	≥ 5	unbounded	≤ 4
$N_{m,n} + G_k$	≥ 5	unbounded	≤ 4

Table 3.1: Lower and upper bounds on the stack, queue and mixed page number of the graphs considered in Chapter 3.

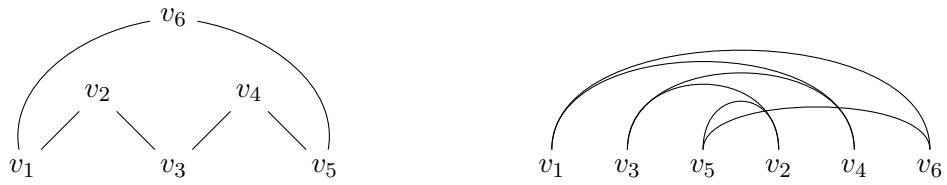


Figure 3.1: Left: A cycle with six vertices and alternating incoming and outgoing edges. Right: The same graph with the vertex ordering from the proof of Proposition 3.1

.

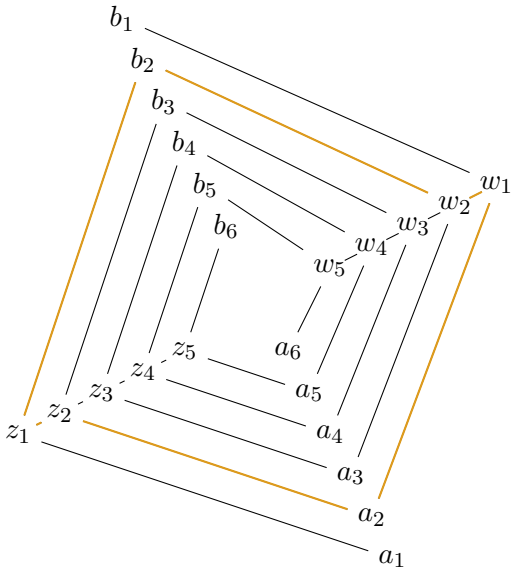


Figure 3.2: The graph G_6 containing a cycle with alternating incoming and outgoing edges (orange).

One example for graphs that contain such a cycle with alternating incoming and outgoing edges, are the graphs G_k for $k \geq 2$ following Kelly's construction [Kel81]. For each $k \geq 2$ the graph G_k consists of vertices $\{a_i \mid 1 \leq i \leq k\} \cup \{b_i \mid 1 \leq i \leq k\} \cup \{w_i \mid 1 \leq i \leq k-1\} \cup \{z_i \mid 1 \leq i \leq k-1\}$. The edges are partitioned into six subsets. Outgoing on the vertices a_i, we have the two edge sets $\{(a_i, w_{i-1}) \mid 2 \leq i \leq k\}$, called *a-w-edges*, and $\{(a_i, z_i) \mid 1 \leq i \leq k-1\}$, called *a-z-edges*. Analogously, incoming on the vertices b_i we have the *w-b-edges* $\{(w_i, b_i) \mid 1 \leq i \leq k-1\}$ and the *z-b-edges* $\{(z_i, b_{i+1}) \mid 1 \leq i \leq k-1\}$. Additionally, the vertices w_i and z_i are connected by *w-edges* $\{(w_i, w_{i-1}) \mid 2 \leq i \leq k-1\}$ and *z-edges* $\{(z_i, z_{i+1}) \mid 1 \leq i \leq k-2\}$ respectively. The graph G_6 is shown in Figure 3.2. Note that a Kelly graph G_k is a planar poset, as we mentioned in Section 1.3.

Since every Kelly graph contains such a cycle with alternating incoming and outgoing edges, for instance $[a_2, z_2, z_1, b_2, w_2, w_1, a_2]$ as illustrated in Figure 3.2, the corollary below follows.

Corollary 3.2. *For each G_k with $k \geq 3$ at least two stacks are necessary.*

Considering Kelly's construction further, we observe that 2 stacks are sufficient for each G_k ($k \geq 2$), i.e., the stack number equals 2 (for $k \geq 3$).

Proposition 3.3. *Each G_k with $k \geq 2$ admits a 2-stack-layout.*

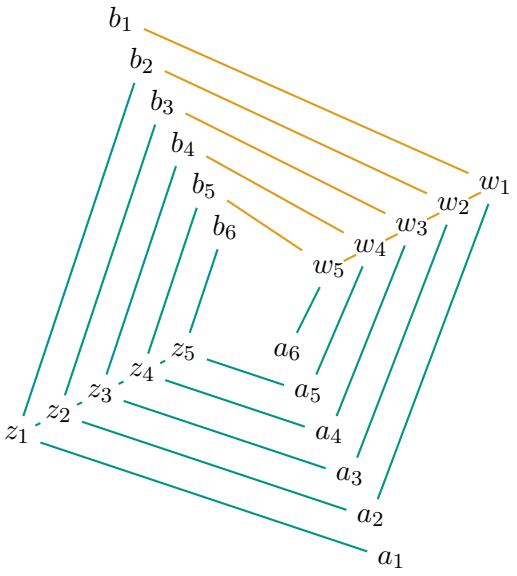


Figure 3.3: The graph G_6 with the edges partitioned into the set E_1 of a-w-edges, a-z-edges, z-b-edges and z-edges (green) and the set E_2 of w-b-edges and w-edges (orange).

Proof. Let $k \geq 2$ be an integer. We choose the vertex ordering $\prec$ with $a_i, w_i \prec b_j, z_j$ for each $1 \leq i, j \leq k$. Furthermore, we have $a_i \prec w_{i-1} \prec a_{i-1}$ $(i = 2, \ldots, k)$ and $b_i \prec z_i \prec b_{i+1}$ $(i = 1, \ldots, k-1)$. In total, we obtain the ordering

$$a_k \prec w_{k-1} \prec a_{k-1} \prec \cdots \prec a_2 \prec w_1 \prec a_1 \prec b_1 \prec z_1 \prec b_2 \prec \cdots \prec b_{k-1} \prec z_{k-1} \prec b_k$$

(see Figure 3.4). Now, we partition the edges into two sets, for which we prove that they satisfy the stack property. In the first edge set E_1 we have the a-w-edges, the a-z-edges, the z-b-edges, and the z-edges. The second edge set E_2 contains the remaining w-b-edges and the w-edges (see Figure 3.3).

For E_1 we observe that the endpoints of the a-w-edges (a_i, w_{i-1}) and the z-b-edges (z_i, b_{i+1}) are neighbours in the vertex ordering. Thus, they do not cross any other edges and do not need to be considered further. Hence, we only need to show that there are no crossings within the sets of a-z- and z-edges and that no a-z-edge crosses a z-edge. First, we observe that the a_i are in descending order and the z_i in ascending order. Thus, for two a-z-edges (a_i, z_i), (a_j, z_j) with $a_i \prec a_j$ $z_j \prec z_i$ holds, so no two a-z-edges cross. Analogously, for two z-edges (z_i, z_{i+1}), (z_j, z_{j+1}) with $z_i \prec z_j$ it follows that $z_i \prec z_{i+1} \preccurlyeq z_j \prec z_{j+1}$, so there are no crossings within the set of z-edges. Lastly, we consider an a-z-edge (a_i, z_i) and a z-edge (z_j, z_{j+1}). Since all vertices a_i precede all vertices z_j, we have $a_i \prec z_i, z_j, z_{j+1}$. The vertex z_i is not located between z_j and z_{j+1} and therefore the a-z-edges do not cross the z-edges.

Similarly, we show that there are no crossings within E_2 consisting of the w-b- and w-edges. We observe that the w_i are in descending order and the b_i are in ascending order and all vertices w_i precede all vertices b_i. With the same arguments as above it follows that there are no crossings within the sets of w-b- and w-edges and that no w-b-edge crosses a w-edge. □

Shifting all vertices a_i forward by one position in the ordering, we obtain a 1-queue 1-stack layout as illustrated in Figure 3.5. To conclude the consideration of Kelly graphs, we lastly show that their queue number is unbounded. For this, we use the following Erdős-Szekeres theorem [ES35].

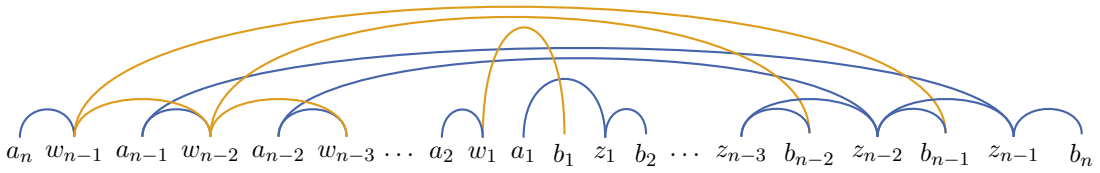


Figure 3.4: A 2-stack layout of G_k with the two stacks E_1 (blue) and E_2 (orange).

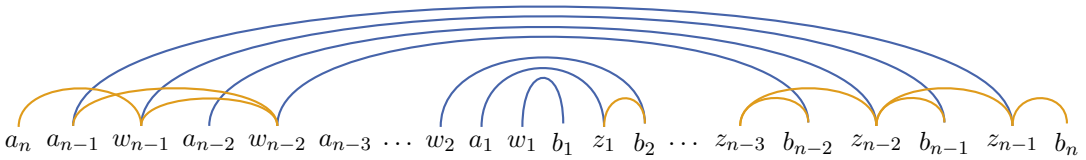


Figure 3.5: A 1-queue (orange) 1-stack (blue) layout of G_k.

Theorem 3.4 (Erdős, Szekeres, 1935 [ES35])**.** *Given $r, s \in \mathbb{N}$, any sequence of distinct real numbers of length at least $(r-1)(s-1)+1$ contains a monotonically increasing subsequence of length r or a monotonically decreasing subsequence of length s.*

Intuitively, a large sequence of numbers always contains a large monotonically increasing or a large monotonically decreasing subsequence.

Proposition 3.5. *For each $t \geq 1$ there is a Kelly graph G_k with $\mathrm{qn}(G_k) \geq t$.*

Proof. Let $t \geq 1$ be an integer. We choose $k = (t-1)^2+3$ and consider the Kelly graph G_k. Now, we show that every topological ordering of the vertices of G_k contains a t-rainbow, and thus requires at least t queues in a queue layout. For this, we consider a fixed topological ordering $\prec$. Due to the w- and z-edges, we know that $w_{i+1} \prec w_i$ and $z_i \prec z_{i+1}$ for each $i = 1, \ldots, k-2$, i.e., the w_i are in decreasing order of their indices, and the z_i in increasing order. However, the vertices b_i are allowed to be in any order. Let $b_{\sigma(1)} \prec b_{\sigma(2)} \prec \cdots \prec b_{\sigma(k)}$ be the order of the b_i for $i = 2, \ldots, k-1$, where $\sigma : \{1, \ldots, k-2\} \to \{2, \ldots, k-1\}$ is some permutation of the indices $i = 2, \ldots, k-1$. Thus, σ is a sequence of integers of size $k-2 = (t-1)^2+1$. By the Erdős-Szekeres theorem (see Theorem 3.4), it follows that σ contains a monotonically increasing subsequence of size t or a monotonically decreasing subsequence of size t. In the following, we investigate these two cases separately.

In the former case, let $\sigma' : \{1, \ldots, t\} \to \{2, \ldots, k-1\}$ be the monotonically increasing subsequence of size t, i.e., $\sigma'(i) < \sigma'(j)$ for each $1 \leq i < j \leq t$. Here, we consider the w-b-edges. Due to the decreasing order of the w_i with respect to their indices, i.e., $w_i \succ w_j$ for $1 \leq i < j \leq t$, and the edge $(w_{\sigma'(1)}, b_{\sigma'(1)})$, it follows that

$$w_{\sigma'(t)} \prec w_{\sigma'(t-1)} \prec \cdots \prec w_{\sigma'(1)} \prec b_{\sigma'(1)} \prec \cdots \prec b_{\sigma'(t-1)} \prec b_{\sigma'(t)}.$$

Hence, the t w-b-edges $(w_{\sigma'(i)}, b_{\sigma'(i)})$ for $i = 1, \ldots, t$ pairwise nest and form a t-rainbow.

In the latter case, let $\sigma' : \{1, \ldots, t\} \to \{2, \ldots, k-1\}$ be the monotonically decreasing subsequence of size t, i.e., $\sigma'(i) > \sigma'(j)$ for each $1 \leq i < j \leq t$. In this case, we consider the z-b-edges. Due to the increasing order of the z_i with respect to their indices, i.e., $z_i \prec z_j$ for $1 \prec i < j \prec t$, and the edge $(z_{\sigma'(1)-1}, b_{\sigma'(1)})$, it follows that

$$z_{\sigma'(t)-1} \prec z_{\sigma'(t-1)-1} \prec \cdots \prec z_{\sigma'(1)-1} \prec b_{\sigma'(1)} \prec \cdots \prec b_{\sigma'(t-1)} \prec b_{\sigma'(t)}.$$

Similar to the first case, the t z-b-edges $(z_{\sigma'(i)-1}, b_{\sigma'(i)})$ for $i = 1, \ldots, t$ pairwise nest and form a t-rainbow. Therefore, G_k requires at least t queues in a queue layout. □

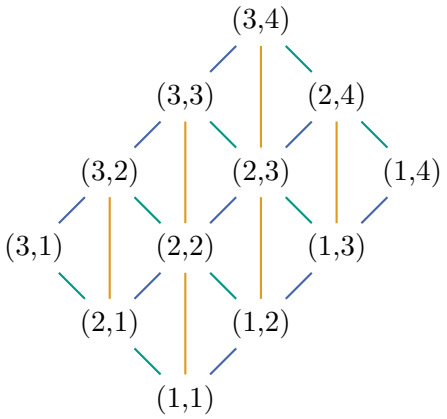


Figure 3.6: A 3×4 upward grid with left edges (green), right edges (blue) and vertical edges (orange).

3.2 Grids and N-Grids

In this section we investigate the mixed page number for grids and N-grids, following the definitions of Jungeblut, Merker, and Ueckerdt [JMU22a].

Let $m, n > 0$ be integers. An $m \times n$ *upward grid* $\text{Grid}_{m,n}$ consists of vertices (l, r) and an edge set which can be partitioned in three subsets. Edges of the form $((l, r), (l + 1, r))$ for each $1 \leq l \leq m - 1$ and $1 \leq r \leq n$ are called *left edges*. Analogously, we have *right edges* $((l, r), (l, r + 1))$ for each $1 \leq l \leq m$ and $1 \leq r \leq n - 1$. The third edge subset is the set of *vertical edges* $((l, r), (l + 1, r + 1))$ for each $1 \leq l \leq m - 1$ and $1 \leq r \leq n - 1$ (see Figure 3.6).

Adding an additional vertex, called *N-vertex*, in each inner face of $\text{Grid}_{m,n}$ and for each N-vertex edges to all three vertices incident to the respective face, we obtain an $m \times n$ *N-grid* $\text{N}_{m,n}$ as illustrated in Figure 3.7. Specifically, for every two triangles sharing a vertical edge and consisting of the vertices (l, r), $(l + 1, r)$, $(l, r + 1)$, and $(l + 1, r + 1)$, we insert *a-vertices* $a_{l,r}$ to the left triangle and *b-vertices* $b_{l,r}$ to the right triangle, if $l - r$ is even, and *c-vertices* $c_{l,r}$ to the left and *d-vertices* $d_{l,r}$ to the right triangle, if $l - r$ is odd. Then, we insert the *N-edges* of the form $(a_{l,r}, (l + 1, r + 1))$ and $((l, r), b_{l,r})$ or $(c_{l,r}, (l + 1, r + 1))$ and $((l, r), d_{l,r})$ respectively. Finally, we add the remaining *non-N-edges* to the other two vertices in each face $((l, r), a_{l,r})$, $(a_{l,r}, (l+1, r))$, $((l, r+1), b_{l,r})$ and $(b_{l,r}, (l+1, r+1))$ in the first case or $((l, r), c_{l,r})$, $(c_{l,r}, (l, r+1))$, $((l+1, r), d_{l,r})$ and $(d_{l,r}, (l+1, r+1))$ in the second case. The vertices and edges of $\text{Grid}_{m,n}$ are called *grid vertices* and *grid edges* respectively.

The best known lower bound on the stack number of upward planar graphs is 5, provided by Jungeblut, Merker, and Ueckerdt with an N-grid [JMU22a]. However, we reduce the number of pages needed for N-grids by admitting queues. Before we get to the N-grids, we first observe the grids, as they are subgraphs of the N-grids.

Proposition 3.6. *For each $m \times n$ upward grid with $m \geq 2$ and $n \geq 4$, at least three stacks are necessary.*

Proof. Let $\text{Grid}_{2,4}$ be a 2×4 upward grid with vertices (l, r) for each $1 \leq l \leq 2$ and $1 \leq r \leq 4$. Now, we try to find a topological ordering $\prec$, where all edges fit into two stacks. In any topological ordering we have $(1, 1) \prec (1, 2) \prec (2, 2) \prec (2, 3)$ and $(1, 2) \prec (1, 3) \prec (2, 3) \prec (2, 4)$ with the edges $((1, 1), (2, 2))$, $((1, 2), (2, 3))$ and $(1, 3), (2, 4))$, which require at least two stacks. If we additionally claim that $(1, 3) \prec (2, 2)$, we get $(1, 1,) \prec (1, 3) \prec (2, 2) \prec (2, 4)$. Thus, no two of these three edges can share the same stack and at least three stacks are necessary.

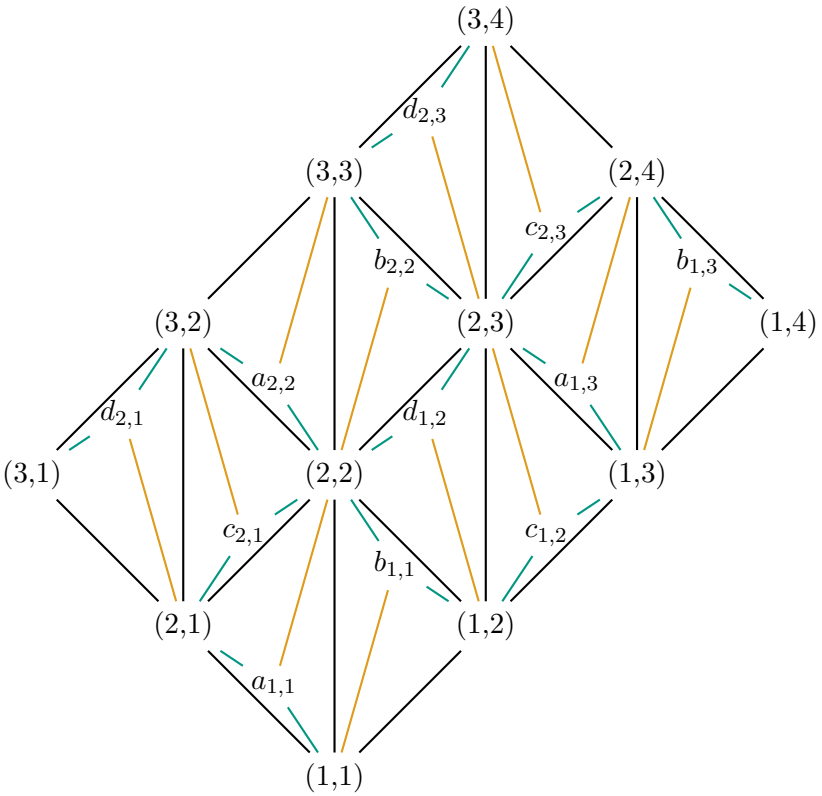


Figure 3.7: A 3×4 N-grid consisting of a 3×4 upward grid (black), N-edges (orange) and non-N-edges (green).

