Crossing Number Is NP-Hard for Constant Path-Width (And Tree-Width)

Petr Hliněný
Masaryk University, Brno, Czech Republic

Liana Khazaliya
Technische Universität Wien, Austria

Abstract

Crossing Number is a celebrated problem in graph drawing. It is known to be NP-complete since the 1980s, and fairly involved techniques were already required to show its fixed-parameter tractability when parameterized by the vertex cover number. In this paper we prove that computing exactly the crossing number is NP-hard even for graphs of path-width 12 (and as a result, for simple graphs of path-width 13 and tree-width 9).

Thus, while tree-width and path-width have been very successful tools in many graph algorithm scenarios, our result shows that general crossing number computations unlikely (under P $\neq$ NP) could be successfully tackled using graph decompositions of bounded width, what has been a "*tantalizing open problem*" [S. Cabello, Hardness of Approximation for Crossing Number, 2013] till now.

2012 ACM Subject Classification Mathematics of computing → Graph theory; Theory of computation → Computational geometry

Keywords and phrases Graph Drawing, Crossing Number, Tree-width, Path-width

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2024.40

Related Version *Full Version*: `https://arxiv.org/abs/2406.18933`

Funding Liana Khazaliya acknowledges support from the Austrian Science Fund (FWF) [Y1329] and the European Union's Horizon 2020 COFUND programme [LogiCS@TUWien, grant agreement No.101034440].

1 Introduction

The notion of a crossing number originally arose during WWII by Turán [16] for completed bipartite graphs in the context of the minimization of the number of crossings between tracks connecting brick kilns to storage sites. Formally, the *crossing number* $\mathrm{cr}(G)$ of a graph G is the minimum number of pairwise edge crossings in a drawing of G in the plane. Determining the crossing number of a graph is one of the most prominent combinatorial optimization problems in graph theory.

Back in the 1980s, Garey and Johnson [8] showed that the crossing number minimization is NP-hard. Their result has been extended even to fairly restrictive graph classes, in particular the problem is NP-hard even for cubic graphs [10], and also for a fixed rotation scheme [14]. Moreover, Crossing Number is APX-hard [2] (does not admit a PTAS unless P = NP) in its general setting.

Another direction of the extensive research is on computation of the crossing number for graphs that are initially close to planar graphs. Surprisingly, Crossing Number remains NP-hard for almost planar graphs (graphs that can be made planar and hence crossing-free by the removal of just a single edge) [3], and remains NP-hard on almost planar graphs even when only 3 vertices are of degree greater than 3 [11, 12]. This means that with respect to the maximum degree of the graph, as well as with respect to the number of edges to remove from the graph to make it planar, Crossing Number is para-NP-hard.

One more way to deal with the hardness of Crossing Number, is exploiting the structure of the input to get an understanding of how it affects the computational feasibility of the problem. From this side, if the input graph has a vertex cover of bounded size, then the crossing number can be computed exactly in FPT-time (some function of the parameter, i.e. vertex cover number, multiplied by a polynomial of the input size) [13]. Thus, investigation of the Crossing Number for other structural parameters (in particular, feedback vertex set number, tree-depth, path-width, and tree-width) not once was mentioned as an interesting research venue to explore [1, 2, 17]. In this direction, it is known that the problem is solvable in linear time on maximal graphs of path-width 3, admits a 2-approximation algorithm on (general) graphs of path-width 3, and admits a $4w^3$-approximation on maximal graphs of path-width w [1]. For a more involved overview of the results on Crossing Number, we refer the reader to a recent survey by Zehavi [17].

In this paper, we present a hardness result: Crossing Number is NP-hard even on a graphs of constant path-width (and, respectively, tree-width), namely, for path-width 12 (and tree-width 9). That also immediately closes the question of whether Crossing Number is FPT or XP on aforementioned graph classes under usual complexity assumptions, since our result shows that the problem is para-NP-hard.

▶ **Theorem 1.1** (cf. Theorem 3.1 and Theorem 3.2). *Given a graph G and an integer k, the problem to decide whether a graph G can be drawn with at most k crossings is* NP*-complete even when G is required to have path-width at most* 12*, and when G is required to be simple of path-width at most* 13 *and tree-width at most* 9*.*

In Section 2 we define the basic concepts, i.e., of a drawing, the crossing number, width decompositions, and the problem itself. In Section 3 we describe a hardness reduction from Satisfiability. Since the proof is rather technical, we propose separately the construction (Section 3.3), necessary conditions for an existence of a drawings with some predefined crossing number (Section 3.4), correctness of the reduction (Section 3.5), and, lastly, that the width parameters, i.e. path-width and tree-width, of the constructed graph are constant (Section 3.6). We conclude with Section 4.

2 Preliminaries

We will consider finite graphs with possible parallel edges throughout the paper. We begin with the standard terminology of graph theory [7], including the notions of tree-width and path-width [5] which are commonly used parameters to capture the complexity of a graph, and of graph drawing concepts [6].

Furthermore, for an integer $n \in \mathbb{N}$, we denote by $[n] = \{1, \dots, n\}$.

2.1 Drawings

A *drawing* $\mathcal{G}$ of a graph G in the plane is a mapping of the vertices $V(G)$ to distinct points in the plane, and of the edges $E(G)$ to simple curves connecting their respective endpoints but not containing any other vertex point. When convenient, we will refer to the elements (vertices and edges) of the drawing by the corresponding elements of G. A *crossing* is the intersection (a common point) of two distinct edge curves, other than their common endpoint. It is well established that the search for an optimal solution to the crossing number problem can be restricted to so called *good drawings*: any pair of edges crosses at most once, adjacent edges do not cross, and there is no crossing point in common to three or more edges.

A drawing $\mathcal{G}$ is *planar* (or a *plane graph*) if $\mathcal{G}$ has no crossings, and a graph in *planar* if it has a planar drawing. The number of crossings in a particular drawing $\mathcal{G}$ is denoted by $\mathrm{cr}(\mathcal{G})$ and the minimum over all good drawings $\mathcal{G}$ of a graph G by $\mathrm{cr}(G)$. We call $\mathrm{cr}(\mathcal{G}_G)$ and $\mathrm{cr}(G)$ the *crossing number* of the drawing $\mathcal{G}_G$ and the graph G, respectively. The CROSSING NUMBER problem for a given graph G asks for a good drawing $\mathcal{G}$ with the least possible number of crossings.

We will also use a common artifice in crossing number research. In a *weighted* graph, each edge is assigned a positive number (the *weight* or thickness of the edge, usually an integer). Now the *weighted crossing number* is defined as the ordinary crossing number, but a crossing between edges e_1 and e_2, say of weights t_1 and t_2, contributes the product $t_1 \cdot t_2$ to the weighted crossing number. For the purpose of computing the crossing number, an edge of integer weight t can be equivalently replaced by t parallel edges of weights 1; this is since we can easily redraw each of the t edges closely along one with the least number of crossings. Hence, from now on, we will use weighted edges instead of parallel edges, and shortly say *crossing number* to the weighted crossing number.

2.2 Tree-width and Path-width

A *tree decomposition* $\mathcal{T}$ of an undirected graph G is a pair (T, χ), where T is a tree (whose vertices we call *nodes*) rooted at a node r and χ is a function that assigns to each node $t \in \mathcal{T}$ a set $\chi(t) \subseteq V(G)$ such that the following holds:

- For every $\{u, v\} \in E(G)$ there is a node t such that $u, v \in \chi(t)$.
- For every vertex $v \in V(G)$, the set of nodes t satisfying $v \in \chi(t)$ forms a nonempty subtree of T.

