Upward Planar Drawings with Three and More Slopes

Jonathan Klawitter and Johannes Zink(✉)

Universität Würzburg, Würzburg, Germany
{klawitter,zink}@informatik.uni-wuerzburg.de

Abstract. We study upward planar straight-line drawings that use only a constant number of slopes. In particular, we are interested in whether a given directed graph with maximum in- and outdegree at most k admits such a drawing with k slopes. We show that this is in general NP-hard to decide for outerplanar graphs ($k = 3$) and planar graphs ($k \geq 3$). On the positive side, for cactus graphs deciding and constructing a drawing can be done in polynomial time. Furthermore, we can determine the minimum number of slopes required for a given tree in linear time and compute the corresponding drawing efficiently.

Keywords: Upward planar · Slope number · NP-hardness

1 Introduction

One of the main goals in graph drawing is to generate clear drawings. For visualizations of directed graphs (or digraphs for short) that model hierarchical relations, this could mean that we explicitly represent edge directions by letting each edge point upward. We may also require a planar drawing and, if possible, we would thus get an upward planar drawing. For schematic drawings, we try to keep the visual complexity low, for example by using only few different geometric primitives [28] – in our case few slopes for edges. If we allow two different slopes we get orthogonal drawings [14], with three or four slopes we get hexalinear and octilinear drawings [37], respectively. Here, we combine these requirements and study upward planar straight-line drawings that use only few slopes.

Upward Planarity. An *upward planar drawing* of a digraph G is a planar drawing of G where every edge is drawn as a monotonic upward curve. We call G *upward planar* if it admits an upward planar drawing and *upward plane* if it is equipped with an upward planar embedding. Note that an upward planar embedding, given by the edge order around each vertex, is necessarily *bimodal*, that is, each cyclic sequence can be split into two contiguous subsequences of incoming edges and outgoing edges [14]. Di Battista and Tamassia [15] have shown that if a digraph is upward planar, then it also admits an upward planar straight-line drawing.

While upward planarity testing is an NP-complete problem for general digraphs [23], there exist several FPT algorithms [9,19,25] and polynomial-time

H. C. Purchase and I. Rutter (Eds.): GD 2021, LNCS 12868, pp. 149–165, 2021.
https://doi.org/10.1007/978-3-030-92931-2_11

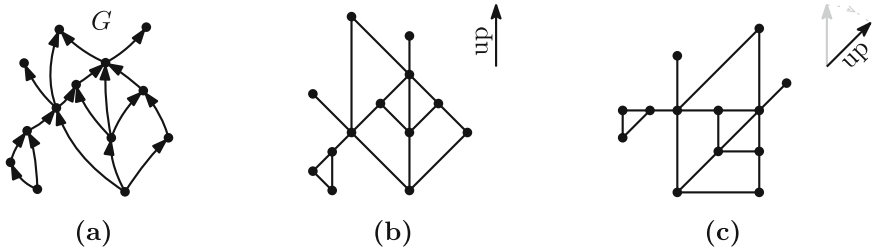


Fig. 1. (a) A digraph G with **(b)** upward planar 3-slope drawing; **(c)** drawing rotated by 45°. For readability, edge directions are now given implicitly.

algorithms for special classes, e.g., for single source digraphs [6], outerplanar digraphs [39], series-parallel digraphs [19], and triconnected digraphs [5]. If the embedding is given, upward planarity can be tested in polynomial time [5].

k-slope Drawings. A *k-slope drawing* of a (not necessarily directed) graph G is a straight-line drawing of G where every edge is drawn with one of at most k different slopes; see Fig. 1a and b. The *slope number* of G is the smallest k such that G admits a k-slope drawing. If only (upward) planar drawings are allowed, the number is called the *(upward) planar slope number* of G. The general and planar slope number have been studied extensively in the past for a variety of classes [8,16,17,20,21,26,27,31,33–35,38,42], Recently, also the interest in upward planar drawings on few slopes has grown. For example, allowing one bend per edge, Bekos et al. [3] studied so-called bitonic *st*-graphs and complementarily Di Giacomo et al. [18] considered series-parallel digraphs. Brückner et al. [8] studied level-planar drawings with a fixed slope set. Older works include results by Czyzowicz et al. [12,13] on lattices and several results for trees [1,2,7,10].