Now, let $(2,2) \prec (1,3)$ and thereby $(1,1) \prec (1,2) \prec (2,2) \prec (1,3) \prec (2,3) \prec (2,4)$. If $((1,2),(2,3))$ and $((2,2),(2,3))$ are in two different stacks, we need a third stack for $((1,3),(2,4))$, which crosses both of the other edges in the given ordering. On the other hand, if we have $((1,2),(2,3))$ and $((2,2),(2,3))$ in the same stack, a second stack for $((1,2),(1,3))$ crossing $((2,2),(2,3))$ is required but then a third stack is necessary for $((1,1),(2,2))$, which crosses $((1,2),(1,3))$ as well as $((1,2),(2,3))$.

Note that $\text{Grid}_{2,4}$ is a subgraph of all $m \times n$ upward grids with $m \geq 2$ and $n \geq 4$, which leads to the statement of Proposition 3.6. □

We remark that a 1×4 and a 2×3 upward grid admits a 2-stack layout, which makes $\text{Grid}_{2,4}$ a minimal counterexample.

Having a lower bound on the stack number of grids, we improve this by admitting queues and give an upper bound on the mixed page number of grids.

Proposition 3.7. *Each $m \times n$ upward grid admits a 2-queue layout and a 1-queue 1-stack layout.*

Proof. Let $\text{Grid}_{m,n}$ be an $m \times n$ upward grid with vertices (l,r) for each $1 \leq l \leq m$ and $1 \leq r \leq n$. Consider the vertex ordering $\prec$ where $v = (l_v, r_v) \prec (l_w, r_w) = w$ if and only if $l_v < l_w$ or $l_v = l_w$ and $r_v < r_w$ as shown in Figure 3.8. The first queue contains the left edges and the vertical edges, the second queue or stack contains the remaining right edges. First, we prove that the set of left and vertical edges actually is a queue, before we show the stack and queue properties for the set of right edges.

Let $(v,v') = ((l_v,r_v),(l'_v,r'_v)), (w,w') = ((l_w,r_w),(l'_w,r'_w))$ be two arbitrarily chosen edges from the set of left and vertical edges with $v \prec w$. We distinguish the two cases $l_v < l_w$

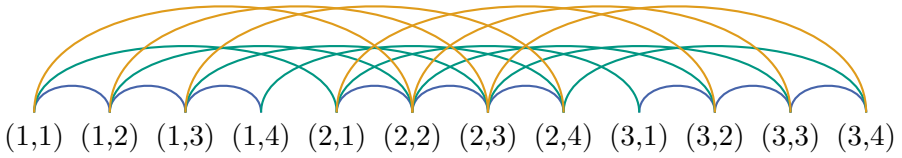


Figure 3.8: The 2-queue or 1-queue 1-stack layout of $\text{Grid}_{3,4}$. The set of left and vertical edges (green and orange) satisfies the queue property, the set of right edges (blue) the queue property as well as the stack property.

and $l_v = l_w$. In the first case, considering only left and vertical edges, we have $l'_v = l_v + 1 < l_w + 1 = l'_w$, which leads to $v' \prec w'$. The latter case $l_v = l_w$ is equivalent to $r_v < r_w$ and therefore $r'_v \leq r_v + 1 \leq r_w \leq r'_w$. With $l'_v = l_v + 1 = l_w + 1 = l'_w$, we get $v' \preccurlyeq w'$. Hence, in both cases the queue property is satisfied.

For the right edges, we observe that the endpoints of each edge $((l, r), (l, r + 1))$ are neighbours in the vertex ordering. Thus, the stack and queue properties are satisfied for this set of edges. □

It follows that $\text{qn}(\text{Grid}_{m,n}) \leq 2$ and $\text{mpn}(\text{Grid}_{a,b}) \leq 2$. Having a stack number $\text{sn}(\text{Grid}_{m,n}) \geq 3$ (see Proposition 3.6), the queue number and the mixed page number are strictly smaller than the stack number. This also applies to the N-grids as shown below.

Proposition 3.8. *Each $m \times n$ N-grid admits a 4-queue layout and a 3-queue 1-stack layout.*

Proof. Let $\text{N}_{m,n}$ be an $m \times n$ N-grid. For the grid vertices, we choose the ordering $\prec$ where $v = (l_v, r_v) \prec (l_w, r_w) = w$ if and only if $l_v < l_w$ or $l_v = l_w$ and $r_v < r_w$ as in the proof of Proposition 3.7. This ordering of the grid vertices is partitioned into m blocks of n vertices, where block l contains all vertices (l, r) for $r = 1, \ldots, n$. Then, we place the N-vertices in between as follows. Vertices $c_{l,r}$ and $d_{l,r}$ are as far left as possible, namely $c_{l,r}$ directly after (l, r) and $d_{l,r}$ directly after $(l + 1, r)$. For each $1 \leq l < m$ we place the N-vertices $a_{l,r}$ and $b_{l,r}$ between the block of vertices (l, r) and $(l + 1, r)$ for $r = 1, \ldots, n$, precisely $(l, n) \prec a_{l,r}, b_{l,r} \prec (l + 1, 1)$ for each $1 \leq r \leq n$, so that we obtain blocks of grid vertices with c- and d-vertices inside and blocks of a- and b-vertices alternatingly. Furthermore, we have the ordering $a_{l,r} \prec b_{l,r} \prec a_{l,r'} \prec b_{l,r'}$ if and only if $r < r'$. Overall, we obtain the ordering

$$(1,1)(1,2)c_{1,2}(1,3)(1,4)c_{1,4}\ldots(1,n)a_{1,1}b_{1,1}a_{1,3}b_{1,3}\ldots$$

$$(2,1)c_{2,1}(2,2)d_{1,2}(2,3)c_{2,3}(2,4)d_{1,4}\ldots a_{2,2}b_{2,2}a_{2,4}b_{2,4}\ldots$$

(see Figure 3.18). Having the same vertex ordering for the grid vertices, we can reuse the 2-queue layout or the 1-queue 1-stack layout for the induced subgraph $\text{Grid}_{m,n}$ as shown in Proposition 3.7 and thus only need to prove that two additional queues are sufficient. For this, we partition the edges of $\text{N}_{m,n}$ into four sets for which we show that they satisfy the queue property. The first set E_1 is the set of left and vertical edges from $\text{Grid}_{m,n}$, which satisfies the queue property as shown before (see Figure 3.10). The right edges share the edge set E_2 with $E'_2 = \{((l, r), c_{l,r}), ((l + 1, r), d_{l,r}), (c_{l,r}, (l, r + 1)), (d_{l,r}, (l + 1, r + 1)) \mid 1 \leq l \leq m - 1, 1 \leq r \leq n - 1\}$, the non-N-edges incident to the c- and d-vertices, whereby E_2 also satisfies the stack property as is shown in the following. The third edge set E_3 contains the edges incident to the a- and b-vertices and E_4 contains the remaining N-edges incident to the c- and d-vertices. An N-grid with the vertex ordering and the four edge sets highlighted is illustrated in Figure 3.9.

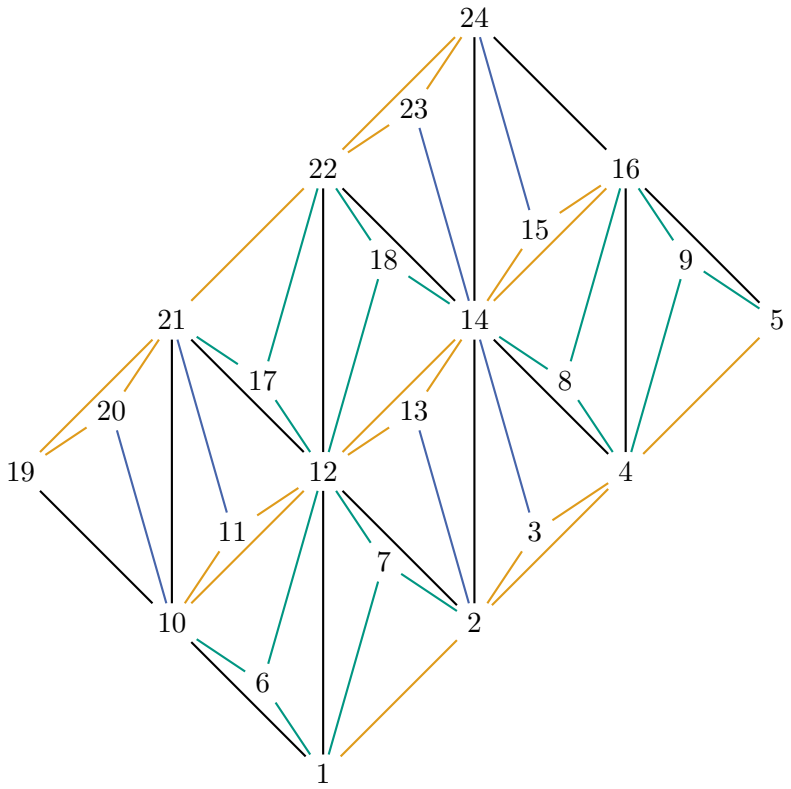


Figure 3.9: A 3×4 N-grid with the edges partitioned into the set E_1 of left and vertical edges from $\text{Grid}_{m,n}$ (black), the set E_2 of right edges with the non-N-edges incident to $c_{l,r}$ and $d_{l,r}$ (orange), the set E_3 of N-edges incident to $c_{l,r}$ and $d_{l,r}$ (blue), and the set E_4 of edges incident to $a_{l,r}$ and $b_{l,r}$ (green). The vertices are numbered as they appear in the 4-queue or 3-queue 1-stack layout in the proof of Proposition 3.8

.

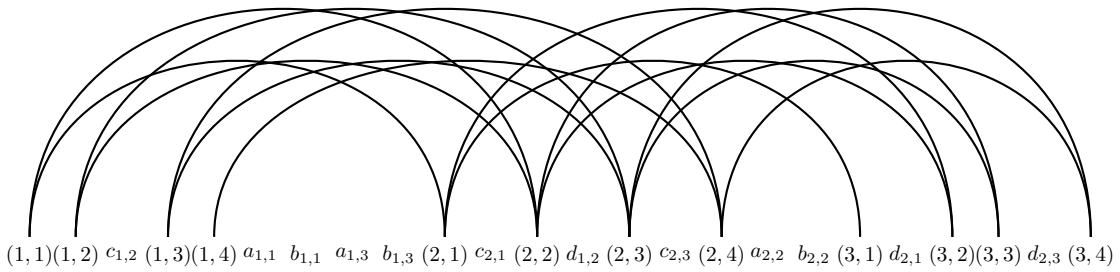


Figure 3.10: The set E_1 of left and vertical edges from $\text{Grid}_{3,4}$ in the 4-queue or 3-queue 1-stack layout of $\text{N}_{3,4}$, which forms a queue.

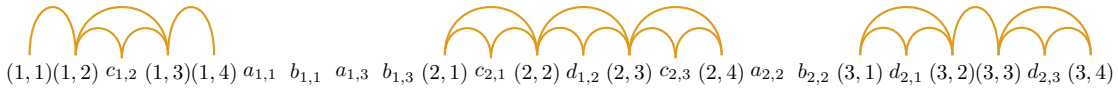


Figure 3.11: The set E_2 of right edges from $\text{Grid}_{3,4}$ and non-N-edges incident to the c- and d-vertices in the 4-queue or 3-queue 1-stack layout of $\text{N}_{3,4}$, which forms a queue or a stack.

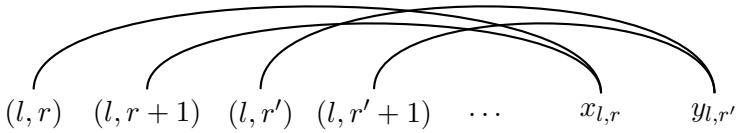


Figure 3.12: The incoming edges at the a- or b-vertices $x_{l,r}$ and $y_{l,r'}$ in the case that $r < r'$.

First, we observe that the endpoints of each edge from E_2' are neighbours in the vertex ordering, so they do not nest with each other. The right edges of $\text{Grid}_{m,n}$ connect two consecutive grid vertices, between which there is at most one N-vertex with the edges from E_2' to its two neighbours. Thus, these edges do not nest or cross with the right edges and the stack and queue properties are satisfied (see Figure 3.11).

Now, we consider the vertices $a_{l,r}$ and $b_{l,r}$ and the edge set E_3 with their incident edges as illustrated in Figure 3.16. We observe that all of these edges are between a block of N-vertices and the preceding or succeeding block of grid vertices. Therefore, we only need to consider edges from the same block to the succeeding block. Let $x_{l,r} \prec y_{l',r'}$ with $x, y \in \{a, b\}$ be two N-vertices from the same block, so $l = l'$ and $r \leq r'$. We distinguish the two cases that the edges are incoming or outgoing at the two vertices. In the first case we have at most the edges $((l, r), x_{l,r})$, $((l, r+1), x_{l,r})$, $((l, r'), y_{l,r'})$, and $((l, r'+1), y_{l,r'})$. If we assume $r < r'$, we obtain $(l, r) \prec (l, r+1) \preccurlyeq (l, r') \prec (l, r'+1)$ (see Figure 3.12), otherwise we have $x = a$ and $y = b$ and the edge $((l, r+1), x_{l,r})$ does not exist, so that the actually existing edges satisfy the queue property with $(l, r) = (l, r') \prec (l, r'+1)$ (see Figure 3.13). In the second case we consider the edges $(x_{l,r}, (l+1, r))$, $(x_{l,r}, (l+1, r+1))$, $(y_{l,r}, (l+1, r'))$, and $(y_{l,r}, (l+1, r'+1))$. Again, we observe that the queue property holds for $r < r'$ and $r = r'$ separately. For $r < r'$ we have $(l+1, r) \prec (l+1, r+1) \preccurlyeq (l+r, r') \prec (l+1, r'+1)$ (see Figure 3.14), and for $r = r'$ the edge $(y_{l,r'}, (l+1, r'))$ does not exist and we have $(l+1, r) \prec (l+1, r+1) = (l+1, r'+1)$ for the relevant endpoints (see Figure 3.15).

Lastly, we consider E_4, the set of N-edges at the c- and d-vertices as shown in Figure 3.17. The N-edges within one square of $\text{Grid}_{m,n}$ consisting of two triangles that share a vertical edge do not nest. Within one such square there are the N-edges $((l, r), d_{l,r})$ and $(c_{l,r}, (l+1, r+1))$ with $(l, r) \prec c_{l,r} \prec (l+1, r) \prec d_{l,r} \prec (l+1, r+1)$. The N-edges of different such squares satisfy the queue property as well because all the involved vertices of one square are separated from all of the vertices of the other square in the vertex ordering. □

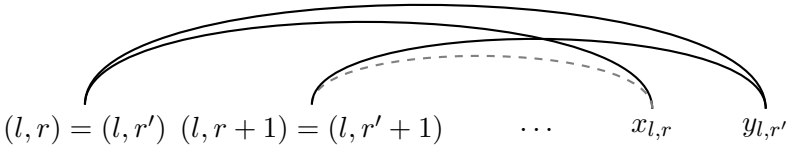


Figure 3.13: The incoming edges at the a- or b-vertices $x_{l,r}$ and $y_{l,r'}$ in the case that $r = r'$.

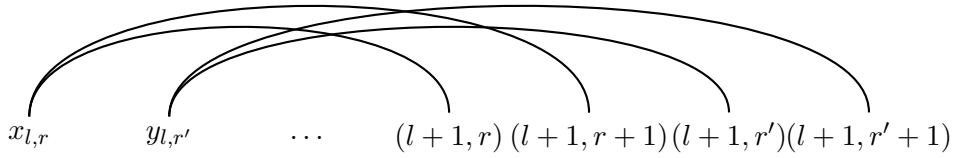


Figure 3.14: The outgoing edges at the a- or b-vertices $x_{l,r}$ and $y_{l,r'}$ in the case that $r < r'$.

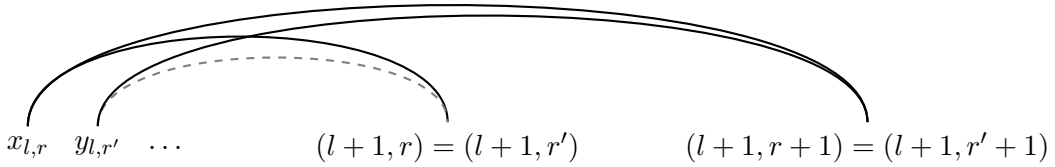


Figure 3.15: The outgoing edges at the a- or b-vertices $x_{l,r}$ and $y_{l,r'}$ in the case that $r = r'$.

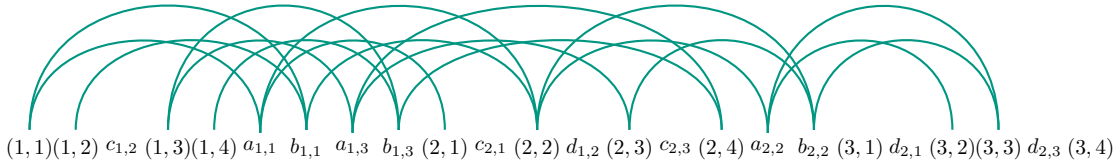


Figure 3.16: The set E_3 of edges incident to the a- and b-vertices, in the 4-queue or 3-queue 1-stack layout of $\mathrm{N}_{3,4}$, which forms a queue.

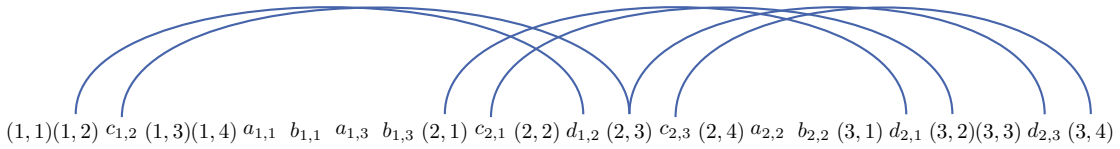


Figure 3.17: The set E_4 of N-edges incident to the c- and d-vertices in the 4-queue or 3-queue 1-stack layout of $\mathrm{N}_{3,4}$, which forms a queue.