The sets $\chi(t)$, for $t \in V(T)$, are called *bags* of the tree decomposition. The *width* of a tree decomposition (T, χ) is the size of a largest set $\chi(t)$ minus 1, and the *tree-width* of the graph G, denoted $\mathrm{tw}(G)$, is the minimum width of a tree decomposition of G.

The *path decomposition* and *path-width* are defined analogously with the only difference that the tree T is required to be a path.

We are going to use the following cops-and-robber game characterization on the graph G.

- The *robber* player can freely move along cop-free paths in the graph.
- The *cops* fly in a helicopter; can land on a vertex or be lifted back up. When the helicopter shows above a vertex v, the robber has time to escape wherever they chooses to.
- The robber is caught whenever a cop lands on the robber's vertex v.

Such a game is called *monotone* if the robber never gets a chance to reach a vertex previously occupied by a cop.

The cited characterization is as follows.

▶ **Theorem 2.1** (Seymour and Thomas [15]).

(1) *The tree-width of G is at most t if and only if $t + 1$ cops can always catch the robber in G in a monotone game if the robber is visible to the cop player.*

(2) *The path-width of G is at most t if and only if $t + 1$ cops can always catch the robber in G in a monotone game provided the robber is* not *visible to the cops.*

3 Hardness Reduction

In this section, we present and prove a polynomial time reduction that given an instance $\mathcal{I} = (\mathcal{C}, \mathcal{V})$ of SATISFIABILITY, constructs an equivalent instance (G, k) of CROSSING NUMBER on a graph of constant path-width (and tree-width).

Satisfiability
Input: A set of clauses $\mathcal{C} = \{C_1, \dots, C_\ell\}$ over variables $\mathcal{V} = \{x_1, \dots, x_n\}$
Question: Does there exist an assignment of the variables $\tau : \mathcal{V} \to \{\texttt{True}, \texttt{False}\}$ satisfying all clauses in $\mathcal{C}$?

Crossing Number
Input: A graph G, and $k \in \mathbb{Z}_{\geq 0}$
Question: Does G admit a drawing $\mathcal{G}$ in the plane such that $\mathcal{G}$ has at most k crossings?

▶ **Theorem 3.1.** *There is a polynomial-time algorithm that, given an instance $\mathcal{I}$ of Satisfiability, outputs an equivalent instance of Crossing Number on a graph G of path-width at most 12 and tree-width at most 9 (where G is allowed to have parallel edges).*

If simplicity of the graph G is desirable, we immediately conclude:

▶ **Corollary 3.2.** *There is a polynomial-time algorithm that, given an instance $\mathcal{I}$ of Satisfiability, outputs an equivalent instance of Crossing Number on a simple graph G of path-width at most 13 and tree-width at most 9.*

Proof. For any graph G and $e \in E(G)$, the same drawing as a point set may be used both for G and for G with the edge e subdivided; informally, subdivisions of edges do not matter for Crossing Number. Hence, if the graph G of Theorem 3.1 contains parallel edges, we form a graph G' by subdividing each such edge of G once and obtain $\mathrm{cr}(G') = \mathrm{cr}(G)$. The tree-width does not change, and the path-width of G' grows by at most 1 compared to G. ◀

3.1 High-level Idea

Naturally, for interpreting a Satisfiability instance $\mathcal{I} = (\mathcal{V}, \mathcal{C})$ in an instance (G, k) of Crossing Number, one would use a large "grid structure". Such structure would allow to separately interpret values of the variables $\mathcal{V}$, and to let all clauses $\mathcal{C}$ interact with their variables; one could imagine, e.g., variables in columns and clauses in rows of the grid structure, and their interaction happening in specially crafted cells in which the row and the column intersect.

However, if a graph contains a large grid as a minor, then its tree-width is also large, and our aim here is to obtain a graph G of constant tree-width and path-width. Thus, we are instead going to base our reduction on a frame graph F with many small separators (here of size $4 + 4$) ordered from left to right, in order to achieve constant path-width of resulting G. The crucial thing is that for each of the separators X, there are three components of $(F - X)$ – the "left", "middle" and "right" ones – such that the left and right components are *forced to cross* with the middle component many times (see Figure 1 and Figure 2 for a brief illustration). This way we enforce the sought large grid structure in any optimal drawing of the frame F, and consequently in any optimal drawing of G.

At the same time, the frame is constructed such that there is certain drawing flexibility possible, namely we can perform "vertical flips" of the middle components of separators mentioned in the previous paragraph (see Figure 1), and these will form a part of the variable gadgets in our reduction. We will use this drawing flexibility of our variable gadgets to encode the truth values of variables in Satisfiability (see Figure 4 for a brief illustration of this encoding). Specific small gadgets (see Figure 3) will be added to the variable gadgets in G to encode in which clauses they participate, and a satisfying assignment of the variables will then be checked as a possibility to draw added global clause edges of G (one edge per

Table 1 Color-encoding of the weights of the corresponding edges; and $\Theta_{n,\ell}(\omega^1)$ denotes the class of functions f such that $C_1 \cdot \omega^1 \leq f(\omega^1) \leq C_2 \cdot \omega^1$ for positive constants C_1, C_2 dependent on n and ℓ, but not on ω.

Color	Usage	Weight
Heavy-brown (`HB`)	The frame and Var-gadgets attachments	ω^8
Light-black (`LB`)	Var-gadgets interior skeletons	ω^6
Red (`R`)	Paths in Var-gadgets (vertical)	$\omega^4 + \Theta_{n,\ell}(\omega^1)$
(`R'`)	Stairs interconnecting Var-gadgets (horizontal)	ω^3
Blue (`B`)	Paths in Var-gadgets (vertical)	$\omega^4 + \Theta_{n,\ell}(\omega^1)$
(`B'`)	Stairs connecting within Var-gadgets (horizontal)	ω^3
Cyan (`C`)	Clauses Encoding within Var-gadgets	ω^2
Green (`G`)	(Global) Clause Edges	$\omega^0 = 1$

each clause, see the green edges in Figure 2) with minimum crossing cost across the whole picture. This is an idea similar to the one used in [3]. The crucial point of the construction, however, is how to enforce the unique right crossing pattern between the frame components as in Figure 1, and for this we build upon an idea originally introduced in [12] and now detailed within Section 3.4.

3.2 Auxiliary Graphs

To facilitate the presentation, we use colors, i.e. heavy-brown (`HB`), light-black (`LB`), red (`R`), blue (`B`), cyan (`C`), green (`G`), to encode the future order of the weights of the corresponding edges (see in Section 3.2). The weights of the edges will play a crucial role in the future description of possible drawings of the constructing graph. The weight values are assigned with respect to a sufficiently large (still polynomial in $|\mathcal{C}| + |\mathcal{V}|$) edge weight ω, e.g., $\omega = |E(G)|^2$. Then, informally, even one crossing of weight ω^{t+1} "outweighs" all crossings of G of weight ω^t. Observe that, importantly, all weights used in our construction will be bounded by a polynomial in $|\mathcal{I}|$.

