Eliminating Crossings in Ordered Graphs

Akanksha Agrawal
Indian Institute of Technology Madras, Chennai, India

Sergio Cabello
Faculty of Mathematics and Physics, University of Ljubljana, Slovenia
Institute of Mathematics, Physics and Mechanics, Ljubljana, Slovenia

Michael Kaufmann
Department of Computer Science, Tübingen University, Germany

Saket Saurabh
Institute of Mathematical Sciences, Chennai, India

Roohani Sharma
University of Bergen, Norway

Yushi Uno
Graduate School of Informatics, Osaka Metropolitan University, Sakai, Japan

Alexander Wolff
Universität Würzburg, Germany

Abstract

Drawing a graph in the plane with as few crossings as possible is one of the central problems in graph drawing and computational geometry. Another option is to remove the smallest number of vertices or edges such that the remaining graph can be drawn without crossings. We study both problems in a book-embedding setting for *ordered graphs*, that is, graphs with a fixed vertex order. In this setting, the vertices lie on a straight line, called the *spine*, in the given order, and each edge must be drawn on one of several pages of a book such that every edge has at most a fixed number of crossings. In book embeddings, there is another way to reduce or avoid crossings; namely by using more pages. The minimum number of pages needed to draw an ordered graph without any crossings is its (fixed-vertex-order) *page number*.

We show that the page number of an ordered graph with n vertices and m edges can be computed in $2^m \cdot n^{\mathcal{O}(1)}$ time. An $\mathcal{O}(\log n)$-approximation of this number can be computed efficiently. We can decide in $2^{\mathcal{O}(d\sqrt{k}\log(d+k))} \cdot n^{\mathcal{O}(1)}$ time whether it suffices to delete k edges of an ordered graph to obtain a *d-planar* layout (where every edge crosses at most d other edges) on *one* page. As an additional parameter, we consider the size h of a *hitting set*, that is, a set of points on the spine such that every edge, seen as an open interval, contains at least one of the points. For $h = 1$, we can efficiently compute the minimum number of edges whose deletion yields fixed-vertex-order page number p. For $h > 1$, we give an XP algorithm with respect to $h + p$. Finally, we consider *spine+t-track drawings*, where some but not all vertices lie on the spine. The vertex order on the spine is given; we must map every vertex that does not lie on the spine to one of t tracks, each of which is a straight line on a separate page, parallel to the spine. In this setting, we can minimize in $2^n \cdot n^{\mathcal{O}(1)}$ time either the number of crossings or, if we disallow crossings, the number of tracks.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Fixed parameter tractability; Human-centered computing → Graph drawings; Mathematics of computing → Graph theory

Keywords and phrases Ordered graphs, book embedding, edge deletion, d-planar, hitting set

Digital Object Identifier 10.4230/LIPIcs.SWAT.2024.1

Related Version *Full Version*: `https://arxiv.org/abs/2404.09771`

Funding Funded in part by Science and Engineering Research Board, Startup Research Grant (SRG/2022/000962). Funded in part by the Slovenian Research and Innovation Agency (P1-0297, J1-2452, N1-0218, N1-0285). Funded in part by the EU (ERC, KARST, project no. 101071836).

© Akanksha Agrawal, Sergio Cabello, Michael Kaufmann, Saket Saurabh, Roohani Sharma, Yushi Uno, and Alexander Wolff;
licensed under Creative Commons License CC-BY 4.0
19th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT 2024).
Editor: Hans L. Bodlaender; Article No. 1; pp. 1:1–1:19
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the EU or the ERC. Neither the EU nor the granting authority can be held responsible for them. Partially supported by JSPS KAKENHI grant no. JP17K00017, 20H05964, and 21K11757.

Acknowledgements We thank the organizers of the 2023 Dagstuhl Seminar "New Frontiers of Parameterized Complexity in Graph Drawing", where this work was initiated.

1 Introduction

Many crossings typically make it hard to understand the drawing of a graph, and thus much effort in the area of Graph Drawing has been directed towards reducing the number of crossings in drawings of graphs. In terms of parameterized complexity, several facets of this problem have been considered. For example, there are FPT algorithms that, given a graph G and an integer k, decide whether G can be drawn with at most k crossings [17, 21]. Crossing minimization has also been considered in the setting where each vertex of the given graph must lie on one of two horizontal lines. This restricted version of crossing minimization is an important subproblem in drawing layered graphs according to the so-called *Sugiyama framework* [33]. There are two variants of the problem; either the vertices on both lines may be freely permuted or the order of the vertices on one line is given. These variants are called two-layer and one-layer crossing minimization, respectively. For both, FPT algorithms exist [22, 23]. Zehavi [38] has surveyed parameterized approaches to crossing minimization.

Surprisingly, crossing minimization remains NP-hard even when restricted to graphs that have a planar subgraph with just one edge less [9]. Another way to deal with crossings is to remove a small number of vertices or edges such that the remaining graph can be drawn without crossings. In fact, it is known that vertex deletion to planarity is FPT with respect to the number of deleted vertices [18, 20, 28]. However, the running times of these algorithms depends at least exponentially on the number of deleted vertices. On the kernelization front, there exists an $\mathcal{O}(1)$-approximate kernel for vertex deletion to planarity [19], whereas vertex deletion to outerplanarity is known to admit an (exact) polynomial kernel [13].

In this paper, we focus on another model to cope with the problem of crossing edges, namely *book embeddings*, drawings where the vertices lie on a straight line, called the *spine*, and each edge must be drawn on one of several halfplanes, called *pages*, such that the drawing on each page is crossing-free (planar) or such that each edge has at most a constant number d of crossings (that is, the drawing is d-planar). We consider the variant of the problem where the order σ of the vertices is given and fixed. The minimum number of pages to draw an (ordered) graph without any crossings is its (fixed-vertex-order) *page number*.

In this paper, we study the problem of designing parameterized algorithms, where the possible parameters are the number k of edges to be deleted, the number c of allowed crossings per edge, the number p of pages, and their combinations.

Problem description. Given a graph G, let $V(G)$ denote the vertex set and $E(G)$ the edge set of G. An *ordered graph* (G, σ) consists of a graph G and an ordering of the vertices of G, that is, a bijective map $\sigma\colon V(G) \to \{1, \ldots, |V(G)|\}$. Henceforth, we specify every edge (u, v) of (G, σ) such that $\sigma(u) < \sigma(v)$. For two edges $e = (u, v)$ and $e' = (u', v')$ of an ordered graph (G, σ), we say that e and e' *cross with respect to* σ if their endpoints interleave, that is, if $\sigma(u) < \sigma(u') < \sigma(v) < \sigma(v')$ or if $\sigma(u') < \sigma(u) < \sigma(v') < \sigma(v)$. The ordered graph models the scenario where the vertices of G are placed along a horizontal line in the given order σ and all the edges are drawn above the line using curves that cross as few times as possible. Whenever e and e' cross with respect to σ, their curves must intersect. Whenever

e and e' do *not* cross with respect to σ, their curves can be drawn without intersections; for example, we may use halfcircles. In this setting, we get a drawing such that two edges of G cross precisely if and only if they cross with respect to σ. Given a positive integer d, we say that an ordered graph (G, σ) is *d-planar* if every edge in G is crossed by at most d other edges (where 0-planar simply means planar).

In this paper, we focus on fast parameterized algorithms for the following problem.

Edge Deletion to p-Page d-Planar

Input: An ordered graph (G, σ) and positive integers k, p, and d.
Parameters: k, p, d
Question: Does there exist a set S of at most k edges of G such that $(G - S, \sigma)$ is p-page d-planar?

We stress that we view p and d, though they appear in the problem name, not as constants, but as parameters.

Related work. Given an ordered graph (G, σ), its *conflict graph* $H_{(G,\sigma)}$ is the graph that has a vertex for each edge of G and an edge for each pair of crossing edges of G. Note that $H_{(G,\sigma)}$ is a *circle graph*, that is, the intersection graph of chords of a circle, because two chords in a circle intersect if and only if their endpoints interleave.

We can express Edge Deletion to 1-Page d-Planar as the problem of deleting from $H_{(G,\sigma)}$ a set of at most k vertices such that the remaining graph has maximum degree at most d. For general graphs, this problem is called Vertex Deletion to Degree-d [31]; it admits a quadratic kernel [15, 37].

Testing whether (G, σ) has (fixed-vertex-order) page number p (without any edge deletions) is equivalent to the p-colorability of the conflict graph $H_{(G,\sigma)}$. For $p = 2$, it suffices to test whether the conflict graph $H_{(G,\sigma)}$ is bipartite. An alternative approach, discussed by Masuda, Nakajima, Kashiwabara, and Fujisawa [29], is to add to G a cycle connecting the vertices along the spine in the given order, and then test for planarity. Another possibility is to use 2-Sat. For $p = 4$, Unger [34] showed that the problem is NP-hard. For $p = 3$, he [35] claimed an efficient solution, but recently his approach was shown to be incomplete [4].

Edge Deletion to p-Page Planar is the special case where $d = 0$; it can be interpreted as deletion of as few vertices as possible in the conflict graph $H_{(G,\sigma)}$ to obtain a p-colorable graph. For $p = 1$, the problem can be solved by finding a maximum independent set in a circle graph, which takes linear time [16, 30, 36]; see Lemma 3 in Section 2. Edge Deletion to 2-Page Planar can be phrased as Odd Cycle Transversal in the conflict graph, which means that it is FPT with respect to the number of edges that must be deleted [32]. The case $p = 2$ can also be modeled as a (geometric) special case of Almost 2-Sat (variable), which can be solved in $2.3146^k \cdot n^{\mathcal{O}(1)}$ time, where k is the number of variables that need to be deleted so that the formula becomes satisfiable [27, Corollary 5.2].