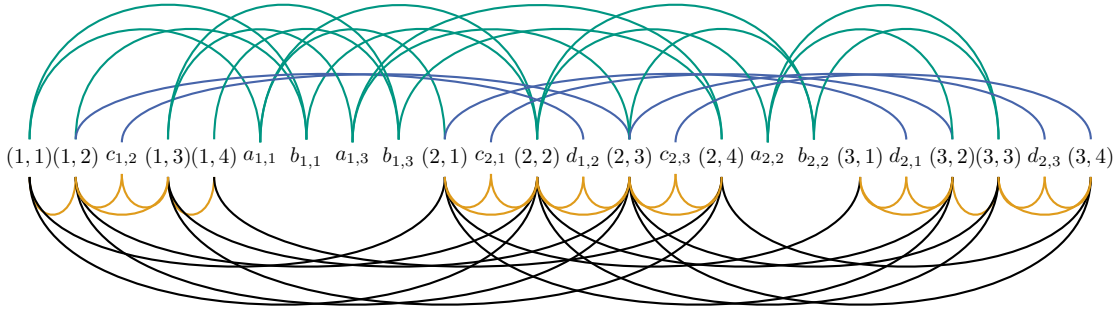


Figure 3.18: The complete 4-queue or 3-queue 1-stack layout of $\mathrm{N}_{3,4}$ consisting of the queues E_1 (black), E_2 (orange), E_3 (blue) and E_4 (green), whereby E_2 also satisfies the stack property.

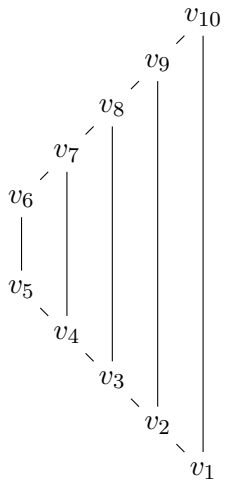


Figure 3.19: The rainbow graph R_{10}.

This leads to the upper bounds $\mathrm{qn}(\mathrm{N}_{m,n}) \leq 4$ and $\mathrm{mpn}(\mathrm{N}_{m,n}) \leq 4$ for the queue number and for the mixed page number. Note that the stack number of each sufficiently large N-grid is at least 5 [JMU22a], which is why the queue number and the mixed page number are strictly smaller than the stack number here. Since the queue number is unbounded for upward planar graphs, we can construct an upward planar graph G with $\mathrm{mpn}(G) < \mathrm{sn}(G), \mathrm{qn}(G)$ by combining an N-grid with an upward planar graph with high queue number but small stack number.

Corollary 3.9. *There is an upward planar graph with* $\mathrm{mpn}(G) < \mathrm{sn}(G), \mathrm{qn}(G)$.

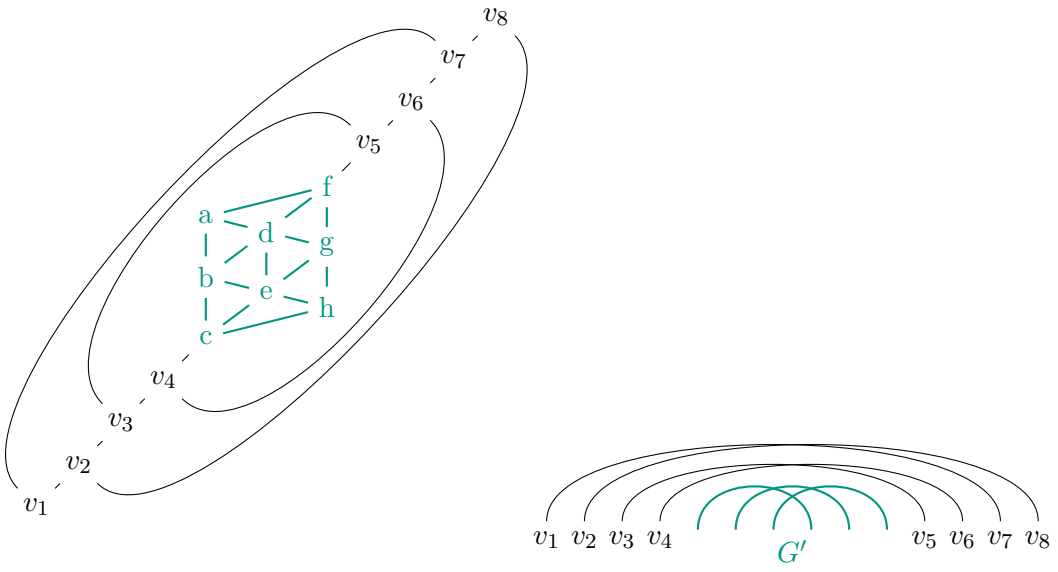
Proof. Consider a *rainbow graph* R_k consisting of the path $(v_1, v_2, \ldots, v_k)$ with the additional non-path-edges $\{(v_i, v_{k-i+1}) \mid 1 \leq i \leq \lfloor \frac{k}{2} \rfloor\}$ (see Figure 3.19). The topological ordering is clearly given by the path. In this ordering the edges do not cross but all of the non-path-edges nest with each other. Thus, the stack number of R_k equals 1, while the queue number is at least $\lfloor \frac{k}{2} \rfloor$. Therefore, $G = N_{m,n} + R_k$ has $\mathrm{sn}(G) \geq 5$ and a queue number linear in the number of edges. However, we still have $\mathrm{mpn}(G) \leq 4$. □

The rainbow graph now also serves as an example for an upward planar graph with unbounded queue number. Instead of the rainbow graph R_k, we could have also chosen a Kelly graph G_k, for which we also know that the queue number is unbounded according to Proposition 3.5. Note that R_k is additionally bipartite for an even k, since then all edges are between a vertex with an even label and a vertex with an odd label. It follows that the queue number of upward planar bipartite graphs is also unbounded.

Also starting from the grids, we now give a lower bound on the mixed page number of upward planar graphs by constructing a graph G with $\mathrm{mpn}(G) \geq 3$ that contains a subgraph of $\mathrm{Grid}_{m,n}$.

Proposition 3.10. *There is an upward planar graph* G *with* $\mathrm{mpn}(G) \geq 3$.

Proof. The graph G with $\mathrm{mpn}(G) \geq 3$ considered here is illustrated in Figure 3.20. It consists of a path $(v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8)$, whereby v_4 and v_5 are not connected by an edge but by the graph G' (green). Note that G' is a subgraph of a grid. Additionally, there are the edges (v_1, v_7), (v_2, v_8), (v_3, v_5), and (v_4, v_6) parallel to the path, called *parallel edges*. For G' it is known that it contains a 3-twist in each possible topological ordering [JMU22a]. The ordering of the path is unique, whereby the parallel edges form two 2-twists. These



An upward planar graph G with $\mathrm{mpn}(G) \geq 3$.

A linear layout of G with only the relevant edges.

Figure 3.20: The subgraph G' (green) contains a 3-twist in each topological ordering that nests with the two other nesting 2-twists.

2-twists and the 3-twist from G' nest with each other (see Figure 3.20) and we obtain a pattern as already investigated in Observation 2.14, for which we know that it requires at least three pages. More detailed, we only consider the seven edges of the pattern consisting of 2- and 3-twists in the following. Assume that two pages are sufficient. By pigeonhole principle, at least four of these seven edges have to be on the same page. However, in a stack there is at most one edge per twist, i.e., not more than three edges in total. Similarly, in a queue there is at most one complete twist and no two edges from different twists as they are nesting. Thus, there are also at most three edges in one queue and at most six edges in two pages. □

4. Upward Planar Bipartite Graphs

After having considered very specific graphs in the previous chapter, we now investigate the mixed page number of the larger class of upward planar bipartite graphs. First, we give upper bounds on the mixed page number for upward planar bipartite graphs with a fixed vertex ordering and with all edges directed from one vertex set to the other. Then, we show that the lower bound of 3, that we found for general upward planar graphs in Chapter 3, also holds for upward planar bipartite graphs, and that there are upward planar bipartite graphs with a mixed page number strictly smaller than the stack and queue number.

4.1 Upper Bound

In this section, we focus on upward planar bipartite graphs with some further constraints and find upper bounds on the mixed page number in these special cases. One approach to find bounds on the mixed page number is to fix a vertex ordering, for instance "from left to right", and try to use this ordering for all graphs. However, this leads to an unbounded number of pages as we show below.

Proposition 4.1. *For each $t \geq 0$ there is an upward planar bipartite graph G_t and a fixed topological ordering $\prec$, so that at least t pages are necessary.*

Proof. We construct an upward planar bipartite graph $G = (V, E)$ that contains a t-nesting twist in the fixed topological ordering of the vertices, which we also define. The graph G consists of vertices $V = \{a_{i,j} \mid 1 \leq i, j \leq t\} \cup \{b_{i,j} \mid 1 \leq i, j \leq t\}$ and edges $E = \{(a_{i,j}, b_{i,j}) \mid 1 \leq i, j \leq t\}$. The fixed vertex ordering is defined as follows: For each $1 \leq i, j, i', j' \leq t$ it holds that $a_{i,j} \prec b_{i',j'}$, i.e., all $a_{i,j}$ are to the left of the $b_{i',j'}$. Further, it is $a_{i,j} \prec a_{i',j'}$ if and only if $i < i'$ or $i = i'$ and $j < j'$, so the $a_{i,j}$ are sorted lexicographically in ascending order by their vertex label. For the $b_{i,j}$ the ordering is $b_{i,j} \prec b_{i',j'}$ if and only if $i > i'$ or $i = i'$ and $j < j'$. In total, we obtain the ordering

$$a_{1,1} \prec a_{1,2} \prec \cdots \prec a_{2,1} \prec \cdots \prec a_{t,t} \prec b_{t,1} \prec b_{t,2} \prec \cdots \prec b_{2,t} \prec \cdots \prec b_{1,t-1} \prec b_{1,t}.$$

Note that there is an upward planar embedding of G, where all vertices are arranged in the ordering $\prec$ from bottom left to top right (see Figure 4.1), which makes $\prec$ an ordering "from left to right". G consists of $2 \times t$ *groups* of t vertices each, whereby all vertices $A_i = \{a_{i,j} \mid 1 \leq j \leq t\}$ with fixed i form a group, and analogously, all vertices $B_i = \{b_{i,j} \mid 1 \leq j \leq t\}$ with fixed i form a group. The edges go from group A_i to group B_i, whereby the ordering of groups B_i is the reverse of that of the groups A_i. Precisely, if $i < i'$

it holds for each $b_{i,j} \in B_i, b_{i',j'} \in B_{i'}$ that $b_{i',j'} \prec b_{i,j}$, while for $a_{i,j} \in A_i, a_{i',j'} \in A_{i'}$ it holds that $a_{i,j} \prec a_{i',j'}$. Due to this reverse ordering, the edges of different groups pairwise nest. Within the groups A_i, B_i, edges are between $a_{i,j}$ and $b_{i,j}$, whereby the vertex ordering is the same within A_i and B_i, i.e., if $j < j'$, it holds that $a_{i,j} \prec a_{i,j'}$ and $b_{i,j} \prec b_{i,j'}$. Thus, it follows that the edges of the same group pairwise cross. Since these crossing edges pairwise nest with the crossing edges of the other groups, we obtain a t-nesting twist, which requires at least t pages.

□

However, the mixed page number is bounded for the subclass of upward planar bipartite graphs, where all edges are oriented from one of the two vertex sets to the other. For this subclass, we observe that it is always a valid topological ordering to put all vertices of one set separated from all the vertices of the other set.

Proposition 4.2. *Each upward planar bipartite graph $G = (A \cup B, E)$ with $E \subseteq A \times B$ has* $\mathrm{qn}(G) \leq 56$.

Note that all edges of $G = (A \cup B, E)$ are oriented from a vertex in A to a vertex in B.

Proof. Let $\overrightarrow{G} = (A \cup B, E)$ be an upward planar bipartite graph with $E \subseteq A \times B$ and G the underlying undirected graph. Förster, Kaufmann, Merker, Pupyrev, and Raftopoulou [FKM+23] proved that the queue number of undirected bipartite planar graphs is at most 28. We now use a queue layout Γ of G with vertex ordering $\prec$ to construct a queue layout Γ' of $\overrightarrow{G}$. For the vertex ordering $\prec'$ of Γ' we choose $a \prec' b$ for each $a \in A$, $b \in B$. Since we only have edges (a, b) with $a \in A$, $b \in B$ in the directed graph $\overrightarrow{G}$, the ordering is indeed topological. Within the vertex sets A and B, we leave the ordering as it is for Γ. More precisely, we define $u \prec' v$ if and only if $u \prec v$ for each $u, v \in A$ or $u, v \in B$. Then, we take a queue Q from Γ and partition it into two queues Q_1, Q_2 for Γ', whereby Q_1 contains all edges $(a, b) \in Q$ with $a \prec b$ and Q_2 contains all edges $(a, b) \in Q$ with $b \prec a$. Next, we show that Q_1 and Q_2 satisfy the queue property.

Suppose, Q_1 is not a queue. Then Q_1 contains two nesting edges (a_1, b_1), (a_2, b_2) with $a_1 \prec' a_2 \prec' b_2 \prec' b_1$. According to the construction of $\prec'$, it follows that $a_1 \prec a_2$ and $b_2 \prec b_1$. It also follows from the definition of Q_1 that $a_2 \prec b_2$, and thus $a_1 \prec a_2 \prec b_2 \prec b_2$, so the two edges also nest in Γ, which is a contradiction.

Similarly, we now assume Q_2 is not a queue, and hence contains two nesting edges (a_1, b_1), (a_2, b_2) with $a_1 \prec' a_2 \prec' b_2 \prec' b_1$. Same as for Q_1, it follows from the construction of $\prec'$ that $a_1 \prec a_2$ and $b_2 \prec b_1$ and from the definition of Q_2 that $b_1 \prec a_1$. Therefore, we have $b_2 \prec b_1 \prec a_1 \prec a_2$ and the two edges also nest in Γ, which is again a contradiction.

Thus, we have shown that for each queue of Γ, at most two queues are necessary in Γ'. Having at most 28 queues in Γ, 56 queues are sufficient for Γ'. □

Thereby, we have not only shown that a 56-queue layout exists for every upward planar bipartite graph $G = (A \cup B, E)$ with $E \subseteq A \times B$, but also that there exists such a layout with first all vertices from A and then all vertices from B. Dujmović, Pór, and Wood observed that such an ordering can be reversed so that stacks are converted to queues and queues are converted to stacks [DPW04]. Thus, the stack and queue number are equivalent in this special case. We prove this observation in the following.

Lemma 4.3. *For each upward planar bipartite graph $G = (A \cup B, E)$ with $E \subseteq A \times B$ and fixed topological ordering $\prec$ with $a \prec b$ for each $a \in A$, $b \in B$ holds: If G admits a q-queue layout with the given ordering $\prec$, there is also an ordering $\prec'$, for which G admits*

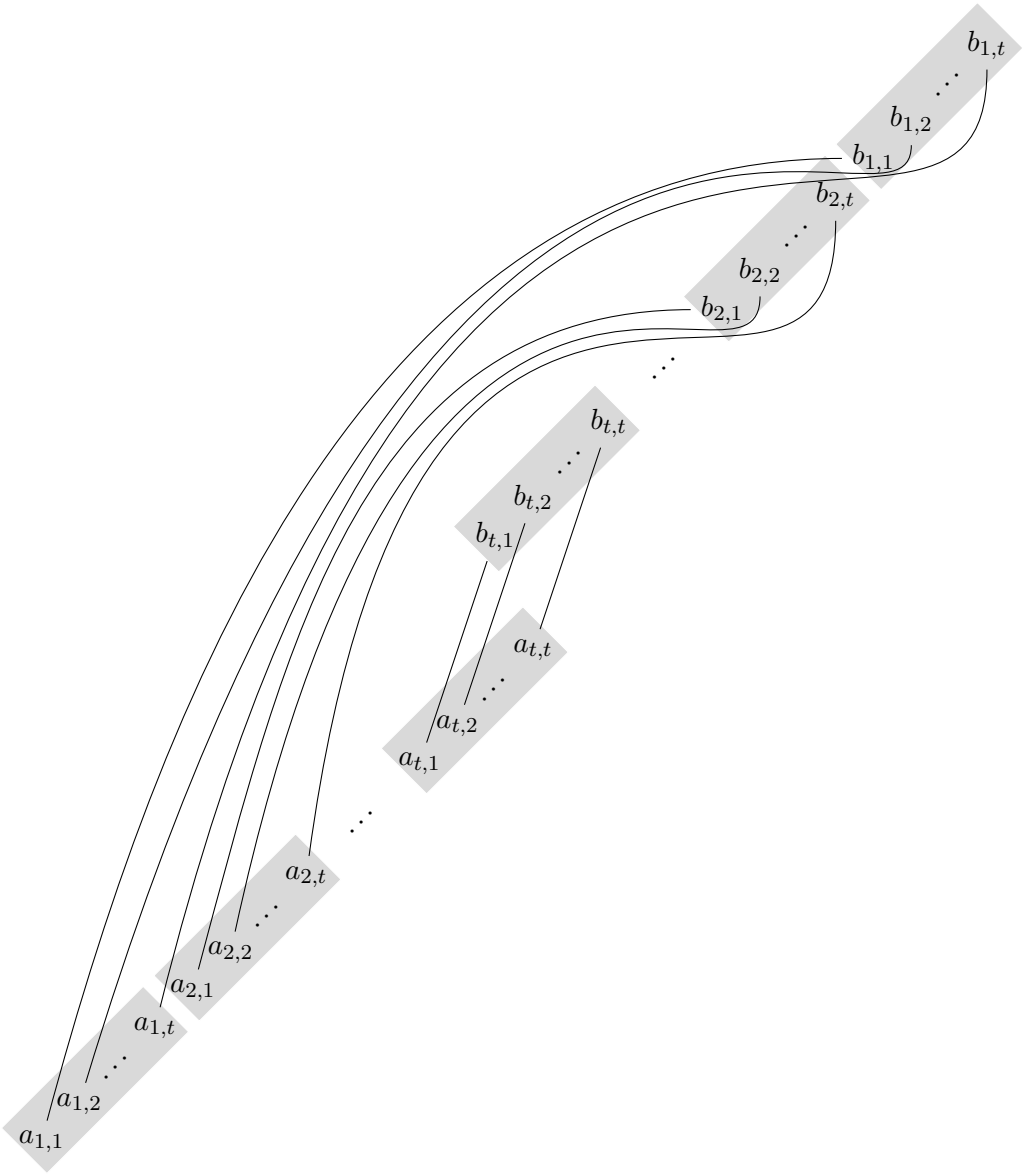


Figure 4.1: An upward planar bipartite graph G_t, for which at least t pages are necessary if we choose a vertex ordering from left to right. The grey boxes mark the vertex groups A_i and B_i for $i = 1, \ldots, t$.

a q-stack layout, and if G admits a s-stack layout with the given ordering $\prec$, there is also an ordering $\prec'$, for which G admits a s-queue layout.