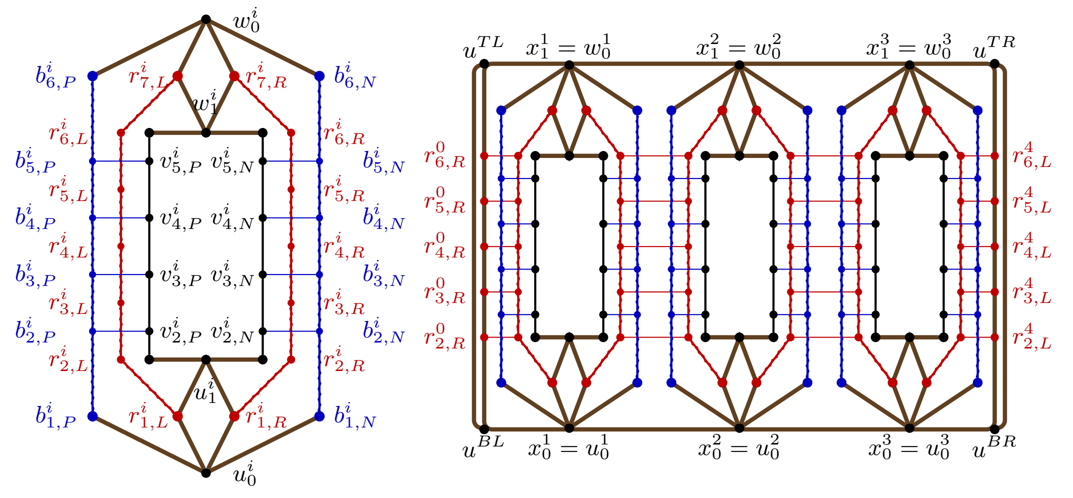
Further, we present auxiliary graphs for use as building blocks (Figure 1), before describing the whole construction of the crossing number instance G.

Variable gadgets. We start by defining Var-*gadgets*. For each $i \in [n]$, we construct a Var^i-gadget of height $h \in \mathbb{Z}_{>0}$ (see an example of Var^i for $h = 4$ in Figure 1a, the value of h to be defined later).

- First, we introduce the vertex set of Var^i as

$$V(\mathsf{Var}^i) = \{b^i_{j,P}, b^i_{j,N}, v^i_{j,P}, v^i_{j,N}\}_{j\in[h+2]} \cup \{r^i_{j,L}, r^i_{j,R}\}_{j\in[h+3]} \cup \{w^i_0, u^i_0, w^i_1, u^i_1\}.$$

- We add 6 paths as follows:
  - two `B`-paths (constructed on `B`-edges) go through vertices $\{b^i_{j,P}\}_{j\in[h+2]}$ and $\{b^i_{j,N}\}_{j\in[h+2]}$, and we will refer to the paths as `B-pos` and `B-neg` respectively;
  - two `LB`-paths go through vertices $\{v^i_{j,P}\}_{j\in[h+2]}$ (`LB-pos`) and $\{v^i_{j,N}\}_{j\in[h+2]}$ (`LB-neg`);
  - two `R`-paths go through vertices $\{r^i_{j,L}\}_{j\in[h+3]}$ (`R-left`) and $\{r^i_{j,R}\}_{j\in[h+3]}$ (`R-right`).
- We make these paths adjacent (with `HB`-edges) to the vertices $w^i_0, u^i_0, w^i_1, u^i_1$ as follows:
  - both `B-pos` and `B-neg` paths by their corner vertices to w^i_0 and u^i_0;



(a) The variable gadget Var^i, $h = 4$. **(b)** The frame with n variable gadgets for $n = 3$, $h = 4$.

Figure 1 Auxiliary graphs. Note that for each $i \in [n]$, the 8-tuple $\{u_0^i, u_1^i, r_{1,L}^i, r_{1,R}^i, w_0^i, w_1^i, r_{h+3,L}^i, r_{h+3,R}^i\}$ is a vertex cut in the frame graph.

- both `LB-pos` and `LB-neg`, analogously, to w_1^i and u_1^i; and
- both `R-left` and `R-right` paths to both w_0^i, u_0^i and w_1^i, u_1^i (see Figure 1a).

- Lastly, for each $j \in [2, h+1]$, we add *stairs* between pairs of `-pos` and `-neg` paths, i.e., pairwise connecting `B-/LB-pos` and `B-/LB-neg` paths with `B'`-edges $b_{j,P}^i v_{j,P}^i$ and $b_{j,N}^i v_{j,N}^i$ respectively.
- The *weight* of each edge of Var^i is as specified in Section 3.2; in particular, for the `R`-paths, the weight of the edges $r_{j,L}^i r_{j+1,L}^i$ and $r_{j,R}^i r_{j+1,R}^i$ is exactly $\omega^4 + j(j+1)\omega$, and for the `B`-paths, the weight of the edges $b_{j,P}^i b_{j+1,P}^i$ and $b_{j,N}^i b_{j+1,N}^i$ is exactly $\omega^4 + j(j+2)\omega$.

The frame. We construct the *frame* for n Var-gadgets of height h, where $n, h \in \mathbb{Z}_{>0}$.

- First, we introduce the `HB`-cycle (with `HB`-edges) on 4 vertices u^{BL} (bottom-left), u^{TL} (top-left), u^{TR} (top-right), u^{BR} (bottom-right) in the specified order.
- Then, we subdivide (n times) the edge between u^{BL} and u^{BR} by adding vertices $\{x_0^i\}_{i\in[n]}$; analogously we subdivide the edge between u^{TL} and u^{TR} by adding $\{x_1^i\}_{i\in[n]}$.
- Further, we add another `HB`-edge between u^{BL} and u^{TL} (resp., between u^{TR} and u^{BR}) and subdivide it h times by adding vertices $\{r_{j,R}^0\}_{j\in[2,h+2]}$ (resp., by adding $\{r_{j,L}^{n+1}\}_{j\in[2,h+2]}$).

We call the resulting graph of this construction (see Figure 1b) the *frame* F.

Now, we attach n Var-gadgets to the frame F.

- For each $i \in [n]$, we introduce a Var^i-gadget (as described in Section 3.2) and pairwise identify vertices u_0^i, w_0^i of Var^i with vertices x_0^i, x_1^i of the frame respectively.
- Lastly, we add *stairs* (interconnections) between `R`-paths of the neighboring Var-gadgets and the frame, i.e. for each $i \in [n+1]$ and $j \in [2, h+2]$, we add `R'`-edge $r_{j,R}^{i-1} r_{j,L}^i$. Thus, for each $i \in [n+1]$, we make stairs between the `R-right` path of Var^{i-1} (or, if $i = n+1$, with a subdivision of the frame's side $u^{TR}u^{BR}$) and the `R-left` path of Var^i (or, if $i = 1$, with a subdivision of the frame's side $u^{TL}u^{BL}$).

The weights of all new edges are again as specified in Section 3.2.

This finishes the construction of our *frame with n* Var*-gadgets* G' (see G' for $h = 4$, $n = 3$ in Figure 1b). Note that G' still lacks the clause edges (see in further Figure 2) and an interpretation of variable occurrences in clauses (the cells of further Figure 3).

So far, for simplicity, we allow ourselves to refer to Figure 1b to illustrate the defined graph G'. Observe the natural meaning the `R-left` and `R-right` paths in each gadget Var^i; in order to facilitate their connections to Var^{i-1} and to Var^{i+1}, `R-left` is naturally drawn to the left of `R-right`. On the other hand, the `B-/LB-pos` and `B-/LB-neg` paths of Var^i are symmetric and not adjacent outside of Var^i, and hence they can be flexibly drawn – `B-/LB-pos` to the left or to the right of `B-/LB-neg`; this is what will later define the truth value of the variable represented by Var^i.

3.3 The Full Reduction

Consider an instance $(\mathcal{C}, \mathcal{V})$ of Satisfiability (with $|\mathcal{C}| = \ell$, $|\mathcal{V}| = n$). We construct an instance (G, k) of Crossing Number as follows. See Figure 2 for a schematic representation.

- First, we introduce G', the frame with n Var-gadgets of height $h = 4\ell + n - 2$.
- Then, for each $i \in [n]$, we encode the occurrence of the variable x_i in clauses $\mathcal{C}$. For that purpose, for each $j \in [\ell]$, between `LB-pos` and `LB-neg` paths of the existing Var^i-gadget we add a *cell*. Each cell is defined by two horizontal `LB`-edges and 3 edges inside, depending on the type (pos, neg, neut) of the cell: pos if $x \in C$; neg if $\overline{x} \in C$; neut if neither x nor $\overline{x}$ is in C. A cell of each type is shown in Figure 3.