Masuda et al. [29] showed that the problem Fixed-Order 2-Page Crossing Number is NP-hard. In this problem, we have to decide, for each edge of the given ordered graph (G, σ), whether to draw it above or below the spine, so as to minimize the number of crossings.

Bhore, Ganian, Montecchiani, and Nöllenburg [7] studied the fixed-vertex-order page number and provide an algorithm to compute it with running time $2^{\mathcal{O}(\mathrm{vc}^3)}n$, where vc is the vertex cover number of the graph. They also proved that the problem is fixed-parameter tractable parameterized by the pathwidth (pw) of the *ordered* graph, with a running time of $\mathrm{pw}^{\mathcal{O}(\mathrm{pw}^2)}\, n$. Note that the pathwidth of an ordered graph is in general not bounded by the vertex cover number [7]. This has been improved by Liu, Chen, Huang, and Wang [26]

Table 1 New and known results concerning Edge Deletion to p-Page d-Planar.

k	p	d	add. param.	ref.	result (runtime, ratio, or kernel size)	
0	min	0	–	Cor. 2	EXP:	$2^m n^{\mathcal{O}(1)}$
0	min	0	–	Thm. 4	approx:	ratio $\mathcal{O}(\log n)$
0	min	param.	–	Cor. 5	approx:	ratio $\mathcal{O}((d+1)\log n)$
param.	1	param.	–	Thm. 6	FPT:	$2^{\mathcal{O}(d\sqrt{k}\log(d+k))} \cdot n^{\mathcal{O}(1)}$
min	param.	0	–	Sect. 4	EXP:	$4^m n^{\mathcal{O}(1)}$
min	param.	0	$h=1$	Thm. 9	P:	$\mathcal{O}(m^3 \log n \log\log p)$
min	param.	0	h	Thm. 12	XP:	$\mathcal{O}(m^{(4h-2)p+3} \log n \log\log p)$
0	–	min	t	Thm. 14	EXP:	$2^n n^{\mathcal{O}(1)}$
0	–	min	$\min t$	Cor. 15	EXP:	$2^n n^{\mathcal{O}(1)}$
param.	1	param.	–	[15, 37]	kernel:	quadratic
0	≥ 4	0	–	[34]	NPC.	
0	≤ 2	0	–	folklore	P:	linear time; e.g., via 2-Sat
min	1	0	–	e.g., [16]	P:	linear time
param.	2	0	–	[32]	FPT:	Odd Cycle Transversal
0	2	min	–	[29]	NPC.	
0	min	0	vc	[25]	FPT:	$(d+2)^{\mathcal{O}(\mathrm{vc}^3)} n$
0	min	0	pw	[25]	FPT:	$(d+2)^{\mathcal{O}(\mathrm{pw}^2)} n$
0	param.	cr	pw	[26]	FPT:	$n \cdot (\mathrm{cr}+2)^{\mathcal{O}(\mathrm{pw}^2)}$
0	param.	param.	pw	[26]	FPT:	$2^{\mathcal{O}(\mathrm{pw}^2)} n$; no poly. pw-kernel

to $2^{\mathcal{O}(\mathrm{pw}^2)} n$. They also showed that the problem does not admit a polynomial kernel if parameterized only by pw (unless NP $\subseteq$ coNP/poly). Moreover, they gave an algorithm that checks in $(\mathrm{cr}+2)^{O(\mathrm{pw}^2)} n$ time whether a graph with n vertices and pathwidth pw can be drawn on a given number of pages with at most cr crossings in total.

Liu, Chen and Huang [25] considered the problem Fixed-Order Book Drawing with bounded number of crossings per edge: decide if there is a p-page book-embedding of G such that the maximum number of crossings per edge is upper-bounded by an integer d. This problem was posed by Bhore et al. [7]. Liu et al. showed that this problem, when parameterized by both the maximum number d of crossings per edge and the vertex cover number vc of the graph, admits an algorithm running in $(d+2)^{\mathcal{O}(\mathrm{vc}^3)} n$ time. They also showed that the problem, when parameterized by both d and the pathwidth pw of the vertex ordering, admits an algorithm running in $(d+2)^{\mathcal{O}(\mathrm{pw}^2)} n$ time.

All these problems can be considered also in the setting where we can choose the ordering of the vertices along the spine; see, for instance, [7, 11].

Our contribution. For an overview over our results and known results, see Table 1. First, we show that the fixed-vertex-order page number of an ordered graph with m edges and n vertices can be computed in $2^m \cdot n^{\mathcal{O}(1)}$ time; see Section 2. We use subset convolution [8]. Alternatively, given a budget p of pages, we can compute a p-page book embedding with the minimimum number of crossings. By combining the greedy algorithm for Set Cover with an efficient algorithm for Maximum Independent Set in circle graphs [16, 30, 36], we obtain an efficient $\mathcal{O}((d+1)\log n)$-approximation algorithm for the fixed-vertex-order d-planar page number.

Second, we tackle Edge Deletion to 1-Page d-Planar; see Section 3. We show how to decide in $2^{\mathcal{O}(d\sqrt{k}\log(d+k))} \cdot n^{\mathcal{O}(1)}$ time whether deleting k edges of an ordered graph suffices to obtain a d-planar layout on *one* page. Note that our algorithm is subexponential in k.

Third, we consider the problem Edge Deletion to p-Page Planar; see Section 4. As an additional parameter, we consider the size h of a *hitting set*, that is, a set of points on the spine such that every edge, seen as an open interval, contains at least one of the points. For $h = 1$, we can efficiently compute the smallest set of edges whose deletion yields fixed-vertex-order page number p. For $h > 1$, we give an XP algorithm with respect to $h + p$.

Finally, we consider *spine+t-track drawings*; see Section 5. In such drawings, some but not all vertices lie on the spine. The vertex order on the spine is again given, but now we must map every vertex that does not lie on the spine to one of t tracks, each of which is a straight line on a separate page, parallel to the spine. Using subset convolution, we can minimize in $2^n \cdot n^{\mathcal{O}(1)}$ time either the number of crossings or, if we disallow crossings, the number of tracks.

We close with some open problems; see Section 6.

2 Computing the Fixed-Vertex-Order Page Number

Let (G, σ) be an ordered graph, and let p be a positive integer. In this section, we consider *p-page book-embeddings* of (G, σ): the vertices of G are placed on a *spine* ℓ according to σ, there are p *pages* (halfplanes) sharing ℓ on their boundary, and for each edge we have to decide on which page it is drawn. The aim is to minimize the total number of crossings for a given number of pages, or minimize the number of pages to attain no crossings; see Figure 1.

Let $\mathrm{cr}_p(G, \sigma)$ be the minimum number of crossings over all possible assignments of the edges of $E(G)$ to the p pages. As discussed in the introduction, we can decide in linear time whether $\mathrm{cr}_2(G, \sigma) = 0$, but in general, computing $\mathrm{cr}_2(G, \sigma)$ is NP-hard [29]. The fixed-vertex-order page number of (G, σ) is the minimum p such that $\mathrm{cr}_p(G, \sigma) = 0$.

▶ **Theorem 1.** *Given an ordered graph (G, σ) with n vertices and m edges, and a positive integer p, we can compute the values $\mathrm{cr}_1(G, \sigma), \ldots, \mathrm{cr}_p(G, \sigma)$ in $2^m \cdot n^{\mathcal{O}(1)}$ time. In particular, given a budget p of pages, we can compute a p-page book embedding with the minimum number of crossings within the given time bound.*

Proof. Consider a fixed-vertex-order graph $((V, E), \sigma)$ with n vertices and m edges. We need to consider only the case $p < m$ because, for $p \geq m$, it obviously holds that $\mathrm{cr}_p((V, E), \sigma) = 0$.

First note that, for any fixed $F \subseteq E$, we can easily compute $\mathrm{cr}_1((V, F), \sigma)$ in $\mathcal{O}(|F|^2) = \mathcal{O}(m^2)$ time by checking the order of the endpoints of each pair of edges. It follows that we can compute $\mathrm{cr}_1((V, F), \sigma)$ for all subsets $F \subseteq E$ in $2^m \cdot n^{\mathcal{O}(1)}$ time.

For every $q > 1$ and every $F \subseteq E$, we have the recurrence

$$\mathrm{cr}_q((V, F), \sigma) \;=\; \min \{\mathrm{cr}_1((V, F'), \sigma) + \mathrm{cr}_{q-1}((V, F \setminus F'), \sigma) \mid F' \subseteq F\}\,. \tag{1}$$

Here, $F' \subseteq F$ corresponds to the edges that in the drawing go to one page, and thus $F \setminus F'$ goes to the remaining $q - 1$ pages, where we can optimize over all choices of $F' \subseteq F$.

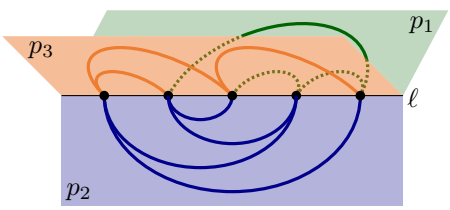


Figure 1 A 3-page book embedding of K_5 with fixed vertex order. For each edge, we can choose on which page it is drawn. Note that K_5 cannot be drawn on two pages without crossings.