Here, all edges are oriented from a vertex in A to a vertex in B, too.

Proof. Let $G = (A \cup B, E)$ be an upward planar bipartite graph with $E \subseteq A \times B$ and vertex ordering $\prec$ with $a \prec b$ for each $a \in A$, $b \in B$. In such an ordering each pair of different edges either crosses or nests. Consider two edges (a_1, b_1), (a_2, b_2) with $a_1 \prec a_2$. There are two valid orderings for the four vertices. It is either $b_1 \prec b_2$ and thus $a_1 \prec a_2 \prec b_1 \prec b_2$ and the two edges cross, or it is $b_2 \prec b_1$ and hence $a_1 \prec a_2 \prec b_2 \prec b_1$ and the two edges nest. While leaving the ordering of the vertices from A, we now reverse the ordering of the vertices from B to obtain $\prec'$. We show that two edges that nest in the ordering $\prec$ cross in the new ordering $\prec'$ and vice versa.

Let (a_1, b_1), (a_2, b_2) be two nesting edges, so without loss of generality $a_1 \prec a_2 \prec b_2 \prec b_1$. Thus, we have $a_1 \prec' a_2 \prec' b_1 \prec' b_2$ and the two edges cross in the new vertex ordering $\prec'$. Now, let (a_1, b_1), (a_2, b_2) be two crossing edges, i.e., $a_1 \prec a_2 \prec b_1 \prec b_2$. Hence, it is $a_1 \prec' a_2 \prec' b_2 \prec' b_1$ and the edges nest in the reversed ordering $\prec'$.

In a queue from the original layout there are no nestings, so each pair of different edges crosses. Hence, in the changed ordering each pair of different edges nests but there are no crossings and the previous queue now satisfies the stack property. The same argument applies in the reversed case and the claim follows. □

This lemma allows us to infer from the queue number to the stack number and vice versa in some cases. For instance, the following corollary can be concluded.

Corollary 4.4. *Each upward planar bipartite graph $G = (A \cup B, E)$ with $E \subseteq A \times B$ has* $\mathrm{sn}(G) \leq 56$.

Again, all edges are oriented from a vertex in A to a vertex in B.

4.2 Lower Bound

Förster, Kaufmann, Merker, Pupyrev, and Raftopoulou [FKM+23] constructed an undirected bipartite planar graph $G_d(w)$ with $d \geq 3$ and $w \geq 154$ that does neither admit a 2-queue layout nor a 1-queue 1-stack layout. The graph $G_d(w)$ is constructed by starting with two vertices, called *depth-0 vertices*. For each $0 \leq i \leq d-1$, w vertices, called *depth-$(i+1)$ vertices*, are added into each inner face of $G_i(w)$ to obtain $G_{i+1}(w)$, except for $i = 0$, where the depth-1 vertices are added into the unique outer face. If the face has two depth-i vertices, the new depth-$(i+1)$ vertices are connected with them. Else, if the face has only one depth-i vertex v, the depth-$(i+1)$ vertices are connected with v and with the other vertex on the boundary of the face, that is not adjacent to v. The subgraph of $G_3(154)$ which is considered below is illustrated in Figure 4.2. Following this approach, we orient all edges such that we obtain an upward planar bipartite graph $\overrightarrow{G_d}(w)$ with $\mathrm{mpn}(\overrightarrow{G_d}(w)) \geq 3$.

Proposition 4.5. *There is an upward planar bipartite graph G with* $\mathrm{mpn}(G) \geq 3$.

Proof. Consider the bipartite planar graph $G_d(w)$. Since $G_3(154)$ already does not admit a 2-queue layout or a 1-queue 1-stack layout in the undirected case, it also does not do so when we orient the edges. To give the lower bound of 3 on the mixed page number of upward planar bipartite graphs, we only need to show that after all edges have been

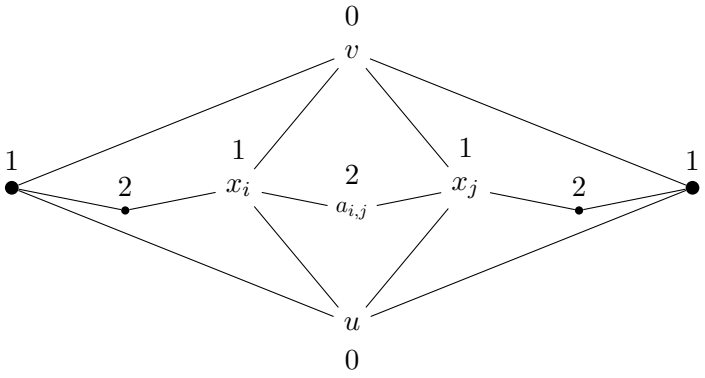


Figure 4.2: A subgraph G of the upward planar bipartite graph $\overrightarrow{G_3}(154)$ with $\text{mpn}(G) \geq 3$. The numbers above and below the vertices indicate the depth of the respective vertex.

oriented in an upward way, the obtained graph $\overrightarrow{G_3}(154)$ does not admit a 2-stack layout. Therefore, we consider a subgraph of $G_3(154)$ with two depth-0 vertices as in $G_3(154)$, but where only four depth-1 vertices, one depth-2 vertex, and no depth-3 vertices are added into the respective faces (see Figure 4.2). The edges of the considered subgraph of the directed graph $\overrightarrow{G_3}(154) = (V, \overrightarrow{E})$ are oriented from bottom to top in the fixed embedding illustrated in Figure 4.2. All other edges of $\overrightarrow{G_3}(154)$ are arbitrarily oriented such that $\overrightarrow{G_3}(154)$ is upward planar. Let u and v be the two depth-0 vertices and x_i for $i = 1, \ldots, 4$ the depth-1 vertices. The depth-2 vertex adjacent to x_i and x_j is called $a_{i,j}$. Then, the edges are oriented from u to the x_i and from the x_i to v, i.e., $(u, x_i), (x_i, v) \in \overrightarrow{E}$ for $i = 1, \ldots, 4$. Further, the edges between depth-1 and the depth-2 vertices are oriented from $a_{i,j}$ to x_i and x_j. Precisely, we have $(a_{i,j}, x_i), (a_{i,j}, x_j) \in \overrightarrow{E}$ for $i, j = 1, \ldots, 4$. Therefore, in any topological ordering $\prec$ it is $u \prec x_i \prec v$ for each $i = 1, \ldots, 4$. The four depth-1 vertices x_i are numbered according to the order in which they occur in $\prec$, i.e., $x_1 \prec x_2 \prec x_3 \prec x_4$ (see Figure 4.3).

Now, we fix such a topological ordering $\prec$ and prove that 2 stacks are not sufficient. To do this, we assume that 2 stacks suffice and consider a 2-stack layout with stacks S_1, S_2. Then, we show that all but at most one edge of the form (u, x_i) are in the same stack S_1, and all but at most one edge of the form (x_i, v) are in the other stack S_2. Without loss of generality, we assume that the edge (u, x_4) is in S_1. Thus, all edges crossing (u, x_4), i.e., all edges (x_i, v) for $i = 1, \ldots, 3$, are in S_2. Hence, all edges crossing (x_1, v), i.e., all edges (u, x_i) for $i = 2, \ldots, 4$, are in S_1. Only the edges (u, x_1) and (v, x_4) are in any of the two stacks.

Lastly, we consider the depth-2 vertices. For this, we only consider the depth-2 vertex $a_{2,3}$, which is adjacent to the depth-1 vertices x_2 and x_3. As we showed before, it is $(u, x_2) \in S_1$ and $(x_3, v) \in S_2$. Next, we show that $(a_{2,3}, x_3)$ crosses edges from both S_1 and S_2, and thus cannot be embedded in the 2-stack layout. It holds that $a_{2,3} \prec x_2 \prec x_3 \prec x_4 \prec v$, whereby the edges $(a_{2,3}, x_3)$ and (x_2, v) cross. Since (x_2, v) is in S_2, $(a_{2,3}, x_3)$ is in S_1. Now, we distinguish the two cases $a_{2,3} \prec u$ and $u \prec a_{2,3}$. In the former case the edge $(a_{2,3}, x_3)$ crosses the edge (u, x_4). As (u, x_4) is in S_1, this is a contradiction. Similarly, in the latter case the edges $(a_{2,3}, x_3)$ and (u, x_2) cross, which is again a contradiction because both of the edges are in S_1. □

Up until now, we considered an upward planar bipartite graph, for which two pages are not sufficient, neither when we use stacks nor queues nor both of them. Now, we focus on another upward planar bipartite graph, that also does not admit a 2-queue and a 2-stack

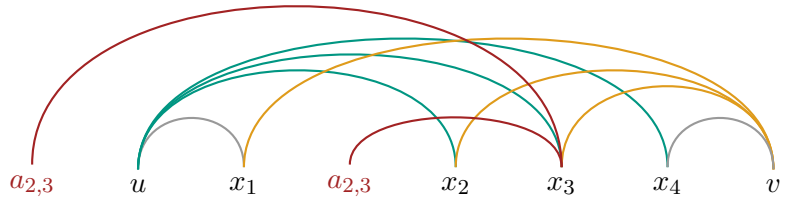


Figure 4.3: The linear layout of the subgraph of $\overrightarrow{G_3}(154)$ considered in Proposition 4.5 with the two stacks S_1 (green) and S_2 (orange). In any topological ordering the edge $(a_{i,j}, x_j)$ crosses both edges from S_1 and S_2.

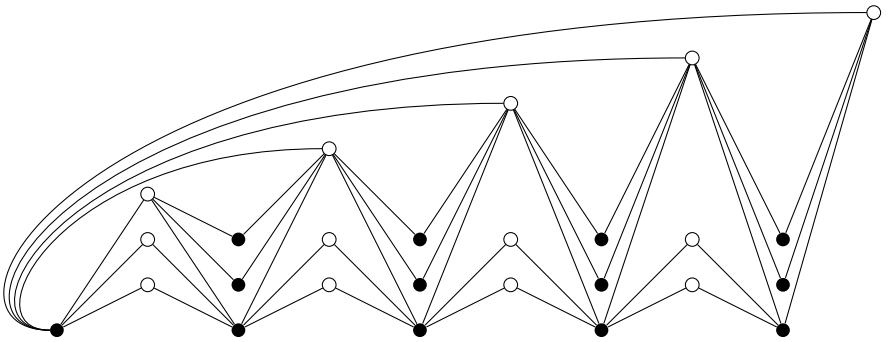


Figure 4.4: An upward planar bipartite graph G with $\mathrm{mpn}(G) < \mathrm{sn}(G), \mathrm{qn}(G)$ and all edges directed from black to white verties.

layout but does admit a 1-queue 1-stack layout. This graph $G = (A \cup B, E)$ is illustrated in Figure 4.4. We verified by using a SAT-solver that $\mathrm{sn}(G), \mathrm{qn}(G) \geq 3$ and $\mathrm{mpn}(G) \leq 2$ [BHKM20]. From this, the proposition below follows.

Proposition 4.6. *There is an upward planar bipartite graph G with* $\mathrm{mpn}(G) < \mathrm{sn}(G)$, $\mathrm{qn}(G)$.

We remark that G is not our first example for an upward planar graph with mixed page number strictly smaller than stack and queue number as we showed in Corollary 3.9 but our first example that is additionally bipartite. Note that all edges are directed from black to white vertices, i.e., $E \subseteq A \times B$. This makes G an example for the graph class considered in Section 4.1, for which we have an upper bound on the stack and queue number $\mathrm{sn}(G), \mathrm{qn}(G) \leq 56$.

5. Directed Acyclic 2-Trees

In contrast to upward planar graphs, the stack number of directed acyclic 2-trees is known to be unbounded [JMU22b]. Up until now, it is not known whether the same applies to the mixed page number. Since an unbounded mixed page number strengthens the proposition of an unbounded stack number, we focus on the mixed page number of directed acyclic 2-trees in the following chapter. First, we investigate the only known directed acyclic 2-tree with unbounded stack number [JMU22b] in hope to find an answer to this question. Surprisingly, the mixed page number of this concrete graph family is bounded by a constant, although the queue number is unbounded as is the stack number. Therefore, we choose a different approach in Section 5.2, where we finally show that the mixed page number of directed acyclic 2-trees is unbounded.

5.1 A Directed Acyclic 2-Tree with Unbounded Stack and Queue Number but Bounded Mixed Page Number

Alam, Bekos, Gronemann, Kaufmann, and Pupyrev [ABG+22b] asked whether graphs with bounded mixed page number have bounded stack or queue number. We answer this question in the negative by giving a graph family with bounded mixed page number but unbounded stack and queue number. This family of directed acyclic 2-trees G_k was constructed by Jungeblut, Merker, and Ueckerdt [JMU22b], who also showed that $\mathrm{sn}(G_k) \geq k$ for each $k \geq 1$. We show in the following that this additionally monotone 2-tree G_k also has an unbounded queue number, but that the mixed page number is bounded by the constant 3. This leads to the theorem below.

Theorem 5.1. *There is a family of directed acyclic (monotone) 2-trees with unbounded stack and queue number but constant mixed page number.*

To prove this theorem, we show both that the queue number is unbounded and that the mixed page number is bounded, below. But first, we describe the construction of G_k.

Let N be an enormous, but constant integer. How large N is exactly is only relevant for the stack number but not for the queue and mixed page number. Therefore we do not need to specify an exact value for N here. For $k \geq 0$ the graph G_k is defined inductively as follows: The graph G_0 only contains the unique base edge (a, b) in the here described construction sequence and has the corresponding endpoints a and b. We define $M_0 := \{(a, b)\}$. In general, for $k \geq 0$ the edge set M_k is a subset of the edges of G_k used to construct G_{k+1}. We define this subset M_k in the following more precisely. The edges in M_k are called *matching*

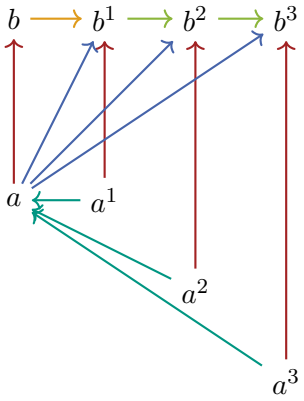


Figure 5.1: The 2-tree obtained from the base edge (a, b) after one step with three left and right descendants. The edges are partitioned into the matching edges E_1 (red), the set E_2 of edges from left descendants a^i to their left ancestor a (dark green), the set E_3 of edges from a to its right descendants b^i (blue), the set E_4 with the edge from b to its right child and first descendant b^1 (orange), and the set E_5 of edges between the right descendants of the same ancestor edge (light green).

edges. By constructing G_k for $k \geq 1$, the edges M_{k-1} from G_{k-1} serve as base edges for the subtrees newly added in this step. Starting with such a matching edge $(a_i, b_i) \in M_{k-1}$, we add a set of N vertices $b_i^1, \ldots, b_i^N$, whereby b_i^j is a right child of the edge (a_i, b_i^{j-1}) and $b_i^0 = b_i$. Then, we add N vertices $a_i^1, \ldots, a_i^N$, whereby a_i^j is a left child of (a_i, b_i^j). The edges $E((a_i, b_i)) = \{(a_i^j, b_i^j) \mid 1 \leq j \leq N\}$ become the new matching edges, i.e.,

$$M_k := \bigcup_{(a_i,b_i)\in M_{k-1}} E((a_i, b_i)).$$

The graph G_2 obtained after one step is illustrated in Figure 5.1. We call the vertices a_i^j *left descendants* and the vertices b_i^j *right descendants* of the matching edge (a_i, b_i) and its two end points a_i and b_i. Conversely, a_i and b_i are called the *ancestors* of a_i^j and b_i^j and (a_i, b_i) the *ancestor edge*. In some cases we also distinguish the two ancestors a_i and b_i by calling a_i the *left ancestor* and b_i the *right ancestor*. Note that in the context of the definition of G_k the terms descendant and ancestor always refer to the matching edge in M_{k-1}, below which the subtree with the here considered vertex was attached by constructing G_k. Vertices and edges added by constructing G_k are said to be in *level* k. Precisely, each matching edge $(a_i, b_i) \in M_k$ is in level k and for each edge $(a_i, b_i) \in M_k$ in level k all newly added descendants a_i^j, b_i^j and all edges incident to a_i^j and b_i^j are in level $k + 1$.

Proposition 5.2. *The monotone* 2*-tree* G_k *has mixed page number* $\mathrm{mpn}(G_k) \leq 3$ *for each* $k \geq 0$.

Proof. We give a 1-queue 2-stack layout of G, and thereby show that $\mathrm{mpn}(G) \leq 3$. We start constructing the topological vertex ordering $\prec$ again with G_0, which consists of the base edge (a, b). For this, it holds that $a \prec b$. For each $k > 0$ the newly added vertices from G_k, i.e., those in level k, are attached to the left and right of the previous ordering as follows: Vertices a_i^j of level k are added to the left of all vertices with smaller levels, and vertices b_i^j of level k are added to the right of all vertices with smaller levels, so that the levels become larger towards the outside of the layout. Left descendants $a_i^j, a_{i'}^{j'}$ of different ancestors a_i, b_i and $a_{i'}, b_{i'}$ in the same level with $a_i \prec a_{i'}$ are added in such a way that

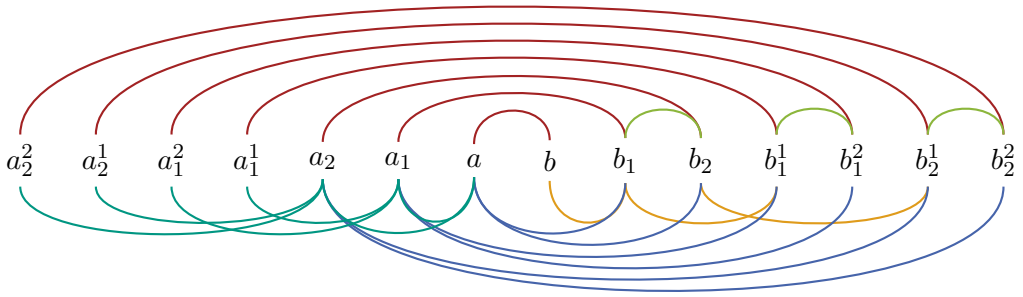


Figure 5.2: The complete 1-queue 2-stack layout of G consisting of the stacks E_1 (red) and $E_3 \cup E_5$ (blue and light green) and the queue $E_2 \cup E_4$ (dark green and orange). The edge (a, b) is the original base edge. The vertices a_1, a_2 are the left descendants of (a, b) and b_1, b_2 are the right descendants of (a, b). The vertices a_i^j, b_i^j are the left and right descendants of the matching edges (a_i, b_i) for $i, j \in \{1, 2\}$.