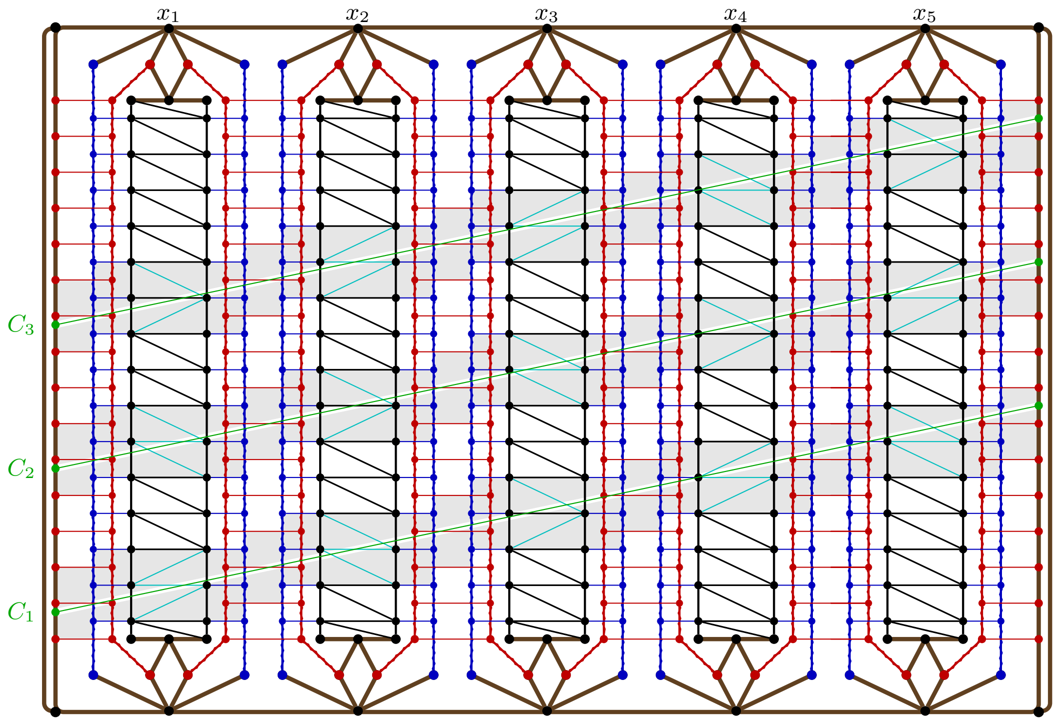
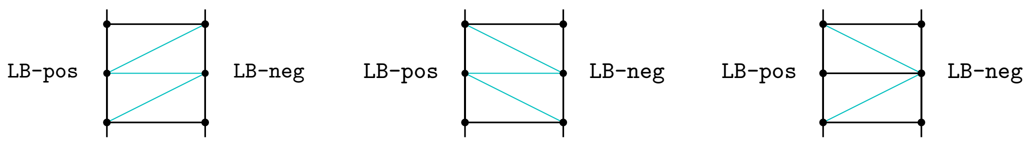


Figure 2 For an example instance of Satisfiability, given by $\mathcal{V} = \{x_1, x_2, x_3, x_4, x_5\}$ and $\mathcal{C} = \{(x_1 \vee \overline{x_2} \vee x_4 \vee \overline{x_5}), (\overline{x_1} \vee \overline{x_3} \vee x_5), (x_2 \vee x_3 \vee \overline{x_4})\}$; the depicted graph G is constructed as an input of the sought reduction to Crossing Number of G. Notice, in particular, the addition of the clause edges (drawn in green from left to right across the frame) and the shaded areas in which the clause edges will presumably be drawn.



(a) C_{pos}: $x \in C$. **(b)** C_{neg}: $\overline{x} \in C$. **(c)** C_{neut}: neither x nor $\overline{x}$ is in C.

Figure 3 Cell types; for cases of variable x occurrence in clause C.

Cells inside the same Var^i-gadget are separated with additional LB-edges as shown in Figure 2. For a formal description, we start with LB-edges (again, all weights are as specified in Section 3.2) inside Var-gadgets which separate cells. For each $i \in [n]$, we add

- an LB-path from $v^i_{1,N}$ to $v^i_{1+i,P}$ (below all cells of this Var^i-gadget), precisely on vertices $v^i_{1,N}$, $v^i_{2,P}$, $v^i_{2,N}$, …, $v^i_{i,N}$, $v^i_{1+i,P}$ in this order;
- another LB-path from $v^i_{4\ell+i-1,N}$ to $v^i_{h+2,P}$ (above all cells in Var^i), precisely on vertices $v^i_{4\ell+i-1,N}$, $v^i_{4\ell+i,P}$, $v^i_{4\ell+i,N}$, …, $v^i_{h+1,N}$, $v^i_{h+2,P}$; and
- for each $j \in [\ell-1]$, a LB-path on vertices $v^i_{4j+i-1,N}$, $v^i_{4j+i,P}$, $v^i_{4j+i,N}$, $v^i_{4j+i+1,P}$ (between cells number j and $j+1$).

After that, we add cells themselves in the bottom-up order. For all $j \in [\ell]$, we introduce two LB-edges $v^i_{4j+i-3,P}v^i_{4j+i-3,N}$ and $v^i_{4j+i-1,P}v^i_{4j+i-1,N}$, and then we proceed with encoding of our SATISFIABILITY instance $(\mathcal{C}, \mathcal{V})$ it the following way:

- if $x_i \in C_j$, we introduce a C_{pos} cell (Figure 3a), i.e. we add three C-edges as a path through $v^i_{4j+i-3,P}$, $v^i_{4j+i-2,N}$, $v^i_{4j+i-2,P}$, $v^i_{4j+i-1,N}$;
- in case $\overline{x_i} \in C_j$, we introduce a C_{neg} cell (Figure 3b), i.e. we add three C-edges as a path through $v^i_{4j+i-3,N}$, $v^i_{4j+i-2,P}$, $v^i_{4j+i-2,N}$, $v^i_{4j+i-1,P}$;
- lastly, if neither x_i nor $\overline{x_i}$ is in C_j, we introduce a C_{neut} cell (Figure 3c), formed by one LB-edge $v^i_{4j+i-2,P}v^i_{4j+i-2,N}$ and two C-edges $v^i_{4j+i-3,P}v^i_{4j+i-2,N}$ and $v^i_{4j+i-2,N}v^i_{4j+i-1,P}$.

- Finally, for each $j \in [\ell]$, we add a G-edge that corresponds to the clause C_j itself. Here, we subdivide two vertical HB-edges of the frame and connect these newly added vertices. Precisely, for each $j \in [\ell]$, we subdivide the edge between $r^0_{4j-2,R}$, $r^0_{4j-1,R}$ (on the left vertical side $u^{BL}u^{TL}$ of the frame, cf. Figure 1b) and the edge between $r^{n+1}_{4j+n-1,L}$, $r^{n+1}_{4j+n,L}$ (on the right vertical side $u^{BR}u^{TR}$). Note the shift in the indices of the subdivided edges, up by $n+1$ from left to right.

This concludes the construction of G.

The reduction returns (G, k) as an instance of CROSSING NUMBER where, for $h = 4\ell+n-2$,

$$k = 2n(2h+1)\omega^7 + 2n\ell\omega^6 + 4n\ell\omega^4 + 2n\sum_{j=2}^{h+1} j(j+1)\omega^4 + 2n\sum_{j=1}^{h+1} j(j+2)\omega^4 + n\ell\omega^2 + (\omega^2-1).$$

3.4 Drawings Claims

Until this point, all Figures were provided as an illustration without arguing why a certain drawing of the corresponding graph was selected. This subsection is dedicated to shed light on the conditions that necessarily have to be satisfied for a drawing $\mathcal{G}$ of the previously constructed instance (G, k) of the CROSSING NUMBER unless $\mathcal{G}$ is not a solution.