From the recurrence in Equation (1) we see that, for $q > 1$, the function $F \mapsto \mathrm{cr}_q((V,F),\sigma)$ is, by definition, the *subset convolution* of the functions $F \mapsto \mathrm{cr}_1((V,F),\sigma)$ and $F \mapsto \mathrm{cr}_{q-1}((V,F),\sigma)$ in the $(\min,+)$ ring. Since $\mathrm{cr}_q((V,F),\sigma)$ takes integer values from $\{0,\ldots,m^2\}$ for every q and F, it follows from [8] that one can obtain $\mathrm{cr}_q((V,F),\sigma)$ for all $F \subseteq E$ in $2^m \cdot n^{\mathcal{O}(1)}$ time, for a fixed $q > 1$, assuming that $\mathrm{cr}_1((V,F),\sigma)$ and $\mathrm{cr}_{q-1}((V,F),\sigma)$ are already available. Therefore, we can compute the values $\mathrm{cr}_q((V,F),\sigma)$ for $q \in \{2,\ldots,p\}$ in $2^m \cdot n^{\mathcal{O}(1)}$ time since $p \leq m < n^2$. ◀

▶ **Corollary 2.** *The fixed-vertex-order page number of a graph with n vertices and m edges can be computed in $2^m \cdot n^{\mathcal{O}(1)}$ time.*

▶ **Lemma 3.** *Given an ordered graph (G,σ), we can compute in polynomial time a smallest subset $S \subseteq E(G)$ such that $\mathrm{cr}_1(G-S,\sigma) = 0$.*

Proof. Consider the *conflict graph* $H_{(G,\sigma)}$ of (G,σ), already defined in the introduction. Note that $H_{(G,\sigma)}$ is a circle graph. Therefore, a largest independent set in $H_{(G,\sigma)}$ corresponds to a largest subset F of edges with $\mathrm{cr}_1((V,F),\sigma) = 0$, which corresponds to a minimum set $S \subseteq E(G)$ such that $\mathrm{cr}_1(G-S,\sigma) = 0$. Finally, note that a largest independent set in circle graphs can be computed in polynomial time [16, 30, 36]. ◀

▶ **Theorem 4.** *We can compute an $\mathcal{O}(\log n)$-approximation to the fixed-vertex-order page number of a graph with n vertices in polynomial time.*

Proof. Let $((V,E),\sigma)$ be the given ordered graph, and let OPT be its fixed-vertex-order page number. Define the family $\mathcal{F} = \{F \subseteq E \mid \mathrm{cr}_1((V,F),\sigma) = 0\}$. Consider the Set Cover instance $(E,\mathcal{F})$, where E is the universe and $\mathcal{F} \subseteq 2^E$ is a family of subsets of E. A feasible solution of this Set Cover instance is a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ such that $\bigcup \mathcal{F}' = E$. The task in Set Cover is to find a feasible solution of minimum cardinality.

Each feasible solution $\mathcal{F}'$ to the Set Cover instance $(E,\mathcal{F})$ corresponds to a fixed-vertex-order drawing of (G,σ) with $|\mathcal{F}'|$ pages. Similarly, each fixed-vertex-order drawing of (G,σ) with p pages represents a feasible solution to Set Cover with p sets. In particular, the size of the optimal solution to the Set Cover instance $(E,\mathcal{F})$ is equal to OPT, the fixed-vertex-order page number of (G,σ).

Consider the usual greedy algorithm for Set Cover, which works as follows. Set $E_1 = E$ and $i = 1$. While $E_i \neq \emptyset$, we set F_i to be the element of $\mathcal{F}$ that contains the largest number of edges from E_i, increase i, and set $E_i = E_{i-1} \setminus F_{i-1}$. Let $i^\star$ be the maximum value of i with $E_i \neq \emptyset$. Thus $E_{i^\star+1} = \emptyset$, and the algorithm finishes. It is well known that $i^\star \leq \mathrm{OPT} \cdot \log|E|$; see for example [12, Section 5.4]. Therefore, this greedy algorithm yields an $\mathcal{O}(\log|V|)$-approximation for our problem.

Finally, note that the greedy algorithm can be implemented to run efficiently. Indeed, F_i can be computed from E_i in polynomial time because of Lemma 3, and the remaining computations in every iteration are trivially done in polynomial time. The number of iterations is polynomial because $i^\star \leq |E|$. ◀

▶ **Corollary 5.** *We can compute an $\mathcal{O}((d+1)\log n)$-approximation to the fixed-vertex-order d-planar page number of a graph with n vertices in polynomial time.*

Proof. Consider first an ordered graph (H,σ) that is d-planar if drawn on a single page, with $d > 0$. Let F_d be the subset of $E(H)$ such that each edge in F_c participates in exactly d crossings, and let S_d be a maximal subset of F_d such that no two edges in S_d cross each other. Then, $(H - S_d,\sigma)$ is $(d-1)$-planar because each edge of H has fewer than d crossings, is in S_c, or is crossed by some edge in S_d. It follows by induction that (H,σ) can be embedded in $d+1$ pages without crossings.

Consider now the input ordered graph (G, σ) and let OPT_d be the minimum number of d-planar pages needed for (G, σ). By the argument before applied to each page, we know that the minimum number of planar pages, OPT_0, is at most $(d+1)\,\mathrm{OPT}_d$. Using Theorem 4, we obtain a drawing of (G, σ) without crossings with at most $\mathrm{OPT}_0 \cdot \mathcal{O}(\log n) \leq (d+1)\,\mathrm{OPT}_d \cdot \mathcal{O}(\log n)$ (planar) pages, where $n = |V(G)|$. Such a drawing is of course also d-planar. ◀

3 Edge Deletion to 1-Page d-Planar

The main result of this section is as follows.

▶ **Theorem 6.** *Edge Deletion to 1-Page d-Planar admits an algorithm with running time $2^{\mathcal{O}(d\sqrt{k}\log(d+k))} \cdot n^{\mathcal{O}(1)}$, where n is the number of vertices in the input graph and k is the number of edges to be deleted.*

In other words, we obtain a subexponential fixed-parameter tractable algorithm for Edge Deletion to 1-Page d-Planar parameterized by k, the number of edges to be deleted; note that we consider d to be a constant here (although we made explicit how the running time depends on d). Our algorithm to prove Theorem 6 has two steps. First it branches on edges that are crossed by at least $d + \sqrt{k}$ other edges. When such edges do not exist, we show that the conflict graph $H_{(G,\sigma)}$ has treewidth $\mathcal{O}(d + \sqrt{k})$. This is done by showing that the conflict graph has balanced separators. Finally the bound on the treewidth allows us to use a known (folklore) algorithm [24] for Vertex Deletion to Degree-d whose dependency is singly exponential in the treewidth of $H_{(G,\sigma)}$.

3.1 Branching

Let $\mathtt{cross}_{(G,\sigma)}(e)$ denote the set of edges of G that cross e with respect to σ. We drop the subscript (G, σ) when it is clear from the context. We show that we can use branching to reduce any instance to a collection of instances where each edge e of the graph satisfies $|\mathtt{cross}(e)| < d + \sqrt{k}$. In particular we show the following lemma.

▶ **Lemma 7.** *Let (G, σ, k) be an instance of Edge Deletion to 1-Page d-Planar. There is a $2^{\mathcal{O}(d\sqrt{k}\log(d+k))} \cdot n^{\mathcal{O}(1)}$-time algorithm that outputs $2^{\mathcal{O}(d\sqrt{k}\log(d+k))}$ many instances of Edge Deletion to 1-Page d-Planar $(G_1, \sigma, k_1), \ldots, (G_r, \sigma, k_r)$ such that for each $i \in [r]$, G_i is a $(d + \sqrt{k})$-planar graph, and (G, σ, k) is a Yes-instance of Edge Deletion to 1-Page d-Planar if and only if (G_i, σ, k_i) is a Yes-instance of Edge Deletion to 1-Page d-Planar for some $i \in [r]$.*

Proof. Let e be an edge of G with $|\mathtt{cross}(e)| \geq d + \lceil\sqrt{k}\rceil$. If $|\mathtt{cross}(e)| > d + k$, then e must be deleted, as we cannot afford to keep e and delete enough edges from $\mathtt{cross}(e)$. If $|\mathtt{cross}(e)| \leq d + k$, then either e must be deleted or at least $|\mathtt{cross}(e)| - d$ many edges from $\mathtt{cross}(e)$ must be deleted, so that at most d edges of $\mathtt{cross}(e)$ stay. This results in the following branching rule, where we return an OR over the answers of the following instances:

1. Recursively solve the instance $(G - e, \sigma, k - 1)$. This branch is called the *light* branch.
2. If $|\mathtt{cross}(e)| > d + k$, we do not consider other branches. Otherwise, for each subset X of $\mathtt{cross}(e)$ with $|\mathtt{cross}(e)| - d$ many edges, recursively solve the instance $(G - X, \sigma, k - |X|)$. Each of these branches is called a *heavy* branch.