$a_i^j \prec a_{i'}^{j'}$. Analogously, for descendants $b_i^j, b_{i'}^{j'}$ of different ancestors a_i, b_i and $a_{i'}, b_{i'}$ in the same level with $b_i \prec b_{i'}$, it holds that $b_i^j \prec b_{i'}^{j'}$. Thus, left descendants of different ancestors are in the same order as their left ancestor, and right descendants of different ancestors are in the same order as their right ancestor. Furthermore, it follows that left descendants of the same ancestors are consecutive in the ordering, and the same applies to the right descendants. Moreover, we define for descendants of the same ancestor edge (a_i, b_i) that $a_i^{j'} \prec a_i^j$ and $b_i^j \prec b_i^{j'}$ if $j < j'$, i.e., the left descendants are in the reverse order of the right descendants (see Figure 5.2).

Next, we partition the edges of G into five sets as illustrated in Figure 5.1. For these five edge sets, we prove the stack or queue property, before we union some of them, so that we finally have three pages. The first edge set E_1 contains the matching edges of all levels, i.e., $E_1 = \bigcup_{i=0}^{k} M_i$. All edges from left descendants a_i^j to their left ancestor a_i are in E_2, and analogously all edges from left ancestors a_i to their right descendants b_i^j are in E_3. In the fourth edge set E_4 are the edges from vertices b_i to their only child and first descendant b_i^1. The last set E_5 contains the remaining edges from one right descendant b_i^j to the next right descendant b_i^{j+1} of the same ancestors for $j = 1, \ldots, N-1$. The sets E_1 and E_3 both are stacks, and E_2 and E_4 both are queues. The last set E_5 satisfies both the stack and the queue property. We prove this in the following by showing that no two edges from different levels, no two edges in the same level, where the endpoints are descendants from different ancestors, and no two edges, where the endpoints are descendants from the same ancestors, nest or cross respectively for each edge set.

First, we show that E_1 is a stack, i.e., no two edges from E_1 cross with respect to $\prec$ (see Figure 5.3). Let $(a_i, b_i), (a_{i'}, b_{i'}) \in E_1$ be two edges from different levels. Without loss of generality, we assume that (a_i, b_i) is in level l and $(a_{i'}, b_{i'})$ is in level l' with $l < l'$. Since the levels become larger towards the outside of the linear layout, we have $a_{i'} \prec a_i \prec b_i \prec b_{i'}$ and the edges do not cross. Now, consider two edges $(a_i^j, b_i^j), (a_{i'}^{j'}, b_{i'}^{j'}) \in E_1$ in the same level, but where the endpoints are descendants of different ancestors a_i, b_i and $a_{i'}, b_{i'}$ with $a_i \prec a_{i'}$ and thus $b_{i'} \prec b_i$. It follows that $a_i^j \prec a_{i'}^{j'} \prec b_{i'}^{j'} \prec b_i^j$. Lastly, let $(a_i^j, b_i^j), (a_i^{j'}, b_i^{j'}) \in E_2$ be two edges where the endpoints are descendants of the same ancestors and $j < j'$. Due to the reverse ordering of the left descendants in comparison to the right descendants, we have $a_i^{j'} \prec a_i^j \prec b_i^j \prec b_i^{j'}$. Hence, there are no crossings within the set E_1. We remark that the edges from E_1 all pairwise nest, and thus form a large rainbow.

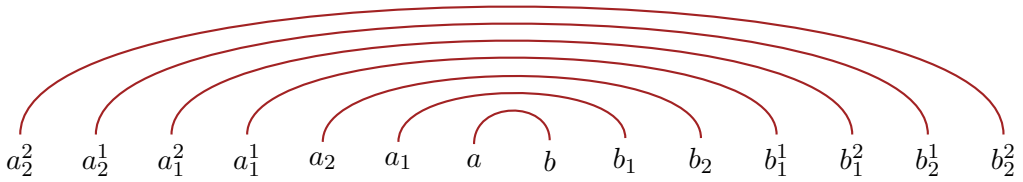


Figure 5.3: The set E_1 of matching edges in the 1-queue 2-stack layout of G.

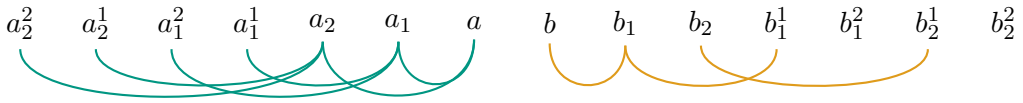


Figure 5.4: The queue consisting of the edge sets E_2 (dark green), which contains edges from left descendants a_i^j to their left ancestor a_i, and E_4 (orange), which contains edges from vertices b_i to their right child b_i^1 in the 1-queue 2-stack layout of G.

Next, we show that E_2 is a queue, so no two edges from E_2 nest in the linear layout (see Figure 5.4). Again, we first consider edges $(a_i^j, a_i), (a_{i'}^{j'}, a_{i'}) \in E_2$ from different levels l and l' with $l < l'$, i.e., a_i^j is in level l, a_i in level $l-1$, $a_{i'}^{j'}$ in level l' and $a_{i'}$ in level $l'-1$. From $l < l'$ it follows that $l - 1 < l \leq l' - 1 < l'$, which leads to $a_{i'}^{j'} \prec \{a_{i'}, a_i^j\} \prec a_i$, as the levels become larger towards the outside of the linear layout. With this ordering, the two edges do not nest. Now, let $(a_i^j, a_i), (a_{i'}^{j'}, a_{i'})$ be two edges from the left descendants a_i^j and $a_{i'}^{j'}$ to their respective left ancestor a_i or $a_{i'}$ in the same level with $a_i \prec a_{i'}$. Descendants are always in a larger level than their ancestor, which is why left descendants are always to the left of their left ancestors. Since further left descendants of different ancestors are in the same ordering as their left ancestors, it follows that $a_i^j \prec a_{i'}^{j'} \prec a_i \prec a_{j'}$ and the edges do not nest. Finally, edges from the left descendant a_i^j to the common ancestor a_i are all incident to the same vertex a_i, and thus do not nest.

Similarly, we show that E_3, which contains all edges from vertices a_i to their right descendants b_i^j, is a stack (see Figure 5.5). Consider two edges $(a_i, b_i^j), (a_{i'}, b_{i'}^{j'}) \in E_3$ of different levels l and l' with $l < l'$, i.e., b_i^j is in level l, $b_{i'}^{j'}$ in level l', a_i is in level $l-1$, and $a_{i'}$ in level $l'-1$. Since the levels become larger towards the outside of the linear layout, it follows that $a_{i'} \prec a_i \prec b_i^j \prec b_{i'}^{j'}$ and the edges do not cross. Next, let $(a_i, b_i^j), (a_{i'}, b_{i'}^{j'}) \in E_3$ be two edges of the same level with $a_i \prec a_{i'}$. Due to the reverse ordering of right descendants in comparison to the left descendants, it follows that $b_{i'} \prec b_i$. Since right descendants are in the same ordering as their right ancestors, it holds that $a_i \prec a_{i'} \prec b_{i'}^{j'} \prec b_i^j$. Analogously to E_2, edges from the same ancestor a_i to its descendants b_i^j are all incident to the same vertex a_i, and therefore do not cross.

Now, we show that the set of edges from vertices b_i to their only child and first right descendant b_i^1, E_4, satisfies the queue property (see Figure 5.4). First, we consider edges $(b_i, b_i^1), (b_{i'}, b_{i'}^1) \in E_4$ of different levels l, l' with $l < l'$, i.e., b_i^1 is in level l, $b_{i'}^1$ in level l', b_i in level $l-1$ and $b_{i'}$ in level $l'-1$. It holds that $l - 1 < l \leq l' - 1 < l'$. Since the levels become larger towards the outside of the linear layout, it follows that $b_i \prec \{b_i^1, b_{i'}\} \prec b_{i'}^1$ and the edges do not nest. Next, let $(b_i, b_i^1), (b_{i'}, b_{i'}^1) \in E_4$ be two edges of the same level with $b_i \prec b_{i'}$. Since right descendants of different ancestors are in the same ordering as their right ancestors, it holds that $b_i \prec b_{i'} \prec b_i^1 \prec b_{i'}^1$ and the edges do not nest. Lastly, for each parent vertex b_i there is only one edge to its right child in E_4, so there are no two edges from the same parent, and thus no nestings.

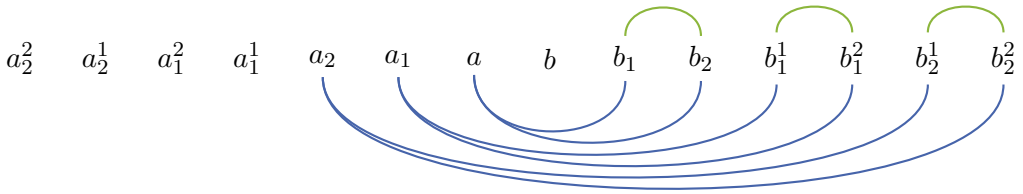


Figure 5.5: The stack consisting of the edge sets E_3 (blue), which contains edges from vertices a_i to their right descendants b_i^j, and E_5 (light green), which contains edges from one right descendant b_i^j to the next right descendant b_i^{j+1} of the same ancestors, in the 1-queue 2-stack layout of G.

The set E_5 contains edges from one right descendant to the next right descendant of the same ancestors, which are consecutive in the vertex ordering. Thus, the edges do not cross or nest with each other and satisfy the stack and queue property (see Figure 5.5).

Since the endpoints of the edges in E_5 are consecutive in $\prec$, it also follows that the edges from E_5 do not cross any other edge from another edge set, particularly not from E_3. Hence, $E_5 \cup E_3$ is still a stack (see Figure 5.5). We remark that the endpoints of the edges in E_2 are all separated from the endpoints of the edges in E_4. Thus, these two edge sets can also be combined to one larger queue $E_2 \cup E_4$ (see Figure 5.4). In total only three pages are necessary, namely the stacks E_1 and $E_3 \cup E_5$ and the queue $E_2 \cup E_4$. □

Proposition 5.3. *The monotone 2-tree G_k has queue number* $\mathrm{qn}(G_k) \geq k+1$.

Proof. We consider a subgraph G' of G with only one left and right descendant per matching edge. The graph G' only consists of the vertices a_i and b_i for $i = 0, \ldots, k$, whereby a_i and b_i are the only left and right descendants of the matching edge (a_{i-1}, b_{i-1}), respectively. In every topological ordering it is $a_i \prec a_{i-1}$ because of the edge (a_i, a_{i-1}), and analogously $b_{i-1} \prec b_i$ because of the edge (b_{i-1}, b_i). Due to the matching edge (a_{i-1}, b_{i-1}) it is $a_i \prec a_{i-1} \prec b_{i-1} \prec b_i$ for each $i = 1, \ldots, k$. In total, we obtain the vertex ordering $a_k \prec \cdots \prec a_2 \prec a_1 \prec b_1 \prec b_2 \prec \cdots \prec b_k$, where the edges (a_i, b_i) pairwise nest for each $i = 0, \ldots, k$, so that we have a $k+1$-rainbow. Hence, at least $k+1$ queues are necessary for G'. □

Now, we not only have graph families with mixed page number strictly smaller than stack and queue number, but also with constant mixed page number and arbitrarily large stack and queue number. The question that now arises is whether all directed acyclic 2-trees have a mixed page number bounded by a constant. In the following section, we answer this question in the negative.

5.2 The Mixed Page Number of General Directed Acyclic 2-Trees

Even though the graph constructed by Jungeblut, Merker, and Ueckerdt [JMU22b] to prove the unbounded stack number of directed acyclic 2-trees does not serve as an example for a graph family with an unbounded mixed page number, we sketch a proof that the mixed page number is nevertheless unbounded in the following section.

Theorem 5.4. *The mixed page number of directed acyclic 2-trees is unbounded, i.e., for each $k \geq 1$ there is a directed acyclic 2-tree G_k with* $\mathrm{mpn}(G_k) \geq k$.

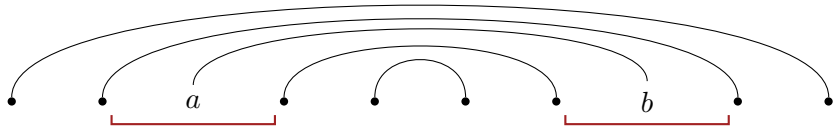


Figure 5.6: A rainbow $R = (V_R, E_R)$, where all children of a rainbow's edge $(a, b) \in E_R$ are between their parent edge (a, b) and its two neighbouring rainbow edges as in Case (i), i.e., here all children of (a, b) are in the red area around a and b.

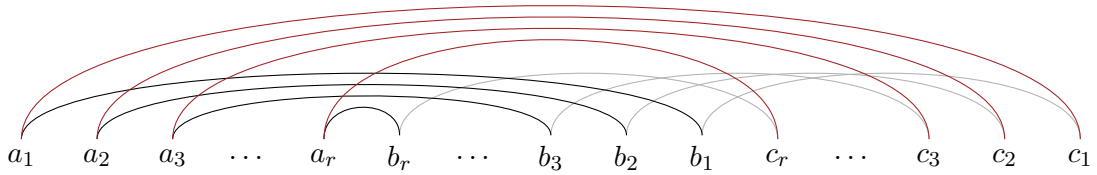


Figure 5.7: An r-rainbow $R = (V_R, E_R)$ (black), whose edges all have a right child. Those right children c_i of edges $(a_i, b_i) \in E_R$ are to the right of the rainbow in the reversed ordering as their left parents a_i as in Case (ii), such that the edges (a_i, c_i) form an r-rainbow (red).

Proof Sketch

To prove Theorem 5.4, we construct an enormous directed acyclic 2-tree G, from the outset with a topological vertex ordering. Since the queue number of directed acyclic 2-trees is unbounded (see Proposition 5.3), we may start our construction with a large rainbow. For this, we only need to assume that G contains a subgraph with an unbounded queue number, and thus unbounded size of a rainbow in any topological vertex ordering (see Theorem 2.11). Specifically, the r-rainbow $R = (V_R, E_R)$ consists of vertices $V_R = \{a_i \mid 1 \leq i \leq r\} \cup \{b_i \mid 1 \leq i \leq r\}$ and edges $E_R = \{(a_i, b_i) \mid 1 \leq i \leq r\}$ with $a_1 \prec \cdots \prec a_r \prec b_r \prec \cdots \prec b_1$. Next, we add children to the edges of this rainbow R and consider different cases for the vertex ordering, until we obtain a large crossing rainbow or a large nesting twist in each of those cases. As a consequence of Ramsey's theorem [Ram30], there are only the following three cases to be considered for children c_i of edges $(a_i, b_i) \in E_R$. If R is a twist, there are the same cases to be considered, which we will also need later.

(i) Children c_i can be close to their parent edge, i.e., $a_{i-1} \prec c_i \prec a_{i+1}$ or $b_{i+1} \prec c_i \prec b_{i-1}$, if R is a rainbow (see Figure 5.6), or $b_{i-1} \prec c_i \prec b_{i+1}$, if R is a twist, for $i = 2, \ldots, r-1$.

(ii) Right children c_i can be to the right of R in the reversed ordering as their left parents, i.e., $c_r \prec \cdots \prec c_1$ (see Figure 5.7). Symmetrically, left children c_i can be to the left of R in the reversed ordering as their right parents, i.e., $c_1 \prec \cdots \prec c_r$. Then, for right children, edges (a_i, c_i) pairwise nest, while for left children, the edges (c_i, b_i) pairwise nest.

(iii) Right children c_i can be to the right of R in the same ordering as their left parents, i.e., $c_1 \prec \cdots \prec c_r$ (see Figure 5.8). Symmetrically, left children c_i can be to the left of R in the same ordering as their right parents, i.e., $c_r \prec \cdots \prec c_i$. Then, for right children, edges (a_i, c_i) pairwise cross, and for left children, edges (c_i, b_i) pairwise cross.

In truth, not all children and edges behave according to the same case. We choose the case that fits to the majority of children and ignore all other vertices and edges. Thus, if we apply these cases repeatedly, we always obtain a smaller but still very large rainbow or

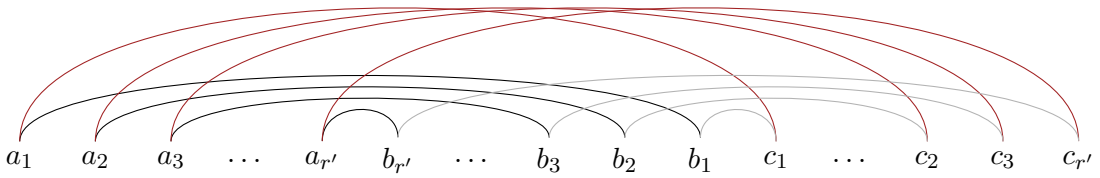


Figure 5.8: An r-rainbow $R = (V_R, E_R)$ (black), whose edges all have a right child. Those right children c_i of edges $(a_i, b_i) \in E_R$ are to the right of the rainbow in the same ordering as their left parents a_i as in Case (iii), such that the edges (a_i, c_i) form an r-twist (red).

twist. In this regard, we go into detail later. For simplification, we now assume that one of these cases always applies to all children of the considered rainbow's or twist's edges.

In the following lemmas, we investigate these possible cases more detailed, first separately, before we later combine them. Particularly, we now assume that only one of the three cases occurs repeatedly in a row. Here, we first consider Case (i) and assume that all descendants of the rainbow R behave according to Case (i). Lemma 5.5 below follows from the proof of Theorem 4 in [JMU22b].