So, we are considering the constructed instance (G, k) of CROSSING NUMBER from Section 3.3. Following the way the graph G was introduced, we begin to formulate observations and claims that every drawing of the graph with at most k crossings has to satisfy. Naturally,

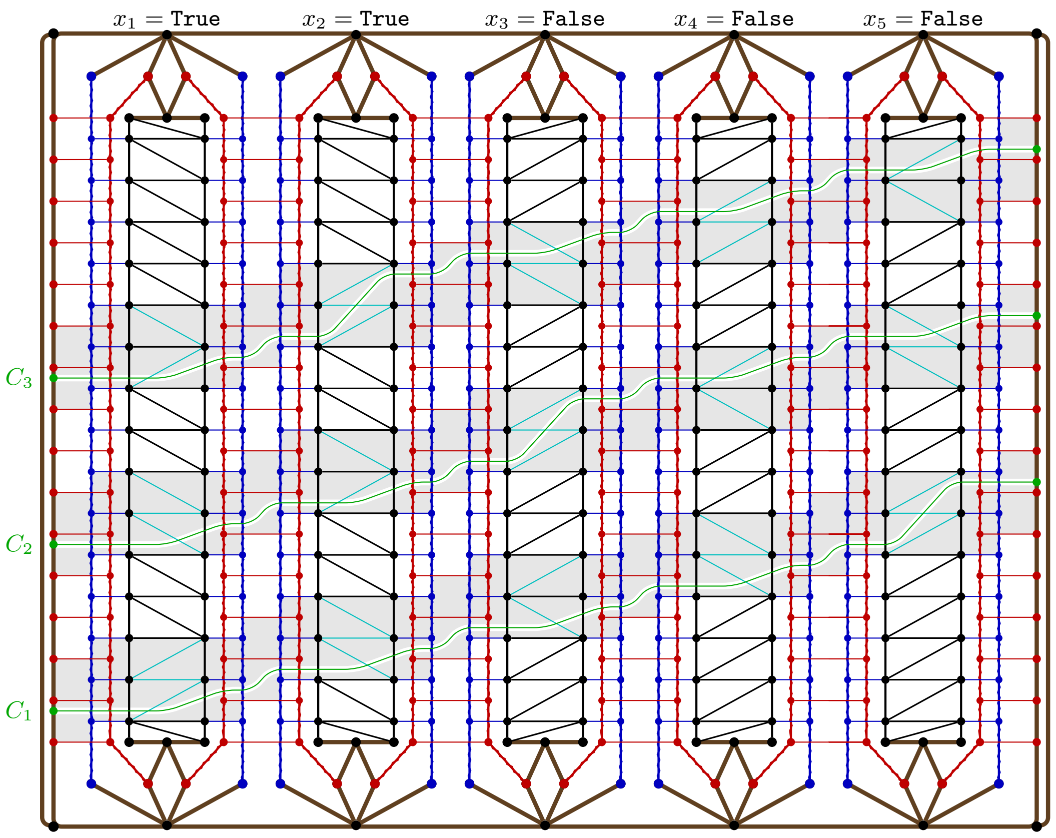


Figure 4 A drawing of the graph G from Figure 2, constructed from an instance of SATISFIABILITY given by $\mathcal{V} = \{x_1, x_2, x_3, x_4, x_5\}$ and $\mathcal{C} = \{(x_1 \vee \overline{x_2} \vee x_4 \vee \overline{x_5}), (\overline{x_1} \vee \overline{x_3} \vee x_5), (x_2 \vee x_3 \vee \overline{x_4})\}$. The depicted drawing $\mathcal{G}$ of G corresponds to the satisfying assignment $x_1 = x_2 = \texttt{True}$, $x_3 = x_4 = x_5 = \texttt{False}$. The clause $C_1 = (x_1 \vee \overline{x_2} \vee x_4 \vee \overline{x_5})$ is satisfied by the variable x_5 (observe that the G-edge of C_1 makes an extra jump-up in the drawing area of Var^5 to the right, yet crossing only one C-edge there – same as in other gadgets), $C_2 = (\overline{x_1} \vee \overline{x_3} \vee x_5)$ is satisfied by x_3 and $C_3 = (x_2 \vee x_3 \vee \overline{x_4})$ is satisfied by x_2. See Figure 5 for a drawing representing an unsatisfying assignment.

we start with conditions for the heaviest edges, for the frame and Var-gadgets, and after that we argue about the clauses encoding. Due to space restrictions, we leave proofs of the (*)-marked statements to the full arXiv version of the paper.

▶ **Observation 3.3.** **(*)** *If there exists a drawing $\mathcal{G}$ of G such that $\mathrm{cr}(\mathcal{G}) \leq k$, then $\mathcal{G}$ has no crossing that involves any of* `HB`*-edges.*

▶ **Observation 3.4.** **(*)** *Let H is a subgraph of G. If (H, k) is a no-instance of the CROSSING NUMBER, then (G, k) is a no-instance of the CROSSING NUMBER.*

Based on Theorem 3.4, we now show some claims that speak only about a subgraph of G. As in Section 3.2, let F be the frame and G' be the frame with n Var-gadgets (Figure 1) of the graph G; so, we are going to speak about properties which hold regardless of our clauses encoding in G. However, both next claims still follow from the same type of argument, namely, any crossing between the considered edges would be more costly than the selected value of k allows.

▶ **Claim 3.5.** **(*)** *If there exists a drawing $\mathcal{G}'$ of G' such that $\mathrm{cr}(\mathcal{G}') \leq k$, then there are no other crossings than crossings between* `R`*- or* `B`*-edges with* `R'`*- or* `B'`*-edges in $\mathcal{G}'$.*