We are going to show that the recursion tree has $2^{\mathcal{O}(d\sqrt{k}\log(d+k))}$ branches. Note that the number of possible heavy branches at each is node is

$$\binom{|\mathtt{cross}(e)|}{|\mathtt{cross}(e)| - d} = \binom{|\mathtt{cross}(e)|}{d} \leq \binom{d+k}{d} \leq (d+k)^d.$$

To prove the desired upper bound, we interpret the branching tree as follows. First note that, in each node, we have at most $(d+k)^d$ heavy branches. We associate a distinct word over the alphabet $\Sigma = \{0, 1, \ldots, (d+k)^d\}$ to each leaf (or equivalently each root to leaf path) of the recurrence tree. For each node of the recurrence tree, associate a character from Σ with each of its children such that the child node corresponding to the light branch gets the character 0 and the other nodes (corresponding to the heavy branches) get a distinct character from $\Sigma \setminus \{0\}$. Now a word over the alphabet Σ for a leaf ℓ of the recurrence tree is obtained by taking the sequence of characters on the nodes of the root to leaf ℓ path in order. In order to bound the number of leaves (and hence the total number of nodes) of the recurrence tree, it is enough to bound the number of such words. The character 0 is called a *light* label and all other characters are called *heavy* labels. Recall that a light label corresponds to the branch where k drops by 1, while the heavy labels correspond to the branches where k drops by $|\texttt{cross}(e)| - d \geq \sqrt{k}$. This implies that each word (that is associated with the leaf of the recurrence tree) has at most $\sqrt{k}$ heavy labels. In order to bound the number of such words, we first guess the places in the word that are occupied by heavy labels and then we guess the (heavy) labels themselves at these selected places. All other positions have the light label on them and there is no choice left. Hence, the number of such words is upper-bounded by

$$\sum_{i=0}^{\sqrt{k}} \binom{k}{i} ((d+k)^d)^i \;\leq\; \sqrt{k}\binom{k}{\sqrt{k}}((d+k)^d)^{\sqrt{k}} \;=\; 2^{\mathcal{O}(d\sqrt{k}\log(d+k))}.$$

This shows that the number of such words is bounded by $2^{\mathcal{O}(d\sqrt{k}\log(d+k))}$, and hence the number of leaves (and nodes) of the recurrence tree is bounded by $2^{\mathcal{O}(d\sqrt{k}\log(d+k))}$. ◀

3.2 Balanced Separators in the Conflict Graph

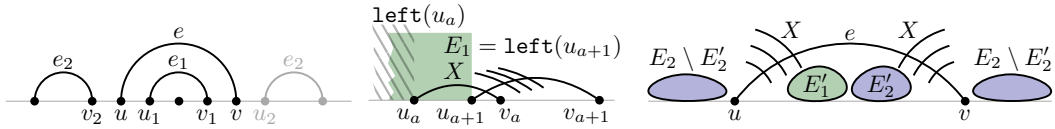
Let (G, σ) be an ordered graph. For any edge $e = (u, v)$ of G, let $\texttt{span}_{(G,\sigma)}(e)$ be the set of all edges $(u', v') \neq e$ of G such that $\sigma(u) \leq \sigma(u') \leq \sigma(v') \leq \sigma(v)$. For example, in Figure 2a, $\texttt{span}(e) = \{e_1\}$. For any vertex w of G, let $\texttt{left}_{(G,\sigma)}(w)$ be the set of all edges (u, v) of G such that $\sigma(u) < \sigma(v) \leq \sigma(w)$. Whenever it is clear from the context, we will drop the subscript (G, σ). We say that an edge e of G is *maximal* if G contains no edge e' such that $e \in \texttt{span}(e')$.

▶ **Lemma 8** (Balanced Separator in the Conflict Graph). *If (G, σ) is an ordered d-planar graph, then G contains a set X of at most $3(d+1)$ edges such that $E(G) \setminus X = E_1 \cup E_2$, $E_1 \cap E_2 = \emptyset$, $|E_1| \leq 2m/3$, $|E_2| \leq 2m/3$, and no edge $e_1 \in E_1$ crosses an edge $e_2 \in E_2$ with respect to σ.*

The proof in the full version of this paper [1] considers three cases depending on the spans of the edges of G; see Figure 2. Either there exists an edge $e = (u, v) \in E(G)$ such that $m/3 \leq |\texttt{span}(e)| \leq 2m/3$ (case 1), or for every edge $e \in E(G)$, it holds that $|\texttt{span}(e)| \leq m/3$ (case 2), or there exists an edge $e \in E(G)$ such that $|\texttt{span}(e)| > 2m/3$ (case 3). Note that every outer d-planar graph contains a balanced *vertex* separater of size at most $2d + 3$ [10].

3.3 Proof of Theorem 6

We now need to establish a relation between the treewidth of the graph and the size of a balanced separator in it. For this we use the result of Dvořák and Norin [14] that shows a linear dependence between the treewidth and the *separation number* of a graph: the separation number of a graph is the smallest integer s such that every subgraph of the given



(a) Case 1. **(b)** Case 2. **(c)** Case 3.

Figure 2 Case distinction for the proof of Lemma 8.

graph has a balanced separator of size at most s. A balanced separator in a graph H is a set of vertices B such that the vertex set of $H - B$ can be partitioned into two parts V_1 and V_2 such that $E(V_1, V_2) = \emptyset$ and $|V_1|, |V_2| \leq 2|V(H)|/3$. In other words, they show that if the separation number of the graph is s, then the treewidth of such a graph is $\mathcal{O}(s)$.

Recall that (G, σ, k) is an instance of Edge Deletion to 1-Page d-Planar. By Lemma 8, if the ordered graph (G, σ) is $(d + \sqrt{k})$-planar, then the conflict graph $H_{(G,\sigma)}$ has a balanced separator of size at most $3(d + \sqrt{k} + 1)$. Thus, due to the result of Dvořák and Norin [14], the treewidth of $H_{(G,\sigma)}$ is $\mathcal{O}(d + \sqrt{k})$.

Given a graph with N vertices and treewidth tw, one can compute, in $(d + 2)^{\mathrm{tw}} \cdot N^{\mathcal{O}(1)}$ time, the smallest set of vertices whose deletion results in a graph of degree at most d [24]. Applying this result to the conflict graph $H_{(G,\sigma)}$, which has at most $|V(G)|^2 = n^2$ vertices and treewidth $\mathcal{O}(d + \sqrt{k})$, we conclude that Edge Deletion to 1-Page d-Planar can be solved in $2^{\mathcal{O}((d+\sqrt{k}) \log d)} \cdot n^{\mathcal{O}(1)}$ time if the given ordered graph (G, σ) is $(d + \sqrt{k})$-planar.

From Lemma 7, we can assume, at the expense of a multiplicative factor of $2^{\mathcal{O}(d\sqrt{k} \log(k+d))} \cdot n^{\mathcal{O}(1)}$ on the running time, that the given ordered graphs (G, σ) to consider are $(d + \sqrt{k})$-planar. Thus, given (G, σ, k), we can solve Edge Deletion to 1-Page d-Planar in $2^{\mathcal{O}(d\sqrt{k} \log(d+k))} \cdot n^{\mathcal{O}(1)}$ time. This concludes the proof of Theorem 6.

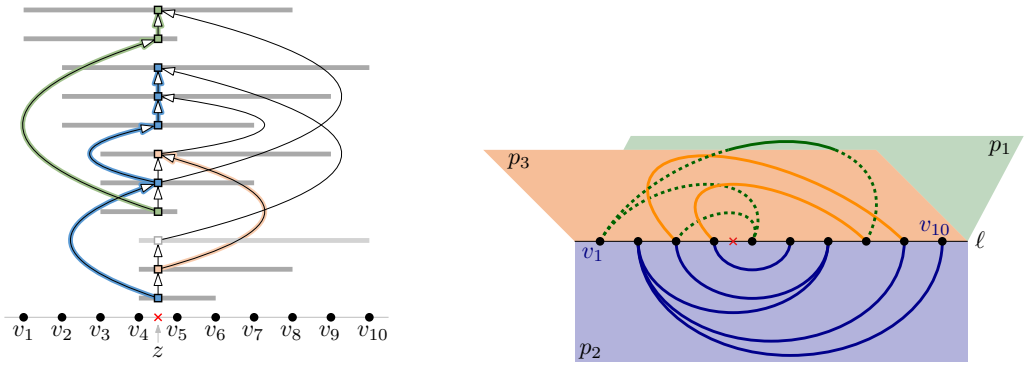
4 Edge Deletion to p-Page Planar

In this section we treat the problem Edge Deletion to p-Page Planar, which is the special case of Edge Deletion to p-Page d-Planar for $d = 0$. It can be solved by brute force in $\mathcal{O}((p + 1)^m \cdot n^2)$ time: For each mapping of the m edges to the p pages, with the "+1" to mark edge deletion, check for each pair of edges assigned to the same page whether they intersect. It can also be solved in $4^m \cdot n^{\mathcal{O}(1)}$ time: for each of the 2^m subsets of $E(G)$, use Corollary 2 to decide whether its fixed-vertex-order page number is at most p.

We now consider a new parameter in addition to p. The edge set of an ordered graph (G, σ) corresponds to a set of open intervals on the real line; namely every edge (u, v) of G is mapped to the interval $(\sigma(u), \sigma(v))$. Given a set $\mathcal{I}$ of intervals, a *hitting set* for $\mathcal{I}$ is a set of points on the real line such that each interval contains at least one of the points. Note that a hitting set can be much smaller than a vertex cover: an ordered graph (G, σ) with a hitting set of size 1 can have linear vertex cover number (e.g., $G = K_{n,n}$). Given a set $\mathcal{I}$ of m open intervals, a minimum-size hitting set for $\mathcal{I}$ can be found in $\mathcal{O}(m \log m)$ time by a simple greedy algorithm.

For two edges $(u, v), (u', v')$ of (G, σ), we say that (u, v) *contains* (u', v') if the interval $(\sigma(u), \sigma(v))$ contains the interval $(\sigma(u'), \sigma(v'))$. If (u, v) and (u', v') cross with respect to σ, then there is no containment, otherwise one contains the other.

Hitting set of size 1. We start by treating the following special case of Edge Deletion to p-Page Planar. Given an ordered graph (G, σ), a point z on the real line that is contained in every interval defined by $E(G)$, a number p of pages, and a threshold $k \geq 0$, we



(a) Intervals corrsponding to the edges of G; auxiliary graph G^+ without transitive edges.

(b) Optimal solution for (a): only the edge (v_4, v_{10}) is deleted; the pages correspond to the colored paths in (a).