Lemma 5.5. *Let $R = (V_R, E_R)$ be an r-rainbow with ordered vertices $a_1 \prec \cdots \prec a_r \prec b_r \prec \cdots \prec b_1$ and edges $E_R = \{(a_i, b_i) \mid 1 \leq i \leq r\}$. For each $t > 0$, there is a set of edges $E_{i,t}$, whose endpoints in $D_{i,t}$ are descendants of the edge $(a_i, b_i) \in E_R$, such that the following holds: If those descendants behave according to Case (i), i.e., it holds that $a_{i-1} \prec c \prec a_{i+1}$ or $b_{i+1} \prec c \prec b_{i-1}$ for each $c \in D_{i,t}$, then those edges in $E_{i,t}$ form a t-twist.*

Now, consider such an r-rainbow $R = (V_R, E_R)$ with descendants of an edge (a_i, b_i) localised between a_{i-1} and a_{i+1} or between b_{i+1} and b_{i-1}, such that they form a t-twist. For two edges $(a_i, b_i), (a_j, b_j) \in E_R$ that are not neighbours in R, i.e., $|i - j| > 1$, the regions where their descendants are localised are disjoint. Therefore, if we only consider every second edge of R with its descendants forming a t-twist as in Lemma 5.5, we obtain $r/2$ pairwise nesting t-twists. It follows that Case (i) may not occur arbitrarily often in a row without generating a large nesting twist. In particular, an $r/2$-nesting t-twist always contains an $r/2$-rainbow, i.e., after placing some descendants of the rainbow's edges close to their ancestor edge from E_R as in Case (i), we still have a large rainbow of size $r/2$, onto which further right children can be stacked applying Cases (ii) and (iii). The start and end points of such a new $r/2$-rainbow are approximately at the same place as the start and end points of the original rainbow R. This observation is recorded below. We will later go into more detail what "approximately" means.

Observation 5.6. *Let $R = (V_R, E_R)$ be an r-rainbow with ordered vertices $a_1 \prec \cdots \prec a_r \prec b_r \prec \cdots \prec b_1$ and edges $E_R = \{(a_i, b_i) \mid 1 \leq i \leq r\}$. If we apply Case (i) (repeatedly) to R, we obtain an $r/2$-rainbow $R' = (V'_R, E'_R)$ with ordered vertices $a'_1 \prec \cdots \prec a'_{r/2} \prec b'_{r/2} \prec \cdots \prec b'_1$ and edges $E'_R = \{(a'_i, b'_i) \mid 1 \leq i \leq r/2\}$, for which it holds that $a_{2i-1} \prec a'_i \prec a_{2i+1}$ and $b_{2i+1} \prec b'_i \prec b_{2i-1}$ for each $i = 1, \ldots, r/2$.*

Next, we consider Case (ii) and repeat this case several times in a row. Due to symmetry, we only consider right children. Those right children of a rainbow's edge are placed outside the rainbow, so that the edges (a_i, c_i) from left parents to the right children pairwise nest. Specifically, we start again with our r-rainbow, which we call $R_0 = (V_0, E_0)$ here. R_0 consists of ordered vertices $a_1 \prec \cdots \prec a_r \prec c^0_r \prec \cdots \prec c^0_1$ and edges $E_0 = \{(a_i, c^0_i) \mid 1 \leq i \leq r\}$. For

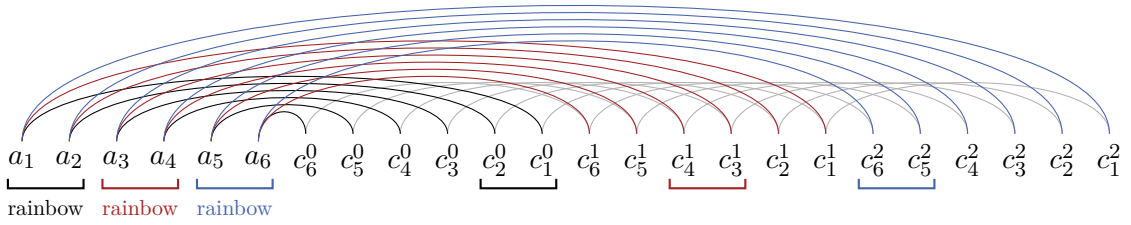


Figure 5.9: A 6-rainbow R (black), onto whose edges right children are stacked repeatedly. Those right children are placed to the right of their parent's rainbow, such that the edges from their left parents to them (red and blue) pairwise nest, as in Case (ii). The outer third of R, the middle third of the red rainbow, and the inner third of the blue rainbow pairwise cross and form a 3-crossing 2-rainbow.

$j = 1, \dots, k$, let c_i^j be a right child of (a_i, c_i^{j-1}). As we always apply Case (ii), the vertex ordering is $c_i^j \prec c_{i'}^{j'}$ for each $j < j'$ or $j = j'$ and $i > i'$, i.e.,

$$a_1 \prec \cdots \prec a_r \prec c_r^0 \prec \cdots \prec c_1^0 \prec c_r^1 \prec \cdots \prec c_1^1 \prec \cdots \prec c_r^k \prec \cdots \prec c_1^k$$

(see Figure 5.9). The pattern that we obtain contains the rainbows $R_j = (V_j, E_j)$ with ordered vertices $a_1 \prec \cdots \prec a_r \prec c_r^j \prec \cdots \prec c_1^j$ and edges $E_j = \{(a_i, c_i^j) \mid 1 \leq i \leq r\}$ for $j = 1, \dots, k$. In total, these rainbows R_j form the edge pattern $P = (V, E, \prec)$ with $V = \bigcup_{j=0}^{k} V_j$ and $E = \bigcup_{j=0}^{k} E_j$.

Lemma 5.7. *The edge pattern P contains a $(k+1)$-crossing $\lfloor r/(k+1) \rfloor$-rainbow.*

Proof. We show that each rainbow R_j contains a rainbow R'_j with $\lfloor r/(k+1) \rfloor$ edges, such that those smaller rainbows R'_j pairwise nest (see Figure 5.9). Let $R'_j = (V'_j, E'_j)$ be the $\lfloor r/(k+1) \rfloor$-rainbow with ordered vertices

$$a_{j\lfloor r/(k+1) \rfloor + 1} \prec a_{j\lfloor r/(k+1) \rfloor + 2} \prec \cdots \prec a_{(j+1)\lfloor r/(k+1) \rfloor} \prec$$

$$c^j_{(j+1)\lfloor r/(k+1) \rfloor} \prec \cdots \prec c^j_{j\lfloor r/(k+1) \rfloor + 2} \prec c^j_{j\lfloor r/(k+1) \rfloor + 1}$$

and the respective edges $E'_j = E_j \cap (V'_j \times V'_j)$. Intuitively, each of the r-rainbows R_j is split into $k+1$ parts of the same size from the outside to the inside, where we only consider the j-th part of each rainbow R_j. Now, we consider two of these $\lfloor r/(k+1) \rfloor$-rainbows R'_j, R'_l with $j < l$ and show that they cross. It holds that

$$a_{j\lfloor r/(k+1) \rfloor + 1} \prec \cdots \prec a_{(j+1)\lfloor r/(k+1) \rfloor} \prec a_{l\lfloor r/(k+1) \rfloor + 1} \prec \cdots \prec a_{(l+1)\lfloor r/(k+1) \rfloor},$$

since $(j+1)\lfloor r/(k+1) \rfloor < l\lfloor r/(k+1) \rfloor + 1$. Furthermore, we have the ordering $a_i \prec c^j_{i'} \prec c^l_{i''}$ for each $1 \leq i, i', i'' \leq r$. Thus, the start points of edges in R'_j are all to the left of the start points of R'_l, which are all to the left of the endpoints of edges in R'_j, which are all to the left of the endpoints of edges in R'_l. Hence, these $\lfloor r/(k+1) \rfloor$-rainbows R'_j pairwise cross for $j = 0, \dots, k$, and thereby form a $(k+1)$-crossing $\lfloor r/(k+1) \rfloor$-rainbow. □

Again, it follows that Case (ii) may not occur arbitrarily often in a row, since this would result in a large crossing rainbow. After having applied Case (ii), we obtain the new rainbows R_j, onto which further children can be stacked. The range over which such a rainbow R_j spans is larger than for R_0 and includes the complete range over which R_0 spans. As we need this to later combine the three cases, we record this observation in the following.

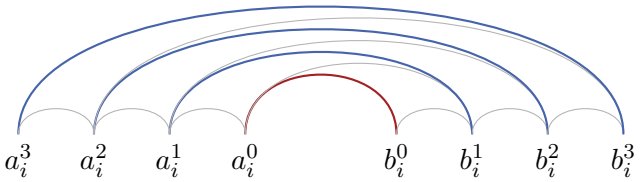


Figure 5.10: Starting with an edge (a_i^0, b_i^0) (red), if we repeatedly stack right children b_i^j onto each edge (a_i^{j-1}, b_i^{j-1}) and left children a_i^j onto each edge (a_i^{j-1}, b_i^j) for $j = 1, \ldots, 3$, we obtain a 4-rainbow (red and blue).

Observation 5.8. *For each $j = 1, \ldots, k$, R_j forms an r-rainbow, whose edges have the same start points as the edges in E_{j-1}, and endpoints to the right of the endpoints in E_{j-1}, i.e., with $c_i^{j-1} \prec c_{i'}^j$ for each $1 \leq i, i' \leq r$.*

Lastly, we investigate Case (iii). In this case, we stack further right children onto the edges of the resulting twist, i.e., each edge (a_i, c_i) gets a right child c_i^1. For those right children, we need to consider all three Cases (i), (ii), and (iii) again. Specifically, we once apply Case (iii) to our r-rainbow R. Then, we investigate the three subcases that Case (i), (ii), or (iii) is applied to the resulting twist very often in a row. Once more, we start with Case (i), and assume all descendants of a twist's edge to be close to their ancestor from the twist.

Lemma 5.9. *Let $T = (V_T, E_T)$ be a t-twist with ordered vertices $a_1 \prec \cdots \prec a_t \prec b_1 \prec \cdots \prec b_t$ and edges $E_T = \{(a_i, b_i) \mid 1 \leq i \leq t\}$. For each $r > 0$, there is a set of edges $E_{i,r}$, whose endpoints in $D_{i,r}$ are descendants of the edge $(a_i, b_i) \in E_R$, such that the following holds: If those descendants behave according to Case (i), i.e., it holds that $a_{i-1} \prec c \prec a_{i+1}$ or $b_{i-1} \prec c \prec b_{i+1}$ for each $c \in D_{i,r}$, then, those edges in $E_{i,r}$ form an r-rainbow.*

Proof. Consider a t-twist $T = (V_T, E_T)$ with respect to the vertex ordering $\prec$ and an edge $(a_i, b_i) \in E_T$. For $r > 0$, let $D_{i,r} = \{a_i^j \mid 1 \leq j \leq r-1\} \cup \{b_i^j \mid 1 \leq j \leq r-1\}$ be a set of descendants, where b_i^j is a right child of (a_i^{j-1}, b_i^{j-1}) and a_i^j is a left child of (a_i^{j-1}, b_i^j) with $a_i = a_i^0$ and $b_i = b_i^0$ for each $j = 1, \ldots, r-1$. It follows that the edges from left children to their left parent (a_i^j, a_i^{j-1}) and the edges from right parents to their right child (b_i^{j-1}, b_i^j) for $j = 1, \ldots, r-1$ exist. This leads to the topological vertex ordering $a_i^{r-1} \prec \cdots \prec a_i^0 \prec b_i^0 \prec \cdots \prec b_i^{r-1}$. With respect to this ordering, the edges (a_i^j, b_i^j) from left children to their right parent pairwise nest for $j = 0, \ldots, r-1$, and thus form an r-rainbow (see Figure 5.10). □

Similar as for Lemma 5.5, we consider such a t-twist $T = (V_T, E_T)$, whose edges (a_i, b_i) have descendants localised between a_{i-1} and a_{i+1} or between b_{i-1} and b_{i+1} for $i = 2, \ldots, t-1$, such that their incident edges form an r-rainbow. For any two edges $(a_i, b_i), (a_j, b_j) \in E_T$ that are not neighbours in T, i.e., with $|i - j| > 1$, the regions where their descendants are placed, are disjoint. Therefore, if we only consider every second edge of T with its descendants forming an r-rainbow as in Lemma 5.9, we obtain $t/2$ pairwise crossing r-rainbows. Again, it follows that Case (i) may not occur arbitrarily often in a row as a subcase of Case (iii). Particularly, a $t/2$-crossing r-rainbow contains a $t/2$-twist, i.e., after placing some descendants of a twist edge close to their ancestor from E_T as in Case (i), we still have a large twist of size $t/2$, onto whose edges further children can be stacked applying Cases (ii) and (iii). The start and endpoints of such a new $t/2$-twist are approximately at the same place as the start and endpoints of T. This observation will later be useful for combining the three cases.

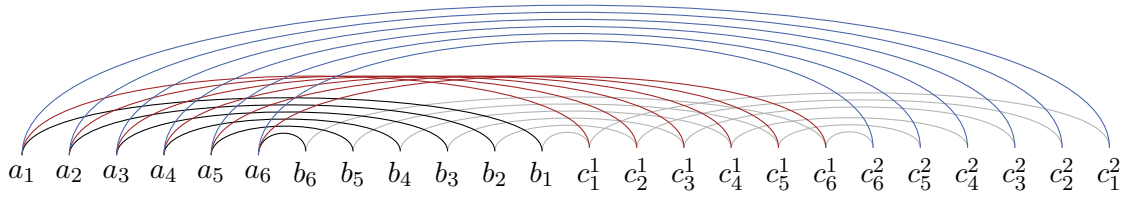


Figure 5.11: Given an r-rainbow R (black). If we first apply Case (iii) to obtain an r-twist T (red), and then apply Case (ii) to this r-twist, then we obtain a new r-rainbow with the same start points as R and T and endpoints to the right of the endpoints from R and T.

Observation 5.10. *Let $T = (V_T, E_T)$ be a t-twist with ordered vertices $a_1 \prec \cdots \prec a_t \prec b_1 \prec \cdots \prec b_t$ and edges $E_R = \{(a_i, b_i) \mid 1 \leq i \leq t\}$. If we apply Case (i) (repeatedly) to T, we obtain a $t/2$-twist $T' = (V'_T, E'_T)$ with ordered vertices $a'_1 \prec \cdots \prec a'_{t/2} \prec b'_1 \prec \cdots \prec b'_{t/2}$ and edges $E'_R = \{(a'_i, b'_i) \mid 1 \leq i \leq t/2\}$, for which it holds that $a_{2i-1} \prec a'_i \prec a_{2i+1}$ and $b_{2i-1} \prec b'_i \prec b_{2i+1}$ for each $i = 1, \ldots, t/2$.*

Next, we consider subcase (ii). Here, we investigate the resulting edge pattern, when we start with our r-rainbow R, then apply Case (iii) once, and then apply Case (ii) to the obtained r-twist T (see Figure 5.11). In this case, we only obtain a new r-rainbow R', whose edges have the same start points as the edges of the r-twist T obtained after applying Case (iii), and as the original r-twist R. The endpoints of the edges of R are all to the right of the endpoints of the r-twist T. Thus, the range over which the r-rainbow R spans, includes the complete range over which T spans. To this r-rainbow R the three cases can again be applied. We need this observation for combining the cases later.

Observation 5.11. *Applying Case (ii) to an r-twist $T = (V_T, E_T)$, we obtain an r-rainbow R, whose edges have the same start points as the edges from T, and endpoints to the right of the endpoints from E_T.*

Lastly, we consider Subcase (iii) of Case (iii), i.e., we assume that starting with the r-rainbow R only Case (iii) occurs several times in a row. Due to symmetry, we only consider right children. Thus, we assume the right children of R's edges to behave according to Case (iii) (see Figure 5.8), and the right children of the resulting twist to behave according to Case (iii) (see Figure 5.12), repeatedly. Specifically, we start with an r-rainbow $R = (V_R, E_R)$ with ordered vertices $a_1 \prec \cdots \prec a_r \prec b_r \prec \cdots \prec b_1$ and edges $E_R = \{(a_i, b_i) \mid 1 \leq i \leq r\}$. For $j = 2, \ldots, k$, let c_i^j be a right child of (a_i, c_i^{j-1}) and let c_i^1 be a right child of (a_i, b_i) with the vertex ordering $c_i^j \prec c_{i'}^{j'}$ for $j < j'$ or $j = j'$ and $i < i'$. As we apply Case (iii) k times, we obtain the topological vertex ordering

$$a_1 \prec \cdots \prec a_r \prec b_r \prec \cdots \prec b_1 \prec c_1^1 \prec \cdots \prec c_r^1 \prec \cdots \prec c_1^k \prec \cdots \prec c_r^k$$

(see Figure 5.12). The resulting pattern contains the r-twists $T_j = (V_j, E_j)$ with ordered vertices $a_1 \prec \cdots \prec a_r \prec c_1^j \prec \cdots \prec c_r^j$ and edges $E_j = \{(a_i, c_i^j) \mid 1 \leq i \leq r\}$ for $j = 1, \ldots, k$. In sum, these twists T_j form the edge pattern $P = (V, E, \prec)$ with $V = \bigcup_{j=1}^k V_j$ and $E = \bigcup_{j=1}^k E_j$.

Lemma 5.12. *The edge pattern P contains a k-nesting $\lfloor r/k \rfloor$-twist.*

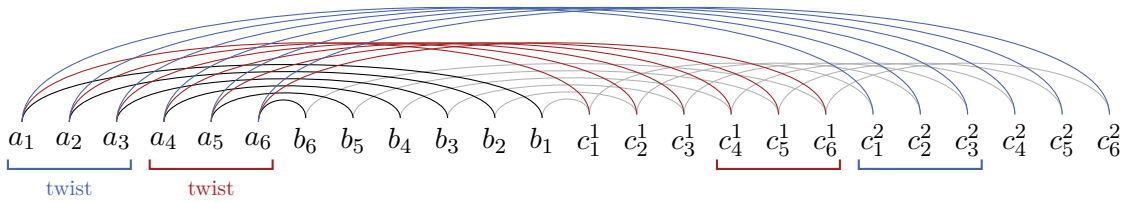


Figure 5.12: A 6-rainbow (black), onto whose edges right children are stacked accordingly to Case (iii), i.e., the children are to the right of their parent's rainbow, such that the edges from their left parents to them (red) pairwise cross. Onto this resulting twist, right children are stacked, again as in Case (iii), such that the edges from left parents to the right children (blue) pairwise cross. The left half of the blue twist nests with the right half of the red twist, such that they form a 2-crossing 3-rainbow.