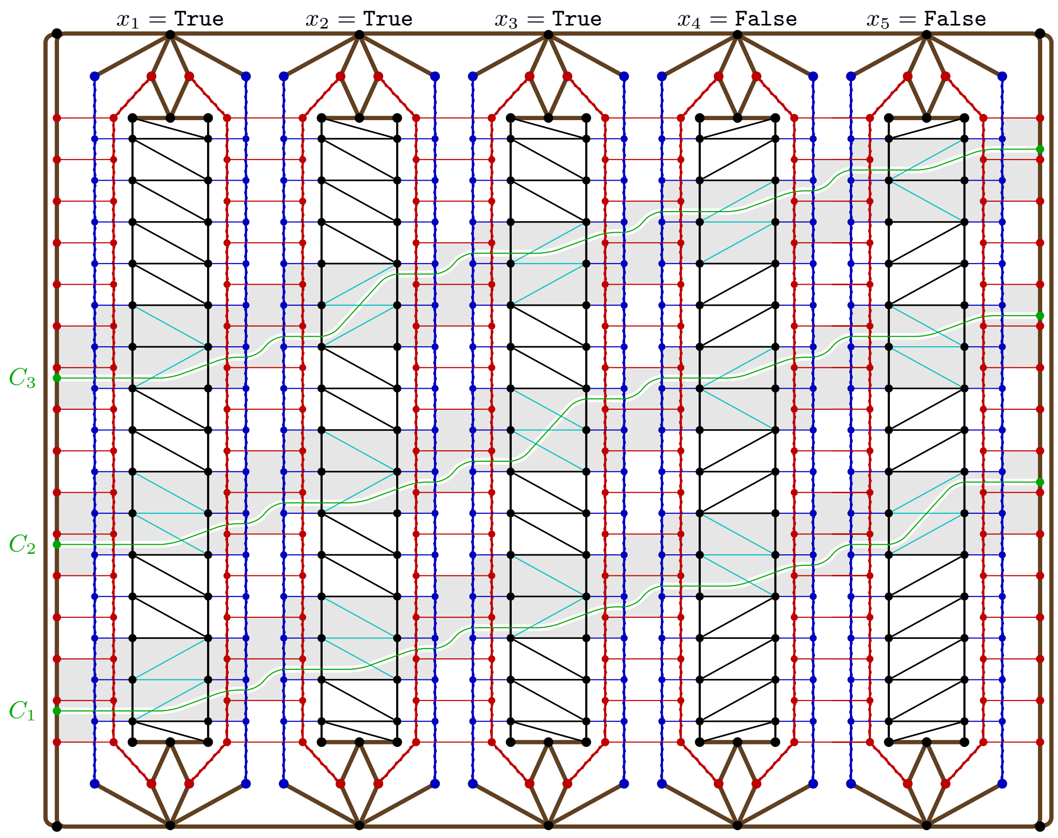


Figure 5 Another drawing of the graph G from Figure 2, constructed from an instance of Satisfiability given by $\mathcal{V} = \{x_1, x_2, x_3, x_4, x_5\}$ and $\mathcal{C} = \{(x_1 \vee \overline{x_2} \vee x_4 \vee \overline{x_5}), (\overline{x_1} \vee \overline{x_3} \vee x_5), (x_2 \vee x_3 \vee \overline{x_4})\}$, for comparison with the drawing in Figure 4.
The depicted drawing $\mathcal{G}'$ of G corresponds to the unsatisfying assignment $x_1 = x_2 = x_3 = \texttt{True}$, $x_4 = x_5 = \texttt{False}$. The clause $C_1 = (x_1 \vee \overline{x_2} \vee x_4 \vee \overline{x_5})$ is satisfied by the variable x_5 and $C_3 = (x_2 \vee x_3 \vee \overline{x_4})$ is satisfied by x_2, same as in Figure 4. The difference for the unsatisfied clause $C_2 = (\overline{x_1} \vee \overline{x_3} \vee x_5)$ is that here the G-edge of C_2 has no way to make the required extra jump-up without crossing more than one C-edge inside one of Var-gadgets (or other heavier edges). Hence, here the G-edge of C_2 makes extra crossing with two C-edges in the drawing area of Var^3, and this unavoidably leads to $\mathrm{cr}(\mathcal{G}') > k$.

▶ **Claim 3.6.** **(*)** *If a drawing $\mathcal{G}'$ of G' is a solution of the instance (G', k), then, for all $u, v \in V(G' \setminus F)$, both u and v are lying in the same face of the frame F in $\mathcal{G}'$, and the selection of such a face is uniquely predetermined.*

Theorem 3.6 allows us, without loss of generality, fix a drawing of the frame F as it is shown in Figure 1b. So, we define a positioning of left and right sides. This way, the fixed drawing of the frame determines a linear order of gadgets Var^i (from left to right following increasing order of $i \in [n]$).

Furthermore, we already almost fixed a drawing of Var-gadgets inside the frame. Still, the fact that we have not enough budget for any other crossing (namely, crossings between R'- and B'-edges of weight ω^3 have not been covered yet) needs some proper counting that we will provide further. By now, according to Theorem 3.5, the R/B/LB-paths of Var-gadgets are not allowed to cross each other in any solution if such a one exists.

Lastly, let us notice that the positioning of the vertical R-paths is also fixed.

▶ **Claim 3.7.** **(*)** *If a drawing $\mathcal{G}'$ of G' is a solution of the instance (G', k), then, for all $i \in [n]$, the* `R-left` *path of the* Var^i *is drawn to the left from both* `LB`*-paths of* Var^i*, while the* `R-right` *path is to the right.*

Now, let us check how crossings of `R`- and `B`-edges with `R'`- and `B'`-edges behave. Briefly saying, by the following Theorem 3.8, we show that crossings between the `R`- and `B`-edges are also predefined by a construction. For this purpose, while constructing the graph G, we played a bit with additional *adjustment* weight of order ω^1 for the vertical `R`- and `B`-edges. This adjustment weight selection forces the unique alternation of crossings exactly as shown in Figure 1b and subsequent figures.

▶ **Lemma 3.8.** **(*)** *If a drawing $\mathcal{G}'$ of G' is a solution of the instance (G', k), then each* `B`*-edge (*`R`*-edge) crosses exactly one* `R'`*-edge (*`B'`*-edge). The total weight of all crossings between* `B(R)`*- and* `R'(B')`*-edges adds* $2n\left((2h+1)\omega^7 + \sum_{j=2}^{h+1} j(j+1)\omega^4 + \sum_{j=1}^{h+1} j(j+2)\omega^4\right)$ *to the count* $\mathrm{cr}(\mathcal{G}')$*.*

Next, we return back to the full reduction graph G, and investigate properties of its admissible drawings.

▶ **Lemma 3.9.** **(*)** *If the crossing number of a drawing $\mathcal{G}$ of G does not exceed k, then the* `G`*-edges add at least $2n\ell\omega^6 + 4n\ell\omega^4 + n\ell\omega^2$ more crossings. In particular, every* `G`*-edge in $\mathcal{G}$ crosses precisely one* `C`*-edge of every* Var^i*-gadget.*

For each `G`-edge, we define its *cells area* as a connected region (union) of faces in a drawing $\mathcal{G}$ of G as it is shown in Figure 4 with a grey fill, i.e. for each $j \in [\ell]$, there is a single connected horizontal block of faces that includes all cells of the clause C_j and will correspond to the cells area for the `G`-edge which is j^{th} in bottom-up order. For each such a cells area we furthermore define its *up*- and *down-level* as the subsets of faces that are higher and lower, respectively, by subscripts of their vertices. To clarify the last point, by the next lemma we show that for each $j \in [\ell]$, if the crossing number of the drawing $\mathcal{G}$ of the graph G does not exceed k, then the j^{th} `G`-edge does not leave its cells area. Now, we proceed with that formally.

▶ **Lemma 3.10.** **(*)** *If the crossing number of a drawing $\mathcal{G}$ of G does not exceed k, then any* `G`*-edge cannot be drawn (even partially) outside of its cells area (see Figure 4).*

3.5 Correctness

Before proceeding with the correctness arguments, let us look back and see that, indeed, all drawing claims imply that if some drawing $\mathcal{G}$ of G with $\mathrm{cr}(\mathcal{G}) \le k$ exists, then almost all possible crossings are predefined. Practically the only freedom still left is the possibility to *flip* (independently of others) each Var-gadget. By *flips* of the Var-gadget we mean its two possible embeddings, which differ only by the order of `B`- and `LB`-paths, e.g. going from left to right in our Figures. As an example, let us consider Figure 4: here, we meet first from left

- either a pair of `B-pos, LB-pos` paths and then call such an embedding a *pos-side flip* (see Var^3-gadget in the middle in Figure 5);
- or a pair of `B-neg, LB-neg` paths and then call such an embedding a *neg-side flip* (see Var^3-gadget in the middle in Figure 4).

And this is exactly the intuition behind transferring a possible variable assignment from Satisfiability instance to a possible drawing of the Crossing Number instance, and back.

Suppose, given an instance $(\mathcal{C}, \mathcal{V})$ of Satisfiability, that the reduction from the previous subsection returns (G, k) as an instance of Crossing Number.