Figure 3 Instance with hitting set of size 1 and optimal solution for three pages.

want to decide whether there is a set $E' \subseteq E(G)$ of size at most k such that that $G - E'$ can be drawn without crossings on p pages (respecting vertex order σ). Note that if there is a hitting set of size 1, then G is necessarily bipartite and that $z \notin \sigma(V(G))$. We show that Edge Deletion to p-Page Planar can be solved efficiently if $h(G, \sigma) = 1$.

Alam et al. [3] have called this setting *separated*; they showed that the *mixed* page number of an ordered $K_{n,n}$ is $\lceil 2n/3 \rceil$ in this case. While we study the (usual) page number of an ordered graph where each page corresponds to a stack layout, the mixed page number asks for the smallest number of stacks and queues (where nested edges are not allowed on the same page) needed to draw an ordered graph.

▶ **Theorem 9.** *Given an ordered graph (G, σ) with n vertices, m edges, and $h(G, \sigma) = 1$, Edge Deletion to p-Page Planar can be solved in $\mathcal{O}(m^3 \log n \log\log p)$ time.*

Proof. From (G, σ) we construct an acyclic directed auxiliary graph G^+, from which we then construct an s–t flow network $\mathcal{N}$ such that an integral maximum s–t flow of minimum cost in $\mathcal{N}$ corresponds to p vertex-disjoint directed paths in G^+ of maximum total length, and each path in G^+ corresponds to a set of edges in G that can be drawn without crossings on a single page in a book embedding of (G, σ). The set E' of edges that need to be deleted from G such that $G - E'$ has page number p corresponds to the vertices of G^+ that do not lie on any of the p paths.

We now describe these steps in detail. The auxiliary graph G^+ has a node for each edge of G and an arc from edge node (a, b) to edge node (a', b') if in (G, σ) the edge (a', b') contains the edge (a, b) (meaning that the edges do not cross); see Figure 3. Hence G^+ has exactly m nodes and at most $\binom{m}{2}$ edges, and can be constructed from (G, σ) in $\mathcal{O}(m^2)$ time.

The s–t flow network $\mathcal{N}$ is defined as follows. For each node v of G^+, introduce two vertices v_{in} and v_{out} in $\mathcal{N}$, connected by the arc $(v_{\text{in}}, v_{\text{out}})$ of capacity 1 and cost -1. All other arcs in $\mathcal{N}$ have cost 0. For each arc (u, v) of G^+, add the arc $(u_{\text{out}}, v_{\text{in}})$ of capacity 1 to $\mathcal{N}$. Then add to $\mathcal{N}$ new vertices s, s', and t, the edge (s, s') of capacity p, and the edges $\{(s', v_{\text{in}}), (v_{\text{out}}, t) \colon v \in V(G^+)\}$ of capacity 1. Summing up, $\mathcal{N}$ has $2m + 3$ vertices, at most $\binom{m}{2} + 3m + 1$ edges, and can be constructed from G^+ in $\mathcal{O}(m^2)$ time.

Due to the edge (s, s'), a maximum flow in $\mathcal{N}$ has value at most p. If $m \geq p$ (otherwise the instance is trivial, and no edge has to be deleted), then a maximum flow has value exactly p. Since all edge capacities and costs are integral, the minimum-cost circulation algorithm of Ahuja, Goldberg, Orlin, and Tarjan [2] yields an integral flow. Since all edges

(except for (s, s')) have edge capacity 1 and $\mathcal{N}$ is acyclic, the edges (except for (s, s')) with non-zero flow form p paths of flow 1 from s' to t that are vertex-disjoint except for their endpoints. These paths (without s' and t) correspond to vertex-disjoint paths in G^+. Due to the negative cost of the edges of type $(v_{\text{in}}, v_{\text{out}})$, the flow maximizes the number of such edges with flow. This maximizes the number of vertices in G^+ that lie on one of the p paths. This, in turn, maximizes the number of edges of G that can be drawn without crossings on p pages in a book embedding of (G, σ). Given a flow network with n' vertices, m' edges, maximum capacity U, and maximum absolute cost value C, the algorithm of Ahuja et al. runs in $\mathcal{O}(n'm'(\log\log U)\log(n'C))$ time. In our case, $n' \in \mathcal{O}(m)$, $m' \in \mathcal{O}(m^2)$, $U = p$, and $C = 1$. Hence computing the maximum flow of minimum cost in $\mathcal{N}$ takes $\mathcal{O}(m^3 \log n \log\log p)$ time. This dominates the time needed to construct G^+ and $\mathcal{N}$. ◀

In our forthcoming algorithm, we will use an extension of this result, as follows. Two subsets $E', E'' \subset E(G)$ are *compatible* if $|E'| = |E''|$ and there is an enumeration $e'_1, \ldots, e'_{|E'|}$ of E' and an enumeration $e''_1, \ldots, e''_{|E'|}$ of E'' such that e'_i is contained in e''_i for all $i \in [|E'|]$. Note that we may have $E' \cap E'' \neq \emptyset$.

▶ **Lemma 10.** *Given an ordered graph (G, σ) with n vertices, m edges, $h(G, \sigma) = 1$, and subsets $E', E'' \subset E(G)$ with $p = |E'| = |E''|$, we can decide, in $\mathcal{O}(m^3 \log n \log\log p)$ time, whether E' and E'' are compatible and, if yes, solve a version of* Edge Deletion to p-Page Planar *where, on each page, one edge of E' is contained in all other edges and one edge of E'' contains all other edges on that page.*

Proof. We adapt the proof of Theorem 9 by modifying the flow network $\mathcal{N}$ that is considered. More precisely, we insert arcs from s' only to the edges $e' \in E'$, and we insert arcs to t only from the edgs $e'' \in E''$. No other arcs go out from s' nor go into t.

Note that E' and E'' are compatible if and only if the value of the maximum flow in the modified flow network is exactly p. ◀

Our technique, based on flows, does not allow us to enforce a pairing of the edges in E' and in E''. With other words, we cannot select edges $e'_1, e'_2 \in E'$ and $e''_1, e''_2 \in E''$, and insist that e'_1 and e''_1 go to one page, and e'_2 and e''_2 go to another page. This difficulty will play an important role in our forthcoming extension.

An XP algorithm for the general case. Let H be a finite hitting set of (G, σ). We assume, without loss of generality, that $H \cap \sigma(V(G)) = \emptyset$. Given a subset $X \subseteq H$, we say that an edge (u, v) of G with $\sigma(u) < \sigma(v)$ *bridges* X if $\sigma(u) < \min X$, $\max X < \sigma(v)$, and X is the largest subset of H with this property. For each $X \subseteq H$, let E_X be the subset of edges of (G, σ) that bridge X. For example, in Figure 4, $|H| = 3$, and the edges in E_H lie in the outer gray region.

Consider any drawing of a subgraph of (G, σ) with edge set $\tilde{E}$ on p pages without crossings. For each page $q \in [p]$, let $\tilde{E}^q$ be the set of edges in $\tilde{E}$ that are on page q, and let $\mathcal{X}^q$ be the family of subsets of H bridged by some edge of $\tilde{E}^q$. Since there are no crossings on page q, the sets of $\mathcal{X}^q$ form a so-called *laminar* family: any two sets in $\mathcal{X}^q$ are either disjoint or one contains the other. For each $X \in \mathcal{X}^q$, let e^q_X be the smallest edge of $\tilde{E}^q$ that bridges X, and let f^q_X be the largest edge of $\tilde{E}^q$ that bridges X; it may be that $e^q_X = f^q_X$. Note that for each $X, Y \in \mathcal{X}^q$ with $X \subsetneq Y$, the edge e^q_Y contains f^q_X. We say that the *partial encoding* of $\tilde{E}$ on page q is $\mathcal{E}^q = \{(X, e^q_X, f^q_X) \mid X \in \mathcal{X}^q\}$ and the *encoding* of $\tilde{E}$ is $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$.

When a set X is bridged on only one page of an optimal drawing, say $X \in \mathcal{X}^1$, then we just have to select as many edges as possible without crossing from those contained between

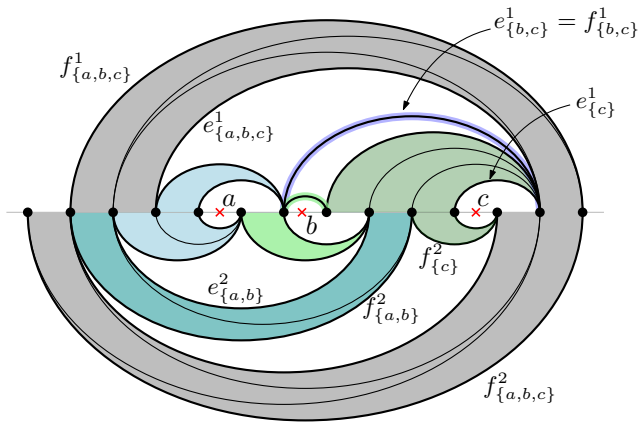


Figure 4 Encoding $\langle \mathcal{E}^1, \mathcal{E}^2 \rangle$ of a 2-page drawing for an instance with hitting set $H = \{a, b, c\}$ (red crosses). For each $X \subseteq H$ and page $q \in [2]$, the edges e^q_X and f^q_X (if they exist) are thicker than the other edges. Each colored region corresponds to a set of edges that bridge the same subset of H.

e^1_X and f^1_X, because the edges of E_X cannot appear in any other page. The challenge that we face is the following: when the same set X appears in $\mathcal{X}^q$ for different $q \in [p]$, the choices of which edges are drawn in each of those pages are not independent. However, we can treat all such pages together, exchanging some parts of the drawings from one page to another, as follows. For each $X \subseteq H$, let $Q_X = \{q \in [p] \colon X \in \mathcal{X}^q\}$ be the set of pages where some edges bridge X.