Proof. We show that each twist T_j contains a twist T'_j with $\lfloor r/k \rfloor$ edges, such that those smaller twists T'_j pairwise nest (see Figure 5.12). Let $T'_j = (V'_j, E'_j)$ be this $\lfloor r/k \rfloor$-twist with ordered vertices

$$a_{(k-j)\lfloor r/k \rfloor+1} \prec a_{(k-j)\lfloor r/k \rfloor+2} \prec \cdots \prec a_{(k-j+1)\lfloor r/k \rfloor} \prec$$

$$c^j_{(k-j)\lfloor r/k \rfloor+1} \prec c^j_{(k-j)\lfloor r/k \rfloor+2} \prec \cdots \prec c^j_{(k-j+1)\lfloor r/k \rfloor}$$

and the respective edges $E'_j = E_j \cap (V'_j \times V'_j)$. Intuitively, each of the r-twists T_j is split into k parts of the same size from left to right, where we only consider the $(k-j+1)$-th part of each twist T_j. Now, we consider two of these $\lfloor r/k \rfloor$-twists T'_j and T'_l with $j < l$ and show that they nest. Since $(k-l+1)\lfloor r/k \rfloor < (k-j)\lfloor r/k \rfloor + 1$, we have the vertex ordering

$$a_{(k-l)\lfloor r/k \rfloor+1} \prec \cdots \prec a_{(k-l+1)\lfloor r/k \rfloor} \prec a_{(k-j)\lfloor r/k \rfloor+1} \prec a_{(k-j+1)\lfloor r/k \rfloor}.$$

Moreover, the vertex ordering is $a_i \prec c^j_{i'} \prec c^l_{i''}$ for each $1 \leq i, i', i'' \leq r$. Thus, the start points of edges in T'_l are all to the left of the start points of edges in T'_j, which are all to the left of the endpoints of edges in T'_j, which are all to the left of endpoints in T'_l. Therefore, these $\lfloor r/k \rfloor$-twists T'_j pairwise nest for $j = 1, \ldots, k$, and form a k-nesting $\lfloor r/k \rfloor$-twist. □

Again, it follows that this case cannot occur arbitrarily often in a row. Similar as for Case (ii), the edges of the newly obtained r-twist T' have the same start points as the edges from the original rainbow R and the edges from the twists obtained in previous steps. Thus, the range over which T' spans includes the complete range of all previous twists and of the original rainbow R. This is an important observation for combining the three cases later.

Observation 5.13. *Applying Case (iii) to an r-twist or an r-rainbow $P = (V_P, E_P)$, we obtain a new r-twist T, whose edges have the same start points as the edges in E_P, and endpoints to the right of the endpoints from E_P.*

After having considered all cases separately, we are finally ready to combine them. Now, we assume that the three cases occur alternately in an arbitrary order. Here, we show that by applying Case (iii) very often – not necessarily in direct succession – we obtain a large nesting twist that forces the mixed page number to become large. To show that Case (iii) does not necessarily have to occur several times in direct succession to result in a large nesting twist, we need to show that the resulting nesting twist (see Lemma 5.12) is not affected by Cases (i) and (ii). That the nesting twist is not affected by Case (ii),

is visualised in Figure 5.13. Furthermore, we show that Case (iii) certainly occurs very often, even if not in direct succession. For this, we show that Case (i) does not affect the crossing rainbow resulting from Case (ii) (see Lemma 5.7). Thus, Cases (i) and (ii) may not even alternately occur arbitrarily often in a row. Hence, Case (iii) must occur in between repeatedly.

Specifically, we start with an r_0-rainbow P_0. For $i = 1, \dots, k$, we apply one of our three cases to the edge pattern P_{i-1} in each step to obtain a new edge pattern P_i, where P_i is either a rainbow or a twist. In step i, P_i is called the *active edge pattern*. Here, the size r_i of these patterns P_i becomes smaller with every step, i.e., $r_{i-1} \geq r_i$ for each $i = 1, \dots, k$. That is, for one thing because we ignore edges and children that do not behave accordingly to the respective case as already mentioned, and secondly because after applying Case (i), we obtain only a rainbow or twist of half the size (see Observation 5.6 and Observation 5.10). However, we assume r_0 to be sufficiently large, such that r_k still is a very large integer. We do not specify an exact value for r_0 here, as well as for the number k of steps.

We say that the start points (or endpoints) of edges of two edge patterns P_i, P_j with $i < j$ are *approximately at the same place* if for any two start points (or endpoints) of P_j it holds that one of them is also start point (or endpoint) of an edge from P_i or there is another start point (or endpoint) x from P_i with $u \prec x \prec v$ or $v \prec x \prec u$. Intuitively, this means that in each subset of P_j's range, there is at least the same number of vertices from P_i (or one less) (see Figure 5.14). By Observation 5.6 it holds that after applying Case (i) to an r-rainbow $R = (V_R, E_R)$ with ordered vertices $a_1 \prec \cdots \prec a_r \prec b_r \prec \cdots \prec b_1$ and edges $E_R = \{(a_i, b_i) \mid 1 \leq i \leq r\}$, we obtain an $r/2$-rainbow $R' = (V'_R, E'_R)$ with ordered vertices $a'_1 \prec \cdots \prec a'_{r/2} \prec b'_{r/2} \prec \cdots \prec b'_1$ and edges $E'_R = \{(a'_i, b'_i) \mid 1 \leq i \leq r/2\}$, for which it holds that $a_{2i-1} \prec a'_i \prec a_{2i+1}$ and $b_{2i+1} \prec b'_i \prec b_{2i-1}$ for each $i = 1, \dots, r/2$. Consider two vertices $a'_i, a'_{i+1} \in V'_R$ that are neighbours in the $r/2$-rainbow R' with respect to $\prec$. It holds that $a'_i \prec a_{2i+1} = a_{2(i+1)-1} \prec a'_{i+1}$, i.e., there is one start point of R between any two start points of R'. The same also applies to endpoints. By Observation 5.10 it follows that the same also holds if we start with a twist. If we apply Case (ii) or Case (iii) to a rainbow or twist P, the start points of the newly obtained rainbow or twist are exactly the same as the start points of P (see Observation 5.8, Observation 5.11, and Observation 5.13). Particularly, they are also approximately at the same place. If we apply Case (i) more than once, we will have to ignore again half of the edges in order to keep the property that the start points are all approximately at the same place. To compensate this, we need to choose the value of r_0 accordingly large. Here, we will not go into more detail. However after ignoring those edges, it follows that if the start points (or endpoints) of P_i are approximately at the same place as the start points (or endpoints) of P_j, and the start points (or endpoints) of P_j are approximately at the same place as the start points (or endpoints) of P_l, then the same applies to P_i and P_l for $i < j < l$, which we will use in the following. Later, we will also need that arguments used in the proofs of Lemma 5.7 and Lemma 5.12 are applicable for patterns, where the start points of the rainbows or twists are not exactly the same but approximately at the same place. Here, we will also not go into more detail.

Now, we first show that the nesting twist that results from several applications of Case (iii) is not affected by applications of Cases (i) and (ii) in between, which is recorded in the lemma below. Since we always start again with a rainbow similar as the original rainbow after having applied Case (i) or Case (ii), we may assume without loss of generality that we apply Case (iii) in the first step.

Lemma 5.14. *Let P_0 be the first active edge pattern, which is an r_0-rainbow. If we apply the three cases k times in any order to the respective active edge pattern so that Case (iii) is applied in the first step and at least t times in total, we obtain a t-nesting $\lfloor r_k/t \rfloor$-twist.*

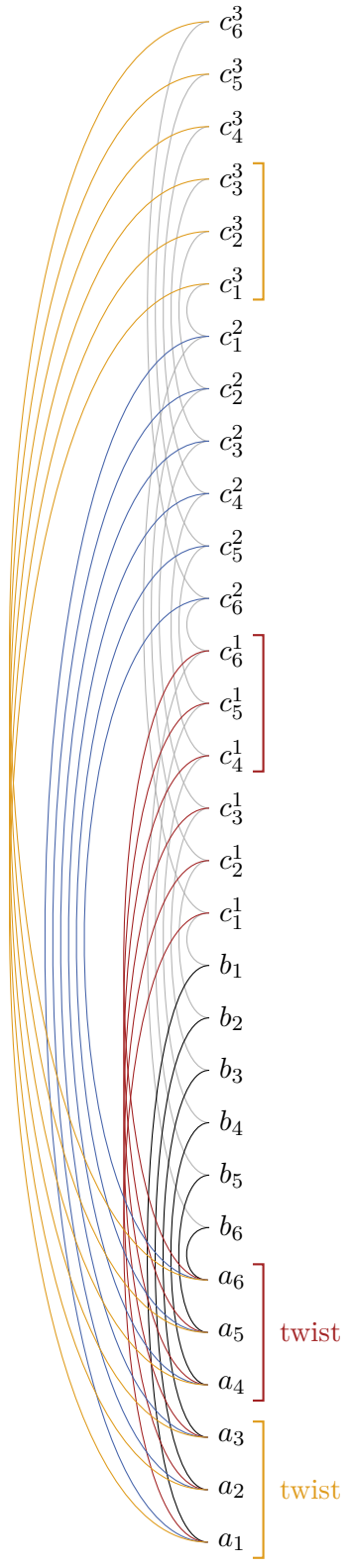


Figure 5.13: A 6-rainbow (black), onto whose edges right children have been stacked repeatedly. The right children of the rainbow's edges behave accordingly to Case (iii) (red). In turn, their children behave accordingly to Case (ii) (blue), and their children again behave accordingly to Case (iii) (orange). The two twists (red and orange) form a 2-nesting 3-twist. In particular, this nesting twist is not affected by Case (ii) in between.

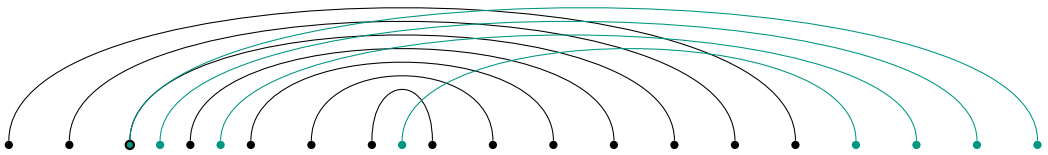


Figure 5.14: Two rainbows, whose start points are approximately at the same place. As the green rainbow is smaller than the black rainbow, there is at least one black start point between any two green start points or one of the green start points is also a black start point.

Proof. We assume that we have already applied the three cases l times in a fixed order σ_l for a fixed $l \in \{1, \dots, k\}$, where we have applied Case (iii) $t(\sigma_l)$ times in total, firstly in step 1 and lastly in step $f(\sigma_l)$ for $t(\sigma_l), f(\sigma_l) \leq l$. Then, we show per induction over l that the obtained edge patterns P_i for $i = 1, \dots, l$ altogether contain $t(\sigma_l)$ large twists of size at least $r_{f(\sigma_l)}$ with start points approximately at the same place and endpoints separated from each other. We call this edge pattern $T^*_{t(\sigma_l),f(\sigma_l)}$. To show that this pattern $T^*_{t(\sigma_l),f(\sigma_l)}$ is obtained, we show that with each application of Case (iii) a new twist is added with start points approximately at the same place as for previous ones and endpoints to the right, even if Cases (i) and (ii) were applied in between. Then, the edge pattern $T^*_{t(\sigma_l),f(\sigma_l)}$ only consists of twists obtained by application of Case (iii). A similar edge pattern with exactly the same start points for all twists is illustrated in Figure 5.12. Similar arguments as in the proof of Lemma 5.12 lead to the fact that $T^*_{t(\sigma_l),f(\sigma_l)}$ contains a $t(\sigma_l)$-nesting $\lfloor r_{f(\sigma_l)}/t(\sigma_l) \rfloor$-twist. Here, we split the $r_{f(\sigma_l)}$-twists into $t(\sigma_l)$ parts of the same size and consider the leftmost part of the twist with the rightmost endpoints, the second leftmost part of the twist with the second rightmost endpoints, and so forth. Then, those $\lfloor r_{f(\sigma_l)}/t(\sigma_l) \rfloor$-twists pairwise nest (see Figure 5.12). Furthermore, we show in each step that the active pattern P_l includes approximately the complete range over which $T^*_{t(\sigma_l),f(\sigma_l)}$ spans. In detail, we show that the start points of edges in P_l are approximately at the same place as the start points of edges in $T^*_{t(\sigma_l),f(\sigma_l)}$ and that the endpoints of edges in P_l are approximately at the same place or to the right of the endpoints of edges in $T^*_{t(\sigma_l),f(\sigma_l)}$.

For $l = 1$, we have already applied Case (iii) once, where we obtained the r_1-twist P_1 (see Figure 5.8). If we apply Case (iii) again, we obtain another r_2-twist P_2 (see Figure 5.12). Together with P_1, there are two twists of size at least r_2 with the same start points and separated endpoints, i.e., they form $T^*_{2,2}$. Note that in this case after increasing l it holds that $t(\sigma_2) = t(\sigma_1) + 1 = 2 = f(\sigma_1) + 1 = f(\sigma_2)$, i.e., $T^*_{2,2}$ is the required edge pattern. Here, P_2's endpoints are to the right of P_1's endpoints. If we apply Case (i) or Case (ii), the edge pattern $T^*_{1,1}$ consisting of P_1 still suffices, since we still have $t(\sigma_2) = t(\sigma_1) = 1 = f(\sigma_1) = f(\sigma_2)$. In Case (i), the start and endpoints of P_2 are approximately at the same place as the start and endpoints of $T^*_{1,1}$ (see Observation 5.10). In Case (ii) (see Figure 5.11), the start points are approximately at the same place and the endpoints are to the right compared to $T^*_{1,1}$ (see Observation 5.11).

For any fixed $1 < l \leq k$ and any fixed order σ_l of previously applied cases, we assume to have such an edge pattern $T^*_{t(\sigma_l),f(\sigma_l)}$ after l steps, where Case (iii) occurred $t(\sigma_l) \leq l$ times, lastly in step $f(\sigma_l) \leq l$. Further, we assume that $T^*_{t(\sigma_l),f(\sigma_l)}$'s range in the vertex ordering is included in the range of the active edge pattern P_l. Note that P_l is the rainbow or twist added in step l. Then, we distinguish which of the three cases is applied in step $l + 1$.

First, we assume that Case (i) is applied in step $l + 1$. As we do not apply Case (iii), we still have $t(\sigma_{l+1}) = t(\sigma_l)$ and $f(\sigma_{l+1}) = f(\sigma_l)$, i.e., the required edge pattern $T^*_{t(\sigma_{l+1}),f(\sigma_{l+1})}$ in step $l + 1$ is the same as the edge pattern $T^*_{t(\sigma_l),f(\sigma_l)}$ from step l. By Observation 5.6

and Observation 5.10, it follows that we obtain a twist or rainbow P_{l+1} of half the size of P_l, depending on whether P_l is a twist or a rainbow. The start and endpoints of P_{l+1} are approximately at the same place as the start and endpoints of P_l. It follows from the prerequisite that the endpoints of edges in P_{l+1} are approximately at the same place as the start points of $T^*_{t(\sigma_{l+1}),f(\sigma_{l+1})}$ and that the endpoints of edges in P_{l+1} are approximately at the same place or to the right of the endpoints of $T^*_{t(\sigma_{l+1}),f(\sigma_{l+1})}$.

Next, we assume that Case (ii) is applied in step $l+1$ (see Figure 5.13). Again, as we do not apply Case (iii), we still have $t(\sigma_{l+1}) = t(\sigma_l)$ and $f(\sigma_{l+1}) = f(\sigma_l)$, i.e., the required edge pattern $T^*_{t(\sigma_{l+1}),f(\sigma_{l+1})}$ in step $l+1$ is the same as the edge pattern $T^*_{t(\sigma_l),f(\sigma_l)}$ from step l. By Observation 5.8 and Observation 5.11, it follows that we obtain a rainbow P_{l+1} with the same start points as P_l and endpoints to the right of the endpoints of P_l. As P_l's start points are approximately at the same place as the start points of $T^*_{t(\sigma_{l+1}),f(\sigma_{l+1})}$, the same applies to P_{l+1}. Since P_l's endpoints are approximately at the same place or to the right of the endpoints of $T^*_{t(\sigma_{l+1}),f(\sigma_{l+1})}$, P_{l+1}'s endpoints are to the right.

Finally, we assume that Case (iii) is applied in step $l+1$ (see Figure 5.13). Consider the edge pattern $T^*_{t(\sigma_l),f(\sigma_l)}$ from step l. The start points of edges in P_l are approximately at the same place as the start points of edges in $T^*_{t(\sigma_l),f(\sigma_l)}$, and the endpoints are also approximately at the same place or to the right. By Observation 5.13, it follows that the start points of P_{l+1} are also approximately at the same place and the endpoints are to the right. Therefore, P_{l+1} is an r_{l+1}-twist with start points approximately at the same place as $T^*_{t(\sigma_l),f(\sigma_l)}$ and endpoints to the right. Thus, together with $T^*_{t(\sigma_l),f(\sigma_l)}$, we obtain the pattern $T^*_{t(\sigma_l)+1,l+1}$, i.e., $t(\sigma_l)+1$ twists of size at least r_{l+1} with start points approximately at the same place and endpoints separated from each other. As we have applied Case (iii), it holds that $t(\sigma_{l+1}) = t(\sigma_l)+1$ and $f(\sigma_{l+1}) = l+1$. Therefore, the obtained edge pattern $T^*_{t(\sigma_l)+1,l+1}$ is equivalent to the required edge pattern $T^*_{t(\sigma_{l+1}),f(\sigma_{l+1})}$.
In sum, we showed that we obtain the edge pattern $T^*_{t(\sigma_l),f(\sigma_l)}$ after l steps with the fixed ordering σ_l of the cases for each $l = 1,\dots,k$ if Case (iii) occurs $t(\sigma_l)$ times in σ_l, firstly in step 1 and lastly in step $f(\sigma_l)$. It follows that after k steps, where Case (iii) occurred $t(\sigma_k) = t$ times, firstly in step 1 and lastly in step $f(\sigma_k) \leq k$, we obtain the edge pattern $T^*_{t,f(\sigma_k)}$. Since particularly $r_k \leq r_{f(\sigma_k)}$, this edge pattern contains a t-nesting $\lfloor r_k/t \rfloor$-twist. □

Now, we have shown that several applications of Case (iii) result in a large nesting twist, even if they are not in direct succession. Finally, we only need to show that Case (iii) certainly occurs very often in order to avoid a large crossing rainbow, if we choose the number k of steps large enough.

Lemma 5.15. *Let P_0 be the first active edge pattern, which is an r_0-rainbow. If we apply the Cases (i) and (ii) k times in any order to the respective active edge pattern so that Case (ii) is applied at least t times, we obtain a $(t+1)$-crossing $\lfloor r_k/(t+1) \rfloor$-rainbow.*

Proof. Again, we assume that we have already applied Cases (i) and (ii) l times in a fixed order σ_l for a fixed $l \in \{0,\dots,k\}$. Thereby, we further assume that Case (ii) has been applied $t(\sigma_l)$ times in total, lastly in step $f(\sigma_l)$ for $t(\sigma_l), f(\sigma_l) \leq l$. Then, we show per induction over l that the obtained edge patterns P_i for $i = 1,\dots,l$ altogether contain $t(\sigma_l)+1$ rainbows of size at least $r_{f(\sigma_l)}$ with start points approximately at the same place and endpoints separated from each other. We call this edge pattern $R^*_{t(\sigma_l)+1,f(\sigma_l)}$. To show that this pattern $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ is obtained, we show that with each application of Case (ii) a new rainbow is added with start points approximately at the same place as for previous ones and endpoints to the right, even if Case (i) was applied in between.