▶ **Lemma 3.11.** *If $(\mathcal{C},\mathcal{V})$ is a satisfiable instance of* Satisfiability*, then the graph G of the constructed instance (G,k) admits a drawing $\mathcal{G}$ such that $\mathrm{cr}(\mathcal{G})\le k$.*

Proof. Again, let G' be the frame with n Var-gadgets which is a subgraph of G. We start with the drawing of G' as specified by Figure 1, which actually corresponds to the "minimal drawing" investigated in Theorem 3.8. We are going to show that this drawing has exactly the same number of crossings as claimed in Theorem 3.8.

Indeed, for every $i\in[n]$, the `R-left` and `R-right` paths of the Var^i-gadget each are crossed by h `B'`-edges in our drawing, and the `B-pos` and `B-neg` paths of the Var^i-gadget each are crossed by $(h+1)$ `R'`-edges. Taking into account the alternating order of these crossings and the exact weights of the edges of these paths (see in Section 3.2), we count exactly $2\omega^3\left(h\omega^4+(h+1)\omega^4+\sum_{j=2}^{h+1}j(j+1)\omega^1+\sum_{j=1}^{h+1}j(j+2)\omega^1\right)$ crossings on Var^i. Summing over all $i\in[n]$, we get $2n\left((2h+1)\omega^7+\sum_{j=2}^{h+1}j(j+1)\omega^4+\sum_{j=1}^{h+1}j(j+2)\omega^4\right)$ crossings. There are no more crossings in our drawing of G'.

Now we add the cell gadgets of G to every Var-gadget, without additional crossings, and the `G`-edges denoted by e_j of all clauses $C_j\in\mathcal{C}$. The goal is to show that G is (can be) drawn with at most k crossings. Each `G`-edge e_j crosses every Var-gadget in total weight of $2\omega^6+4\omega^4+\Theta_{n,\ell}(\omega^1)$ crossings, together giving another $2n\ell\omega^6+4n\ell\omega^4+\Theta_{n,\ell}(\omega^1)$ crossings in our drawing. It only remains to estimate the weight of crossings of the `G`-edges e_j with the cell gadgets. If we can achieve the state that each `G`-edge e_j additionally crosses only one `C`-edge (of weight ω^2) of cell gadgets in every Var-gadget, then, comparing the full count to the reduction parameter definition

$$k=2n(2h+1)\omega^7+2n\ell\omega^6+4n\ell\omega^4+2n\sum_{j=2}^{h+1}j(j+1)\omega^4+2n\sum_{j=1}^{h+1}j(j+2)\omega^4+n\ell\omega^2+(\omega^2-1),$$

we see that it is enough to have $\Theta_{n,\ell}(\omega^1)\le\omega^2-1$. The latter holds true by our (sufficiently large) choice of the base weight ω.

It thus remains to show that aforementioned desired drawings of the `G`-edges e_j are simultaneously possible. Since the given Satisfiability instance is satisfiable, there is a truth assignment τ of $\mathcal{V}$ such that for every clause $C_j\in\mathcal{C}$, an occurrence of some variable $x_i\in\mathcal{V}$ in C_j satisfies C_j in τ. If the variable $x_i\in\mathcal{V}$, $i\in[n]$, is valued `True`, then the Var^i-gadget is pos-side flipped in our drawing of G', i.e., that the `LB-pos` path of Var^i is to the left of the `LB-neg` path of Var^i. If x_i is `False`, then the Var^i-gadget is neg-side flipped, i.e., with `LB-pos` to the right.

Now, for every $C_j\in\mathcal{C}$, the edge e_j is drawn in the down-level of its cells area (as defined in Section 3.4), until we reach the cell in the Var^i-gadget where i is the least index such that an occurrence of x_i satisfies C_j in τ. Then, inside this Var^i-gadget, the edge e_j jumps up to the up-level of its cells area, as depicted in Figure 3a if $x_i\in C_j$, resp. in Figure 3b if $\overline{x_i}\in C_j$. The Figure shows that this jump-up is also possible with crossing only one `C`-edge in the cells gadget of Var^i. For the rest, the edge e_j is drawn in the up-level of its cells area. This is the sought solution to a drawing of the reduction graph G with at most k crossings; indeed, the edge e_j arrives to the right side of the frame by $n+1$ levels higher that its beginning on the left side. ◀

▶ **Lemma 3.12.** *If G admits a drawing $\mathcal{G}$ such that $\mathrm{cr}(\mathcal{G})\le k$, then the original* Satisfiability *instance $(\mathcal{C},\mathcal{V})$ is satisfiable.*

Proof. If $\mathrm{cr}(\mathcal{G}) \le k$, then $\mathcal{G}$ satisfies the drawings claims of Section 3.4 and, in particular, Theorem 3.9 and Theorem 3.10. We define a valuation $\tau : \mathcal{V} \to \{\texttt{True}, \texttt{False}\}$ as follows; $x_i \in \mathcal{V}$ is set `True` if Var^i pos-side flipped in $\mathcal{G}$ (the `LB-pos` path of Var^i is drawn to the left of the `LB-neg` path of Var^i), and x_i is set `False` otherwise.

Now, for every clause $C_j \in \mathcal{C}$, the edge e_j in $\mathcal{G}$ starts in the down-level of its cells area and ends in the up-level. By Theorem 3.10, there has to be a Var^i-gadget, $i \in [n]$, such that e_j jumps up to the up-level of its cells area within the subdrawing of Var^i in $\mathcal{G}$. By the definition of the cell types in the construction of G, see in Figure 3, this is possible in accordance with Theorem 3.9 (only one crossing with a `C`-edge) only if $x_i \in C_j$ and x_i is set to `True`, or if $\overline{x_i} \in C_j$ and x_i is set to `False`. In other words, if every $C_j \in \mathcal{C}$ is satisfied by our valuation τ. And this completes the proof. ◀

3.6 On Path-width of the Resulting Instance

The last missing ingredient in the proof of Theorem 3.1 (and hence, of Theorem 1.1) is an estimate of the path-width and tree-width of the constructed instance G. To obtain it, we will use the cops-and-robber game characterization from Theorem 2.1.

We start with an auxiliary technical claim.

▶ **Lemma 3.13.** (*) *Let H be a graph whose vertex set is partitioned into m disjoint parts $V(H) = A_1 \cup \ldots \cup A_m$, and for some $a_i \le b_i$, $i \in \{1, \ldots, m\}$, let $A_i = \{v_{i,j} : a_i \le j \le b_i\}$. Assume that*

a) *each A_i induces a path in H in the natural order of vertices, i.e. $v_{i,a_i}, v_{i,a_i+1}, \ldots, v_{i,b_i}$;*
b) *if an edge $v_{i,j}v_{i',j'}$ exists in H for $i \ne i'$, then $|i - i'| = 1$ and $|j - j'| \le 1$; and*
c) *there are no indices $i \ne i', j \ne j'$ such that both $v_{i,j}v_{i',j'} \in E(H)$ and $v_{i,j'}v_{i',j} \in E(H)$.*[1]

Then there exists a valid monotone search strategy for the cop player on H using $m + 1$ cops against an invisible robber. Furthermore, this strategy can be assumed to start with the cop player occupying the vertex subset $\{v_{1,a_1}, \ldots, v_{m,a_m}\}$.

▶ **Proposition 3.14.** *For any given* Satisfiability *instance, the graph G constructed in Section 3.3 (for the proof of Theorem 3.1) is of path-width at most 12 and of tree-width at most 9.*

Proof. The proof would be finished if we find a monotone search strategy for the cop player on G using 13 cops against an invisible robber (implies path-width at most 12), and one using 10 cops against a visible robber (implies tree-width at most 9). We start with the former.