▶ **Lemma 11.** *Consider $\tilde{E} \subseteq E(G)$ that can be drawn in p pages without crossings, and let $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$ be the corresponding encoding. For every $X \subseteq H$ with $Q_X \neq \emptyset$, let $\tilde{E}'_X = \{e^q_X \mid q \in Q_X\}$, let $\tilde{E}''_X = \{f^q_X \mid q \in Q_X\}$, and let F_X be the set of edges in E_X obtained when using Lemma 10 for $p' = |Q_X|$ pages with boundary edges $\tilde{E}'_X$ and $\tilde{E}''_X$. Then the ordered subgraph with edge set $\bigcup_X F_X$ can be drawn on p pages without crossings and contains at least as many edges as $\tilde{E}$.*

Proof. Consider a fixed $X \subseteq H$ with $Q_X \neq \emptyset$. For each $q \in Q_X$, let F^q_X be the set of edges in F_X that appear on the same page as $e^q_X \in \tilde{E}'_X$ when using the algorithm of Lemma 10. Since each element of $\tilde{E}''_X$ is on a different page, let $\sigma \colon Q_X \to Q_X$ be the permutation such that $f^{\sigma(q)}_X$ is the unique element of $\tilde{E}''_X$ in F^q_X.

We make a drawing of $\hat{E} := (\tilde{E} \setminus E_X) \cup F_X$ on p pages by assigning edges to pages, as follows. For each $q \in [p] \setminus Q_X$, we just set $\hat{E}^q = \tilde{E}^q$. For each $q \in Q_X$, let $\hat{E}^q$ be obtained from $\tilde{E}^{\sigma(q)}$ by removing the edges contained in $f^{\sigma(q)}_X$, adding the edges of F^q_X, and adding the edges of $\tilde{E}^q$ contained in e^q_X. For an example, see Figure 5. For each q, the edges of $\hat{E}^q$ can be drawn on a single page without crossings. This is obvious for $q \in [p] \setminus Q_X$. For $q \in Q_X$, this is true because e^q_X and $f^{\sigma(q)}_X$ act as shields between F^q_X and the other two groups of edges, one containing $f^{\sigma(q)}_X$ and the other contained in e^q_X.

Since $\tilde{E} \cap E_X = \left(\bigcup_{q \in Q_X} \tilde{E}^q\right) \cap E_X$ is a feasible solution for the problem solved in Lemma 10, we have $|\tilde{E} \cap E_X| \leq |F_X|$. Therefore $\hat{E} = (\tilde{E} \setminus E_X) \cup F_X$ is at least as large as $\tilde{E}$.

Summarizing: for a fixed X, we have converted $\tilde{E}$ into another set of edges $\hat{E}$ that is no smaller and can be drawn without crossings on p pages such that $F_X = \hat{E} \cap E_X$ and such that no edge outside E_X is changed (that is, $\tilde{E} \setminus E_X = \hat{E} \setminus E_X$). In general, the encoding $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$ changes, but the sets $\tilde{E}'_X, \tilde{E}''_X$ remain unchanged for every set X. We now iterate this process for each $X \subseteq H$. The last set $\hat{E}$ that we obtain is $\bigcup_X F_X$ because every edge of $\tilde{E}$ is in E_X for some $X \subseteq H$. The result follows. ◀

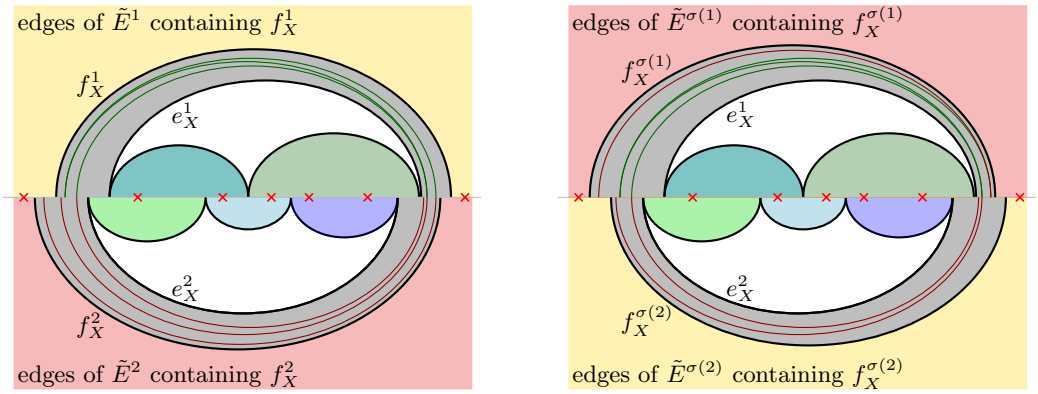


Figure 5 Left: A 2-page drawing of $\tilde{E}$. The gray region corresponds to the set $\tilde{E}_X = \tilde{E}_X^1 \cup \tilde{E}_X^2$ when X is the set of the inner five red crosses. Right: drawing of a set $\hat{E} = (\tilde{E} \setminus E_X) \cup F_X$ where $\sigma(1) = 2$ and $\sigma(2) = 1$. Note that $\tilde{E}_X$ and $\hat{E}_X$ can be different; namely if $F_X \neq \tilde{E}_X^1 \cup \tilde{E}_X^2$.

We now argue that, on a single page $q \in [p]$, the number of possible partial encodings $\mathcal{E}^q$ is at most m^{4h-2}. First note that $\mathcal{X}^q$ contains at most $2h-1$ sets: at most h sets in $\mathcal{X}^q$ are inclusionwise minimal, and any non-minimal element $X \in \mathcal{X}^q$ is obtained by joining two others. This means that $\mathcal{E}^q$ is characterized by selecting at most $4h-2$ edges e_X^q and f_X^q, and such a selection already determines implicitly the sets $\mathcal{X}^q$. When considering all pages together, there are at most $m^{(4h-2)\cdot p}$ encodings $\langle \mathcal{E}^1, \ldots, \mathcal{E}^p \rangle$, and, for each $X \in \bigcup_{q \in [p]} \mathcal{X}^q$, we have to apply the algorithm of Lemma 10, which takes $\mathcal{O}(|E_X|^3 \log n \log\log p)$ time. Since the edge sets E_X are pairwise disjoint for different $X \subseteq H$, for each encoding we spend $\mathcal{O}(m^3 \log n \log\log p)$ time. Finally, we return the best among all encodings that give rise to a valid drawing without crossings. Since the encoding of an optimal solution will be considered at least once, Lemma 11 implies that we find an optimal solution. Therefore, the total running time is $\mathcal{O}(m^{(4h-2)\cdot p+3} \log n \log\log p)$. We summarize our result.

▶ **Theorem 12.** *Edge Deletion to p-Page Planar is in XP with respect to $h+p$.*

5 Multiple-Track Crossing Minimization

Let $G = (A \cup B, E)$ be a bipartite graph where all edges connect a vertex of A to a vertex of B and $A \cap B = \emptyset$. We further have a given linear order σ_A for the vertices of A. For the vertices of B we do not have any additional information or constraints. In this section we consider *spine+t-track drawings* of G, defined as follows:

- the vertices of A are placed on a line ℓ_0, called *spine*, in the order determined by σ_A;
- the vertices of B are placed on t different lines $\ell_1, \ldots, \ell_t$ parallel to the spine; each line ℓ_q is placed on a different *page* (half-plane) π_q of a book;
- all pages $\pi_1, \ldots, \pi_t$ have ℓ_0 as a common boundary and are otherwise pairwise disjoint;
- for each $q \in [t]$, the edges with endpoints in ℓ_0 and ℓ_q are drawn as straight-lines edges in the page π_q.

One can interpret this as a drawing in three dimension, as shown in Figure 6. Note that because the graph is bipartite and each edge has a vertex in A and a vertex in B, there are no edges connecting two vertices in the spine, and in particular there are no "nested" edges.

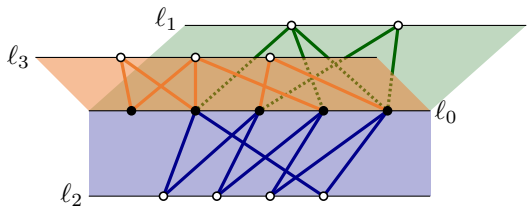


Figure 6 A spine+3-track drawing. In this example, B_1 has two vertices, B_2 has four vertices and B_3 has three vertices. The drawing has $2 + 5 + 2 = 9$ crossings.

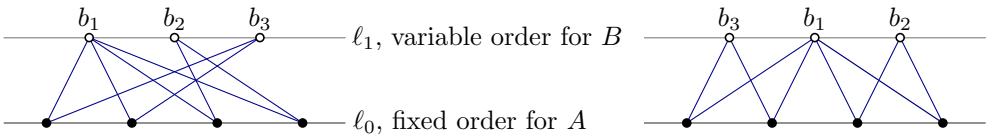


Figure 7 Two different orders σ_B give different number of crossings in the spine+1-track drawing: 10 on the left and 2 on the right.

To describe the drawing combinatorially, it suffices to partition B into sets $B_1, \ldots, B_t$, one per line, and we have to decide for each B_q the order σ_{B_q} of the vertices B_q along ℓ_q. The number of crossings of the drawing is the sum of the number of crossings within each page, where the number of crossings within a page is the number of pairs of edges that cross each other. The value $\mathrm{cr}_t((A, \sigma_A), B, E)$ is the minimum number of crossings over all spine+t-track drawings, and the purpose of this section is to discuss its computation.