Then, the edge pattern $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ only consists of rainbows obtained by application of Case (ii). A similar edge pattern, with exactly the same start points for all rainbows, is illustrated in Figure 5.9. With similar arguments as in the proof of Lemma 5.7 it follows that $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ contains a $(t(\sigma_l)+1)$-crossing $\lfloor r_{f(\sigma_l)}/(t(\sigma_l)+1)\rfloor$-twist. Here, we split the $r_{f(\sigma_l)}$-rainbows into $t(\sigma_l)+1$ parts of the same size and consider the outermost part of the rainbow with the leftmost endpoints, the second outermost part of the rainbow with the second leftmost endpoints, and so forth. Then, those $\lfloor r_{f(\sigma_l)}/(t(\sigma_l)+1)\rfloor$-rainbows pairwise cross (see Figure 5.9). Further, we show in each step that the active pattern P_l includes approximately the complete range over which $R^*_{t(\sigma_l),f(\sigma_l)}$ spans. More detailed, we show that the start points of edges in P_l are approximately at the same place as the start points of edges in $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ and that the endpoints of edges in P_l are approximately at the same place or to the right of the endpoints of edges in $R^*_{t(\sigma_l)+1,f(\sigma_l)}$.

For $l=0$, we start with the rainbow P_0. If we apply Case (ii) in the first step, we obtain an r_1-rainbow P_1. Together with the original r_0-rainbow P_0, they form the edge pattern $R^*_{2,1}$, i.e., two rainbows of size at least r_1 with the same start points and endpoints separated from each other. Note that in this case after increasing l it holds that $t(\sigma_1)=t(\sigma_0)+1=1=f(\sigma_0)+1=f(\sigma_1)$, so $R^*_{2,1}$ is the required edge pattern. Here, the endpoints of P_1 are to the right of the endpoints of $R^*_{2,1}$. If we apply Case (i), the original r_0-rainbow P_0 already forms the required pattern $R^*_{1,0}$, since we still have $t(\sigma_1)=t(\sigma_0)=0=f(\sigma_0)=f(\sigma_1)$. In this case, P_1 has start and endpoints approximately at the same place as $R^*_{1,0}$.

For any fixed $l>0$ and any fixed ordering σ_l of previously applied cases, we assume that the edge patterns P_i altogether contain the edge pattern $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ for $i=1,\dots,l$, where Case (ii) occurred $t(\sigma_l)\leq l$ times, lastly in step $f(\sigma_l)\leq l$. Further, we assume that the start points of the active pattern P_l are approximately at the same place as the start points of $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ and that the endpoints of P_l are approximately at the same place or to the right of endpoints of $R^*_{t(\sigma_l)+1,f(\sigma_l)}$. Now, we distinguish which of the two cases is applied in step $l+1$.

First, we assume that Case (i) is applied in step $l+1$. As we do not apply Case (ii), we still have $t(\sigma_{l+1})=t(\sigma_l)$ and $f(\sigma_{l+1})=f(\sigma_l)$, i.e., the required edge pattern $R^*_{t(\sigma_{l+1})+1,f(\sigma_{l+1})}$ in step $l+1$ is the same as $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ from step l. By Observation 5.6, it follows that we obtain a rainbow P_{l+1} of half the size of P_l with the start and endpoints of P_{l+1} approximately at the same place as the start and endpoints of P_l. It follows from the prerequisite that the start points of edges in P_{l+1} are approximately at the same place as the start points of $R^*_{t(\sigma_{l+1})+1,f(\sigma_{l+1})}$ and that the endpoints of edges in P_{l+1} are approximately at the same place or to the right of the endpoints of $R^*_{t(\sigma_{l+1})+1,f(\sigma_{l+1})}$.

Finally, we assume that Case (ii) is applied in step $l+1$. Consider the edge pattern $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ from the last step l. By prerequisite, the start points of edges in P_l are approximately at the same place as the start points of edges in $R^*_{t(\sigma_l)+1,f(\sigma_l)}$, and the endpoints are also approximately at the same place or to the right. By Observation 5.13, it follows that the start points of P_{l+1} are also approximately at the same place and the endpoints are to the right. Therefore, P_{l+1} is an r_{l+1}-rainbow with start points approximately at the same place as $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ and endpoints to the right. Thus, together with $R^*_{t(\sigma_l)+1,f(\sigma_l)}$, we have $t(\sigma_l)+2$ rainbows of size at least r_{l+1} with start points approximately at the same place and endpoints separated from each other. Therefore, we obtain the pattern $R^*_{t(\sigma_l)+2,l+1}$. As we have applied Case (ii), we now have $t(\sigma_{l+1})=t(\sigma_l)+1$ and $f(\sigma_{l+1})=l+1$, i.e., the obtained edge pattern $R^*_{t(\sigma_l)+2,l+1}$ is equivalent to the required pattern $R^*_{t(\sigma_{l+1})+1,f(\sigma_{l+1})}$.

Now, we showed that we obtain the edge pattern $R^*_{t(\sigma_l)+1,f(\sigma_l)}$ after l steps with the fixed ordering σ_l of the cases for each $l=1,\dots,k$, if Case (ii) occurs $t(\sigma_l)$ times, lastly in step

$f(\sigma_l)$. It follows that after k steps, where Case (ii) occurs $t(\sigma_k) = t$ times, lastly in step $f(\sigma_k) \leq k$, we obtain the edge pattern $R^*_{t+1,f(\sigma_k)}$. Since particularly $r_k \leq r_{f(\sigma_k)}$, this edge pattern contains a $(t+1)$-nesting $\lfloor r_k/(t+1) \rfloor$-twist. □

If we only consider Cases (i) and (ii), since Case (i) may not occur arbitrarily often in a row (see Lemma 5.5), Case (ii) occurs repeatedly. Precisely, in order that Case (i) does not occur to often in a row, for each $t > 0$, there is a number of steps k, such that Case (ii) occurs at least t times. By Lemma 5.15, this results in a t-crossing rainbow, if we choose the size r_0 of the original r_0-rainbow P_0 sufficiently large. Thus, in order to avoid this large crossing rainbow, Case (iii) has to occur frequently. However, by Lemma 5.14, this results in a t-nesting twist, if we choose the parameters k and r_0 large enough. It follows that there is no way to avoid both large crossing rainbows and large nesting twists. Therefore, the mixed page number of directed acyclic 2-trees is unbounded.

6. Conclusion

Building on the present work on the mixed page number of undirected graphs and the stack and queue numbers of DAGs, we initiated the study of the mixed page number of planar DAGs. In particularly, we investigated the mixed page number of upward planar graphs and directed acyclic 2-trees.

Before we turned to planar DAGs, we first investigated the mixed page number more generally for edge patterns, i.e., for edge sets with a fixed vertex ordering. In Chapter 2, we investigated two edge patterns that force the mixed page number to become large, namely t-crossing rainbows and t-nesting twists. Similar edge patterns, called t-twist and t-rainbow, are known for the stack and queue numbers. There, the stack and queue numbers only depend on the size of a largest t-twist or t-rainbow, respectively [Dav22, HR92]. However, it is not known whether the mixed page number of an edge pattern depends only on the size of a largest t-crossing rainbow or t-nesting twist, and this question still remains open.

Open question 6.1. *Does the mixed page number of an edge pattern only depend on the size of a largest t-crossing rainbow or t-nesting twist?*

What we could answer is that if there is a function $f(t)$ in the size of a largest t-crossing rainbow or t-nesting twist in an edge pattern P, such that $\mathrm{mpn}(P) \leq f(t)$, then this function f is at least a quadratic function. This contrasts with the stack number being in $\Theta(t \log(t))$ [Dav22] and the queue number being in $\Theta(t)$ [HR92]. The question concerning an upper bound or even a tight upper bound on f remains, nevertheless, unanswered.

Open question 6.2. *If there is a function $f(t)$ in the size t of a largest t-crossing rainbow or t-nesting twist in an edge pattern P, such that* $\mathrm{mpn}(P) \leq f(t)$ *– what is f?*

In Chapter 3 we finally started our investigation of planar DAGs with upward planar graphs. First, we considered very specific graph families, for instance cycles, Kelly graphs, grids, and N-grids. In doing so, we gave a lower bound of 3 on the mixed page number of upward planar graphs and presented a graph, whose mixed page number is strictly smaller than its stack and queue number. One of the most important open questions in the field of linear layouts is whether the stack number of upward planar graphs is bounded by a constant. However, the slightly weaker question, whether the same applies to the mixed page number, is also open.

Open question 6.3. *Is the mixed page number of upward planar graphs bounded by a constant?*

After having considered those specific graph families, we turned to the class of upward planar bipartite graphs in Chapter 4. Open question 6.3 is also unanswered for this subclass of upward planar graphs.

Open question 6.4. *Is the mixed page number of upward planar bipartite graphs bounded by a constant?*

However, we were able to answer this question for another subclass of upward planar bipartite graphs. We showed that the stack, queue, and mixed page numbers of upward planar bipartite graphs $G = (A \cup B, E)$ with all edges oriented from a vertex in A to a vertex in B is bounded by the constant 56. In contrast, we showed that these graph parameters are all unbounded if we choose a fixed vertex ordering, such as "from left to right". Additionally, we gave again a lower bound of 3 on the mixed page number of upward planar bipartite graphs and presented an upward planar bipartite graph with mixed page number strictly smaller than stack and queue number.

In Chapter 5, we investigated another class of planar DAGs. There, we presented a directed acyclic 2-tree with unbounded stack and queue number but bounded mixed page number. Nevertheless, we showed that the mixed page number of general directed acyclic 2-trees is also unbounded.

Bibliography

[ABG+20] Md. Jawaherul Alam, Michael A. Bekos, Martin Gronemann, Michael Kaufmann, and Sergey Pupyrev. Queue Layouts of Planar 3-Trees. *Algorithmica*, 82(9):2564–2585, 2020. doi: 10.1007/s00453-020-00697-4.

[ABG+22a] Md. Jawaherul Alam, Michael A. Bekos, Martin Gronemann, Michael Kaufmann, and Sergey Pupyrev. Lazy Queue Layouts of Posets. *Algorithmica*, 2022. doi: 10.1007/s00453-022-01067-y.

[ABG+22b] 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: 10.1016/j.tcs.2022.07.036.

[ABKM22] Patrizio Angelini, Michael A. Bekos, Philipp Kindermann, and Tamara Mchedlidze. On mixed linear layouts of series-parallel graphs. *Theoretical Computer Science*, 936(10):129–138, 2022. doi: 10.1016/j.tcs.2022.09.019.

[BGR23] Michael A. Bekos, Martin Gronemann, and Chrysanthi Raftopoulou. An Improved Upper Bound on the Queue Number of Planar Graphs. *Algorithmica*, 85(2):544–562, 2023. doi: 10.1007/s00453-022-01037-4.

[BHKM20] Michael A. Bekos, Mirco Haug, Michael Kaufmann, and Julia Männecke. An online framework to interact and efficiently compute linear layouts of graphs. *CoRR*, abs/2003.09642, 2020. doi: 10.48550/arXiv.2003.09642; online version `http://algo.inf.uni-tuebingen.de/linearlayouts`; source code available at `https://github.com/linear-layouts/SAT`.

[BK79] Frank Bernhart and Paul C. Kainen. The book thickness of a graph. *Journal of Combinatorial Theory Series B*, 27(3):320–331, 1979. doi: 10.1016/0095-8956(79)90021-2.

[BKK+20] Michael A. Bekos, Michael Kaufmann, Fabian Klute, Sergey Pupyrev, Chrysanthi Raftopoulou, and Torsten Ueckerdt. Four pages are indeed necessary for planar graphs. *Journal of Computational Geometry*, 11(1):332–353, 2020. doi: 10.20382/jocg.v11i1a12.

[CDD+12] Peter Clote, Stefan Dobrev, Ivan Dotu, Evangelos Kranakis, Danny Krizanc, and Jorge Urrutia. On the page number of RNA secondary structures with pseudoknots. *Journal of Mathematical Biology*, 65(6–7):1337–1357, 2012. doi: 10.1007/s00285-011-0493-6.

[CLR87] Fan R. K. Chung, Frank Thomson Leighton, and Arnold L. Rosenberg. Embedding graphs in books: A graph layout problem with applications to VLSI design. *SIAM Journal on Algebraic and Discrete Methods*, 8(1):33–58, 1987. doi: 10.1137/0608002.

[Dav22] James Davies. Improved bounds for colouring circle graphs. *Proceedings of the American Mathematical Society*, 150(12):5121–5135, 2022. doi: 10.1090/proc/16044.

[dFOdMP95] Hubert de Fraysseix, Patrice Ossona de Mendez, and János Pach. A left-first search algorithm for planar graphs. *Discrete & Computational Geometry*, 13(3):459–468, 1995. doi: 10.1007/BF02574056.

[DPW04] Vida Dujmović, Attila Pór, and David R. Wood. Track Layouts of Graphs. *Discrete Mathematics and Theoretical Computer Science*, 6(2):497–522, 2004. doi: 10.46298/dmtcs.315.

[ES35] Paul Erdős and George Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935. doi: 10.1007/978-0-8176-4842-8_-3.

[FFRV13] 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: 10.7155/jgaa.00292.

[FKM+23] Henry Förster, Michael Kaufmann, Laura Merker, Sergey Pupyrev, and Chrysanthi Raftopoulou. Linear Layouts of Bipartite Planar Graphs. Technical report, 2023. `http://i11www.ira.uka.de/extra/publications/fkmpr-llbpg-23.pdf`.

[FUW21] Stefan Felsner, Torsten Ueckerdt, and Kaja Wille. On the Queue-Number of Partial Orders. In Helen C. Purchase and Ignaz Rutter, editors, *Graph Drawing and Network Visualization*, pages 231–241, Cham, 2021. Springer International Publishing. doi: 10.1007/978-3-030-92931-2_17.

[Gro21] Tim Groß. Local Page Number und local Queue Number von gerichteten azyklischen Graphen. Bachelor thesis, ITI Wagner, Department of Informatics, Karlsruhe Institute of Technology (KIT), October 2021. `https://i11www.iti.kit.edu/_media/teaching/theses/ba-gross-22.pdf`.

[HLR92] 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: 10.1137/0405031.

[HPT99] 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: 10.1137/S0097539795280287.

[HR92] Lenwood S. Heath and Arnold L. Rosenberg. Laying Out Graphs Using Queues. *SIAM Journal on Computing*, 21(5):927–958, 1992. doi: 10.1137/0221055.

[JMU22a] Paul Jungeblut, Laura Merker, and Torsten Ueckerdt. A Sublinear Bound on the Page Number of Upward Planar Graphs. In *Proceedings of the 2022 Annual ACM–SIAM Symposium on Discrete Algorithms (SODA'22)*, pages 963–978. Society for Industrial and Applied Mathematics, January 2022. doi: 10.1137/1.9781611977073.42.

[JMU22b] Paul Jungeblut, Laura Merker, and Torsten Ueckerdt. Directed Acyclic Outerplanar Graphs Have Constant Stack Number, November 2022. doi: 10.48550/ARXIV.2211.04732.

[Kel81] David Kelly. On the dimension of partially ordered sets. *Discrete Mathematics*, 35(1):135–156, 1981. doi: 10.1016/0012-365X(81)90203-X.

[KMU18] Kolja Knauer, Piotr Micek, and Torsten Ueckerdt. The Queue-Number of Posets of Bounded Width or Height. In Therese Biedl and Andreas Kerren, editors, *Graph Drawing and Network Visualization*, pages 200–212, Cham, 2018. Springer International Publishing. doi: 10.1007/978-3-030-04414-5_14.

[MU19] Laura Merker and Torsten Ueckerdt. Local and Union Page Numbers. In Daniel Archambault and Csaba D. Tóth, editors, *Graph Drawing and Network Visualization*, pages 447–459, Cham, 2019. Springer International Publishing. doi: 10.1007/978-3-030-35802-0_34.

[MU20] Laura Merker and Torsten Ueckerdt. The Local Queue Number of Graphs with Bounded Treewidth. In David Auber and Pavel Valtr, editors, *Graph Drawing and Network Visualization*, pages 26–39, Cham, 2020. Springer International Publishing. doi: 10.1007/978-3-030-68766-3_3.

[NP89] Richard Nowakowski and Andrew Parker. Ordered sets, pagenumbers and planarity. *ORDER*, 6:209–218, 1989. doi: 10.1007/BF00563521.

[NP21] Martin Nöllenburg and Sergey Pupyrev. On Families of Planar DAGs with Constant Stack Number, July 2021. doi: 10.48550/ARXIV.2107.13658.

[Ove98] Shannon Overbay. *Generalized Book Embeddings*. PhD thesis, Colorado State University, Department of Mathematics, May 1998. `https://dl.acm.org/doi/book/10.5555/335806`.

[PTV12] Aduri Pavan, Raghunath Tewari, and N. V. Vinodchandran. On the Power of Unambiguity in Logspace. *computational complexity*, 21(4):643–670, 2012. doi: 10.1007/s00037-012-0047-3.

[Pup18] Sergey Pupyrev. Mixed Linear Layouts of Planar Graphs. In Fabrizio Frati and Kwan-Liu Ma, editors, *Graph Drawing and Network Visualization*, pages 197–209, Cham, 2018. Springer International Publishing. doi: 10.1007/978-3-319-73915-1_17.

[Pup20] Sergey Pupyrev. Book Embeddings of Graph Products. *CoRR*, abs/2007.15102, 2020. doi: 10.48550/arXiv.2007.15102.

[Pup23] Sergey Pupyrev. Queue Layouts of Two-Dimensional Posets. In Patrizio Angelini and Reinhard von Hanxleden, editors, *Graph Drawing and Network Visualization*, pages 353–360, Cham, 2023. Springer International Publishing. doi: 10.1007/978-3-031-22203-0_25.

[Ram30] F. P. Ramsey. On a Problem of Formal Logic. *Proceedings of the London Mathematical Society*, s2-30(1):264–286, 1930. doi: 10.1112/plms/s2-30.1.264.

[Ros83] Arnold L. Rosenberg. The Diogenes Approach to Testable Fault-Tolerant Arrays of Processors. *IEEE Transactions on Computers*, C-32(10):902–910, 1983. doi: 10.1109/TC.1983.1676134.

[Tar72] Robert Tarjan. Sorting Using Networks of Queues and Stacks. *Journal of the ACM*, 19(2):341–346, 1972. doi: 10.1145/321694.321704.

[Yan89] Mihalis Yannakakis. Embedding planar graphs in four pages. *Journal of Computer and System Sciences*, 38(1):36–67, 1989. doi: 10.1016/0022-0000(89)90032-9.

[Yan20] Mihalis Yannakakis. Planar graphs that need four pages. *Journal of Combinatorial Theory Series B*, 145:241–263, 2020. doi: 10.1016/j.jctb.2020.05.008.