Firstly, let us place 8 cops (see Figure 1) on vertices $r^1_{1,L}$, $r^1_{h+3,L}$, u^{BL}, u^{TL} (left side of G) and $r^n_{1,R}$, $r^n_{h+3,R}$, u^{BR}, u^{TR} (right side of G). Note (see Figure 4) that such a placement of cops separates the set $U_0 \subseteq V(G)$ formed by the vertices of the left and right `HB`-paths of the frame, and of the `R-left` path of Var^1 and the `R-right` path of Var^n from the rest of the graph G. We can now use additional $4 + 1 = 5$ cops to search the subgraph induced by U_0 by using Theorem 3.13. Notice, however, that in this application the levels dealt with in Theorem 3.13 are shifted against the natural indexing from the construction of G.

After the previous initial phase, we continue the search by induction on $i = 1, 2, \ldots, n$. We assume 8 cops placed on vertices $r^i_{1,L}$, $r^i_{h+3,L}$, u^{i-1}_0, w^{i-1}_0 (the latter two being u^{BL}, u^{TL} if $i = 1$) and, again, $r^n_{1,R}$, $r^n_{h+3,R}$, u^{BR}, u^{TR}. Further, we assume that $V(\mathsf{Var}^{i-1})$ (if $i > 1$)

[1] The graph H can be easily pictured as having a planar drawing with the paths on $A_1, \ldots, A_m$ drawn vertically in order from left to right, and other edges joining only neighboring verticals on the same horizontal level or between two consecutive levels.

is already robber-free. We place 4 of the remaining cops onto vertices u_0^i, u_1^i, w_1^i, w_0^i, and subsequently lift the cops from $r_{1,L}^i$, $r_{h+3,L}^i$, u_0^{i-1}, w_0^{i-1}. We now have 5 free cops which can be used to search the `B-/LB-pos` and `B-/LB-neg` paths of Var^i by using Theorem 3.13. Then, we place 2 of the free cops onto $r_{1,L}^{i+1}$, $r_{h+3,L}^{i+1}$, and use the remaining 3 cops to search the `R-right` path of Var^i and the `R-left` path of Var^{i+1}, again by Theorem 3.13. After finishing previous, we may lift the cops from u_1^i and w_1^i, and we are back in the induction assumption with $i + 1$ instead of i.

It is easy to verify that the described procedure is a valid monotone search strategy against an invisible robber.

Regarding the tree-width subcase, we use knowledge of robber's position for a slight improvement of the previous search strategy. At the beginning, after placing cops on $r_{1,L}^1$, $r_{h+3,L}^1$, u^{BL}, u^{TL} and $r_{1,R}^n$, $r_{h+3,R}^n$, u^{BR}, u^{TR}, we look at whether the robber is trapped inside the set U_0 and if this is the case, we throw in 9-th cop to catch the robber in U_0 by Theorem 3.13 while using also the 4 cops starting on $r_{1,L}^1$, u^{BL}, $r_{1,R}^n$, u^{BR}.

In the induction phase, whenever the robber is trapped inside the `B-/LB-pos` and `B-/LB-neg` paths of Var^i, we may instead use the 4 cops from $r_{1,R}^n$, $r_{h+3,R}^n$, u^{BR}, u^{TR} to perform the catch by Theorem 3.13. We do likewise in the other subcase of the `R-right` path of Var^i and the `R-left` path of Var^{i+1}. When moving cops from the position on $r_{1,L}^i$, $r_{h+3,L}^i$, u_0^{i-1}, w_0^{i-1} to next u_0^i, u_1^i, w_1^i, w_0^i, we now move cops in pairs – place onto u_0^i, u_1^i and lift from $r_{1,L}^i$, u_0^{i-1}, and then do the other pairs. We observe that the maximum number of cops needed in this strategy is $4 + 4 + 2 = 10$, and this is tight at the moment just before trapping the robber in the `R-right`/`R-left` paths. ◀

4 Conclusion

We have shown that the Crossing Number problem is NP-hard for graphs of path-width 12 (and as a result, even of tree-width 9). It is worth to remark that, since the measures clique-width and rank-width are bounded by $\mathcal{O}(2^{\mathrm{tw}})$, their width decompositions are also too much general to help deal with the Crossing Number problem. On the other hand, there are more restrictive parameterizations worth trying, e.g. treedepth, distance to linear forest (distance to disjoint paths), feedback vertex set number (distance to a forest), or cut-width or bandwidth.

Barely any of the existing results could be extended for the parameters above. Thus, investigation whether any of them could yield fixed-parameter tractability (or W-hardness) of Crossing Number is an interesting venue to explore. However, as it is known that Crossing Number is in FPT when parameterized by the solution value (k) [4, 9], it only makes sense to investigate those parameters which do not bound the crossing number itself.

References

1 Therese Biedl, Markus Chimani, Martin Derka, and Petra Mutzel. Crossing number for graphs with bounded pathwidth. *Algorithmica*, 82(2):355–384, 2020. `doi:10.1007/S00453-019-00653-X`.

2 Sergio Cabello. Hardness of approximation for crossing number. *Discrete Comput. Geom.*, 49(2):348–358, March 2013. `doi:10.1007/S00454-012-9440-6`.

3 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`.

4 Éric Colin de Verdière and Thomas Magnard. An FPT algorithm for the embeddability of graphs into two-dimensional simplicial complexes. In *Proceedings of the 29th European*

Symposium on Algorithms (ESA), pages 32:1–32:17, 2021. See also arXiv:2107.06236. arXiv:2107.06236.

5 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

6 Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice-Hall, 1999.

7 Reinhard Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012. doi:10.1007/978-3-662-53622-3.

8 Michael R. Garey and David S. Johnson. Crossing number is NP-complete. *SIAM J. Algebr. Discrete Methods*, 4(3):312–316, September 1983. doi:10.1137/0604033.

9 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.

10 Petr Hliněný. Crossing number is hard for cubic graphs. *J. Comb. Theory, Ser. B*, 96(4):455–471, 2006. doi:10.1016/j.jctb.2005.09.009.

11 Petr Hliněný. Complexity of anchored crossing number and crossing number of almost planar graphs. *CoRR*, abs/2306.03490, 2023. doi:10.48550/arXiv.2306.03490.

12 Petr Hliněný and Gelasio Salazar. On hardness of the joint crossing number. In *ISAAC*, volume 9472 of *Lecture Notes in Computer Science*, pages 603–613. Springer, 2015. doi:10.1007/978-3-662-48971-0_51.

13 Petr Hliněný and Abhisekh Sankaran. Exact crossing number parameterized by vertex cover. In Daniel Archambault and Csaba D. Tóth, editors, *Graph Drawing and Network Visualization - 27th International Symposium, GD 2019, Prague, Czech Republic, September 17-20, 2019, Proceedings*, volume 11904 of *Lecture Notes in Computer Science*, pages 307–319. Springer, 2019. doi:10.1007/978-3-030-35802-0_24.

14 Michael J. Pelsmajer, Marcus Schaefer, and Daniel Stefankovic. Crossing numbers and parameterized complexity. In Seok-Hee Hong, Takao Nishizeki, and Wu Quan, editors, *Graph Drawing, 15th International Symposium, GD 2007, Sydney, Australia, September 24-26, 2007. Revised Papers*, volume 4875 of *Lecture Notes in Computer Science*, pages 31–36. Springer, 2007. doi:10.1007/978-3-540-77537-9_6.

15 Paul D. Seymour and Robin Thomas. Graph searching and a min-max theorem for tree-width. *J. Comb. Theory B*, 58(1):22–33, 1993. doi:10.1006/jctb.1993.1027.

16 Paul Turán. A note of welcome. *Journal of Graph Theory*, 1(1):7–9, 1977. doi:10.1002/jgt.3190010105.

17 Meirav Zehavi. Parameterized analysis and crossing minimization problems. *Comput. Sci. Rev.*, 45:100490, 2022. doi:10.1016/J.COSREV.2022.100490.