We start discussing spine+1-track drawings and its corresponding value $\mathrm{cr}_1((A, \sigma_A), B, E)$. See Figure 7 for examples of drawings. This is the minimum number of crossings in a two-layer drawing with the order on one layer, A in this case, fixed. We want to choose the order σ_B that minimizes the number of crossings. Let $\mathrm{cr}_1((A, \sigma_A), (B, \sigma_B), E)$ be the crossing number for a fixed order σ_B. Then $\mathrm{cr}_1((A, \sigma_A), B, E)$ is the minimum of $\mathrm{cr}_1((A, \sigma_A), (B, \sigma_B), E)$ when we optimize over all orders σ_B of B. The obvious approach is to try all different possible orders σ_B of B, compute $\mathrm{cr}_1((A, \sigma_A), (B, \sigma_B), E)$ for each of them, and take the minimum. This yields an algorithm with time complexity $2^{\mathcal{O}(n \log n)}$. We improve over this trivial algorithm as follows.

▶ **Theorem 13.** *We can compute* $\mathrm{cr}_1((A, \sigma_A), B, E)$ *in* $\mathcal{O}(2^n n)$ *time, where* $n = |A| + |B|$.

Proof. Construct a complete, directed, edge-weighted graph H as follows:

- $V(H) = B$
- put all directed edges in H;
- the directed edge (x, y) of H gets weight $c_{x,y} = \mathrm{cr}_1((A, \sigma_A), (\{x, y\}, \sigma_{x,y}), E)$, where $\sigma_{x,y}$ is the order for $\{x, y\}$ that places x before y.

An ordering of B corresponds to a Hamiltonian path in H. Consider any Hamiltonian path in H defined by an order σ_B. Since each crossing happens between two edges incident to different vertices of B, we have

$$\mathrm{cr}_1((A, \sigma_A), (B, \sigma_B), E) = \sum_{\substack{x, y \in B \\ \sigma_B(x) < \sigma_B(y)}} c_{x,y} = \sum_{x \in B} \sum_{\substack{y \in B \\ \sigma_B(x) < \sigma_B(y)}} c_{x,y}. \tag{2}$$

With this interpretation, the task is to find in H a Hamiltonian path such that the sum of the $c_{\cdot,\cdot}$-weights from each vertex to all its successors is minimized. This problem is

amenable to dynamic programming across subsets of vertices, as it is done for the Traveling Salesperson Problem; see [5] or [12, Section 6.6].

We define a table C by setting, for each $X \subseteq B$,

$$C[X] \;=\; \mathrm{cr}_1((A,\sigma_A), X, \{(a,x) \in E \mid x \in X, a \in A\}).$$

Then $C[X]$ is the number of crossings when we remove the vertices $B \setminus X$ from H. We are interested in computing $C[B]$ because $C[B] = \mathrm{cr}_1((A,\sigma_A), B, E)$.

We obviously have $C[X] = 0$ for each $X \subseteq B$ with $|X| \leq 1$. Whenever $|X| > 1$, we use (2) and the definition of $C[X]$ to obtain the recurrence

$$C[X] \;=\; \min_{y \in X} \left(C[X \setminus \{y\}] + \sum_{x \in X \setminus \{y\}} c_{x,y} \right). \tag{3}$$

The proof of this is a standard proof in dynamic programming, where y represents the last vertex of X in the ordering.

Each value $c_{x,y}$ can be computed in $\mathcal{O}(\deg_G(x) + \deg_G(y)) = \mathcal{O}(n)$ time, which means that, over all pairs (x,y), we spend $\mathcal{O}(n^3)$ time. Each value $\eta[X,y] := \sum_{x \in X \setminus \{y\}} c_{x,y}$, defined for $X \subseteq B$ and $y \in X$, can be computed for increasing values of $|X|$ in constant time per value by noting that

$$\text{for every } X \subseteq B \text{ and distinct } y, z \in X\text{:} \quad \sum_{x \in X \setminus \{y\}} c_{x,y} = c_{z,y} + \sum_{x \in X \setminus \{y,z\}} c_{x,y}.$$

Therefore, we compute the value $\eta[X,y]$ for every $X \subseteq B$ and $y \in X$ in $\Theta\left(\sum_{k=0}^{n} \binom{n}{k} k\right) = \Theta(2^n n)$ total time. (The direct computation using the sums anew for each value would take $\Theta\left(\sum_{k=0}^{n} \binom{n}{k} k^2\right) = \Theta(2^n n^2)$, which is strictly larger.)

After this we can compute $C[X]$ for increasing values of $|X|$ using the recurrence of Equation (3), which means that we spend $\mathcal{O}(|X|)$ time for each X. This step also takes $\mathcal{O}(2^n n)$ time for all X. Finally we return $C[B]$. An optimal solution can be recovered using standard book-keeping techniques. ◀

Now we consider the case of arbitrary track number t.

▶ **Theorem 14.** *We can compute* $\mathrm{cr}_t((A,\sigma_A), B, E)$ *in* $2^n n^{\mathcal{O}(1)}$ *time for every* $t > 1$*, where* $n = |A| + |B|$*. For* $t = 1$ *and* $t = 2$*, the value can be computed in* $O(2^n n)$ *time.*

Proof. Once we fix a set B_q for the qth page, we can optimize the order σ_{B_q} independently of all other decisions. Therefore, we want to compute

$$\min \sum_{q=1}^{t} \mathrm{cr}_1((A,\sigma_A), B_q, E_q),$$

where E_q is the set of edges connecting vertices from A to B_q, and where the minimum is only over all the partitions $B_1, \ldots, B_t$ of B.

As we did in the proof of Theorem 13, for each subset $X \subseteq B$, we define

$$C[X] \;=\; \mathrm{cr}_1((A,\sigma_A), X, \{(a,x) \in E \mid x \in X, a \in A\}).$$

In the proof of Theorem 13 we argued that the values $C[X]$ can be computed in $\mathcal{O}(2^n n)$ time for all $X \subseteq B$ simultaneously.

We have to compute now

$$\min\left\{\sum_{q=1}^{t} C[B_q] \colon B_1,\ldots,B_t \text{ is a partition of } B\right\}.$$

The case of $t=1$ has been covered in Theorem 13. For $t=2$, we have to compute

$$\min\{C[B_1]+C[B\setminus B_1] \mid B_1\subseteq B\},$$

which can be done in $O(2^n)$ additional time iterating over all subsets B_1 of B.

For $t>2$, we use the algorithm of Björklund et al. [8] for subset convolution, as follows. Define for each $X\subseteq B$ and for $q\in[t]$ the "entry table"

$$\begin{aligned} T[X,q] &= \mathrm{cr}_q((A,\sigma_A),X,\{(a,x)\in E \mid x\in X, a\in A\}) \\ &= \min\{C[B_1]+\ldots+C[B_q] \mid B_1,\ldots,B_q \text{ is a partition of } X\}. \end{aligned}$$

We obviously have $T[X,1]=C[X]$ for all X. For $q>1$, we have the recursive relation

$$T[X,q] = \min\{T[Y,q-1]+C[X\setminus Y] \mid Y\subseteq X\}.$$

Therefore, for $q>1$, the function $X\mapsto T[X,q]$ is, by definition, the subset convolution of the functions $X\mapsto T[X,q-1]$ and $X\mapsto C[X]$ in the $(\min,+)$ ring. These functions take integer values on $\{0,\ldots,n^4\}$ because n^4 is an upper bound for $\mathrm{cr}_q((A,\sigma_A),B,E)$ for any q. It follows from [8] that one can obtain $T[X,q]$ for all $X\subseteq B$ in $2^n n^{\mathcal{O}(1)}$ time, assuming that $T[\cdot,q-1]$ and $C[\cdot]$ are already available. We compute the entries $T[\cdot,q]$ for $q=2,\ldots,t$, which adds a multiplicative $t\le n$ to the final running time. ◀

Using the theorem for increasing values of t, we obtain the following.

▶ **Corollary 15.** *We can compute the smallest value t such that $\mathrm{cr}_t((A,\sigma_A),B,E)=0$ in $2^n\cdot n^{\mathcal{O}(1)}$ time, where $n=|A|+|B|$.*

6 Open Problems

1. Could we use the concept of the conflict graph for other crossing reduction problems?
2. Is Edge Deletion to 1-Page d-Planar $W[1]$-hard with respect to the natural parameter k if d is part of the input? Can we reduce from Independent Set? Note that Vertex Deletion to Degree-d is $W[1]$-hard with respect to treewidth [6] and that outer-d planar graphs have treewidth $\mathcal{O}(d)$ [10] (which also follows from Lemma 8).
3. What if the vertex order is not given? In other words, what is the parameterized complexity of edge deletion to outer-d planarity?
4. What about exact algorithms for computing the crossing number of an ordered graph? As Masuda et al. [29] showed, the problem is NP-hard for two pages. In their NP-hardness reduction, they use a large number of crossings, and it is easy to get an algorithm that is exponential in the number of edges; see Theorem 1. Can we get a running time of $2^n\cdot n^{\mathcal{O}(1)}$ or perhaps even subexponential in n? Recall that the algorithm of Liu et al. [26] checks in $n\cdot(\mathrm{cr}+2)^{\mathcal{O}(\mathrm{pw}^2)}$ time whether a graph with pathwidth pw can be drawn on a given number of pages with at most cr crossings in total.

References

1. Akanksha Agrawal, Sergio Cabello, Michael Kaufmann, Saket Saurabh, Roohani Sharma, Yushi Uno, and Alexander Wolff. Eliminating crossings in ordered graphs. arXiv report, 2024. arXiv:2404.09771.
2. Ravindra K. Ahuja, Andrew V. Goldberg, James B. Orlin, and Robert E. Tarjan. Finding minimum-cost flows by double scaling. *Math. Progr.*, 53:243–266, 1992. doi:10.1007/BF01585705.
3. Jawaherul Md. Alam, Michael A. Bekos, Martin Gronemann, Michael Kaufmann, and Sergey Pupyrev. The mixed page number of graphs. *Theoret. Comput. Sci.*, 931:131–141, 2022. doi:10.1016/j.tcs.2022.07.036.
4. Patricia Bachmann, Ignaz Rutter, and Peter Stumpf. On the 3-coloring of circle graphs. In Michael Bekos and Markus Chimani, editors, *Proc. Int. Symp. Graph Drawing & Network Vis. (GD)*, volume 14465 of *LNCS*, pages 152–160. Springer, 2023. doi:10.1007/978-3-031-49272-3_11.
5. Richard Bellman. Dynamic programming treatment of the travelling salesman problem. *J. ACM*, 9(1):61–63, 1962. doi:10.1145/321105.321111.
6. Nadja Betzler, Robert Bredereck, Rolf Niedermeier, and Johannes Uhlmann. On bounded-degree vertex deletion parameterized by treewidth. *Discrete Appl. Math.*, 160(1):53–60, 2012. doi:j.dam.2011.08.013.
7. Sujoy Bhore, Robert Ganian, Fabrizio Montecchiani, and Martin Nöllenburg. Parameterized algorithms for book embedding problems. *J. Graph Algorithms Appl.*, 24(4):603–620, 2020. doi:10.7155/jgaa.00526.
8. Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Fourier meets Möbius: Fast subset convolution. In David S. Johnson and Uriel Feige, editors, *Proc. 39th Ann. ACM Symp. Theory Comput. (STOC)*, pages 67–74, 2007. doi:10.1145/1250790.1250801.
9. Sergio Cabello and Bojan Mohar. Adding one edge to planar graphs makes crossing number and 1-planarity hard. *SIAM J. Comput.*, 42(5):1803–1829, 2013. doi:10.1137/120872310.
10. Steven Chaplick, Myroslav Kryven, Giuseppe Liotta, Andre Löffler, and Alexander Wolff. Beyond outerplanarity. In Fabrizio Frati and Kwan-Liu Ma, editors, *Proc. 25th Int. Symp. Graph Drawing & Network Vis. (GD)*, volume 10692 of *LNCS*, pages 546–559. Springer, 2018. doi:10.1007/978-3-319-73915-1_42.
11. Fan R. K. Chung, Frank Thomson Leighton, and Arnold L. Rosenberg. Embedding graphs in books: A layout problem with applications to VLSI design. *SIAM J. Algebr. Discrete Meth.*, 8(1):33–58, 1987. doi:10.1137/0608002.
12. Sanjoy Dasgupta, Christos H. Papadimitriou, and Umesh V. Vazirani. *Algorithms*. McGraw-Hill, 2008.
13. Huib Donkers, Bart M. P. Jansen, and Michał Włodarczyk. Preprocessing for outerplanar vertex deletion: An elementary kernel of quartic size. *Algorithmica*, 84(11):3407–3458, 2022. doi:10.1007/s00453-022-00984-2.
14. Zdeněk Dvořák and Sergey Norin. Treewidth of graphs with balanced separations. *J. Comb. Theory, Ser. B*, 137:137–144, 2019. doi:10.1016/j.jctb.2018.12.007.
15. Michael R. Fellows, Jiong Guo, Hannes Moser, and Rolf Niedermeier. A generalization of Nemhauser and Trotter's local optimization theorem. *J. Comput. Syst. Sci.*, 77(6):1141–1158, 2011. doi:10.1016/j.jcss.2010.12.001.
16. Fanica Gavril. Algorithms for a maximum clique and a maximum independent set of a circle graph. *Networks*, 3(3):261–273, 1973. doi:10.1002/net.3230030305.
17. Martin Grohe. Computing crossing numbers in quadratic time. *J. Comput. Syst. Sci.*, 68(2):285–302, 2004. doi:10.1016/j.jcss.2003.07.008.
18. Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh. A near-optimal planarization algorithm. In Chandra Chekuri, editor, *Proc. Ann. ACM-SIAM Symp. Discrete Algorithms (SODA)*, pages 1802–1811, 2014. doi:10.1137/1.9781611973402.130.

19 Bart M. P. Jansen and Michał Włodarczyk. Lossy planarization: a constant-factor approximate kernelization for planar vertex deletion. In Stefano Leonardi and Anupam Gupta, editors, *Proc. 54th Ann. ACM Symp. Theory Comput. (STOC)*, pages 900–913, 2022. doi:10.1145/3519935.3520021.

20 Ken-ichi Kawarabayashi. Planarity allowing few error vertices in linear time. In *Proc. Ann. IEEE Symp. Foundat. Comput. Sci. (FOCS)*, pages 639–648, 2009. doi:10.1109/FOCS.2009.45.

21 Ken-ichi Kawarabayashi and Bruce A. Reed. Computing crossing number in linear time. In David S. Johnson and Uriel Feige, editors, *Proc. 39th Ann. ACM Symp. Theory Comput. (STOC)*, pages 382–390, 2007. doi:10.1145/1250790.1250848.

22 Yasuaki Kobayashi and Hisao Tamaki. A fast and simple subexponential fixed parameter algorithm for one-sided crossing minimization. *Algorithmica*, 72:778–790, 2015. doi:10.1007/s00453-014-9872-x.

23 Yasuaki Kobayashi and Hisao Tamaki. A faster fixed parameter algorithm for two-layer crossing minimization. *Inform. Process. Lett.*, 116(9):547–549, 2016. doi:j.ipl.2016.04.012.

24 Michael Lampis and Manolis Vasilakis. Structural parameterizations for two bounded degree problems revisited. *CoRR*, abs/2304.14724, 2023. doi:10.48550/arXiv.2304.14724.

25 Yunlong Liu, Jie Chen, and Jingui Huang. Parameterized algorithms for fixed-order book drawing with bounded number of crossings per edge. In Weili Wu and Zhongnan Zhang, editors, *Proc. 14th Int. Conf. Combin. Optim. Appl. (COCOA)*, volume 12577 of *LNCS*, pages 562–576. Springer, 2020. doi:10.1007/978-3-030-64843-5_38.

26 Yunlong Liu, Jie Chen, Jingui Huang, and Jianxin Wang. On parameterized algorithms for fixed-order book thickness with respect to the pathwidth of the vertex ordering. *Theor. Comput. Sci.*, 873:16–24, 2021. doi:10.1016/j.tcs.2021.04.021.

27 Daniel Lokshtanov, N. S. Narayanaswamy, Venkatesh Raman, M. S. Ramanujan, and Saket Saurabh. Faster parameterized algorithms using linear programming. *ACM Trans. Algorithms*, 11(2):15:1–15:31, 2014. doi:10.1145/2566616.

28 Dániel Marx and Ildikó Schlotter. Obtaining a planar graph by vertex deletion. *Algorithmica*, 62(3-4):807–822, 2012. doi:10.1007/s00453-010-9484-z.

29 Sumio Masuda, Kazuo Nakajima, Toshinobu Kashiwabara, and Toshio Fujisawa. Crossing minimization in linear embeddings of graphs. *IEEE Trans. Computers*, 39(1):124–127, 1990. doi:10.1109/12.46286.

30 Nicholas Nash and David Gregg. An output sensitive algorithm for computing a maximum independent set of a circle graph. *Inf. Process. Lett.*, 110(16):630–634, 2010. doi:10.1016/j.ipl.2010.05.016.

31 Naomi Nishimura, Prabhakar Ragde, and Dimitrios M. Thilikos. Fast fixed-parameter tractable algorithms for nontrivial generalizations of vertex cover. *Discret. Appl. Math.*, 152(1-3):229–245, 2005. doi:10.1016/j.dam.2005.02.029.

32 Bruce A. Reed, Kaleigh Smith, and Adrian Vetta. Finding odd cycle transversals. *Oper. Res. Lett.*, 32(4):299–301, 2004. doi:10.1016/J.ORL.2003.10.009.

33 Kozo Sugiyama, Shojiro Tagawa, and Mitsuhiko Toda. Methods for visual understanding of hierarchical system structures. *IEEE Trans. Syst. Man Cybernetics*, 11(2):109–125, 1981. doi:10.1109/TSMC.1981.4308636.

34 Walter Unger. On the k-colouring of circle-graphs. In Robert Cori and Martin Wirsing, editors, *Proc. 5th Ann. Symp. Theoret. Aspects Comput. Sci. (STACS)*, volume 294 of *LNCS*, pages 61–72. Springer, 1988. doi:10.1007/BFb0035832.

35 Walter Unger. The complexity of colouring circle graphs. In Alain Finkel and Matthias Jantzen, editors, *Proc. 9th Ann. Symp. Theoret. Aspects Comput. Sci. (STACS)*, volume 577 of *LNCS*, pages 389–400. Springer, 1992. doi:10.1007/3-540-55210-3_199.

36 Gabriel Valiente. A new simple algorithm for the maximum-weight independent set problem on circle graphs. In Toshihide Ibaraki, Naoki Katoh, and Hirotaka Ono, editors, *Proc. Int. Symp. Algorithms Comput. (ISAAC)*, volume 2906 of *LNCS*, pages 129–137. Springer, 2003. doi:10.1007/978-3-540-24587-2_15.

37 Mingyu Xiao. On a generalization of Nemhauser and Trotter's local optimization theorem. *J. Comput. Syst. Sci.*, 84:97–106, 2017. doi:10.1016/j.jcss.2016.08.003.

38 Meirav Zehavi. Parameterized analysis and crossing minimization problems. *Comput. Sci. Rev.*, 45:100490, 2022. doi:10.1016/j.cosrev.2022.100490.