In a companion paper to this one, Klawitter and Mchedlidze [29] show that it can be decided in linear time whether a given upward plane digraph admits an upward planar 2-slope drawing. For the variable embedding scenario and two slopes, they give a linear-time algorithm for single-source digraphs, a quartic-time algorithm for series-parallel digraphs, and an FPT algorithm for general digraphs.

Here, we study the problem of whether a digraph admits an upward planar k-slope drawing for any k – with a special focus on the next natural case $k = 3$. Clearly, we can presume that G has maximum in- and outdegree at most k. Note that a 2-slope drawing can be sheared in the direction of one slope without affecting the length of edges drawn with the other slope. The fact that this does not hold for three or more slopes introduces interesting new geometric aspects.

For the choice of k specific slopes, we propose three settings. In the *general* setting, any set of k distinct slopes can be chosen. In the *uniform (angles)* setting, we use slopes with angles in $\{i \cdot \pi/k \mid i \in \{0, \dots, k-1\}\}$ clockwise (cw) from the x-axis. In the *regular grid* setting, we define a set of slopes as follows. Let c be the middle grid point of a $W \times W$ square grid, where $W = 2\lceil \log_2 k \rceil - 1$. Pick any k distinct slopes that you get from connecting c to any of the other grid points.

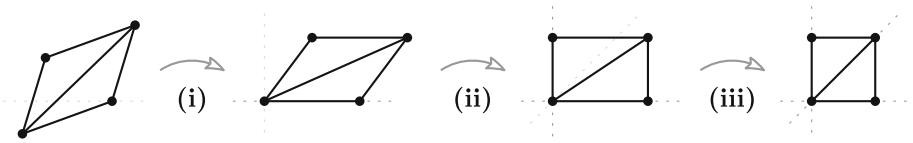


Fig. 2. Given a drawing using any set of three slopes, we can **(i)** Rotate, **(ii)** Shear, and **(iii)** Scale it to only use the slope set $\{\uparrow, \nearrow, \rightarrow\}$.

We remark that these slopes are contained in (an extended version of) the W-th Farey sequence[1]. Uniform angles naturally lead to more balanced drawings with more rotational symmetry, which we find more visually appealing. The downside of this setting is that we cannot always use grid points of the regular 2D grid. E.g., for $k = 6$, the third slope is $\tan(\frac{2}{6}\pi) = \sqrt{3}$, which is an irrational number. Therefore, we assume henceforth for uniform angles a computation and representation model that can handle implicit coordinates or alternatively real numbers. On the other hand in the regular grid setting, we get unbalanced edge angles and irrational edge lengths. Since all of these settings have their natural justification, we consider all of them. However, note that $k = 3$ is a special case because no matter which three slopes we pick, they can be affinely transformed to the slopes of the angles $\{45°, 90°, 135°\}$ as illustrated in Fig. 2. Hence, we restrict considerations to this slope set. For illustrative purposes however, we often rotate drawings by 45° cw and thus use the slope set $\{\uparrow, \nearrow, \rightarrow\}$; see Fig. 1c.

A *k-slope assignment* of a digraph G assigns each edge of G one of k slopes. If G is upward plane, we call a k-slope assignment of G *consistent* if the assignment complies with the cyclic edge order around each vertex; e.g., for $k = 3$, if a vertex has three incoming edges, they need to be assigned the slopes $\rightarrow$, $\nearrow$, and $\uparrow$ in counterclockwise (ccw) order. Clearly, if an upward plane embedding does not admit a consistent k-slope assignment, it also does not admit an upward planar k-slope drawing.

Contribution. We mainly contribute three results to the study of upward planar k-slope drawings. Firstly, we classify the upward planar slope numbers of directed ordered and unordered trees and show how to construct a drawing. Secondly, we show that for cactus graphs we can construct an upward planar k-slope drawing in polynomial time. Thirdly, we show that it is NP-hard to decide whether a given upward outerplanar digraph admits an upward planar 3-slope drawing. We extend the NP-hardness to $k > 3$ but restrict the graph class to upward planar (except for $k = 4$ if no embedding is given).

For statements marked with "⋆", a proof is available in the full version [30].

[1] The W-th Farey sequence is the sequence of all completely reduced fractions where the nominator and denominator is at most W in order of increasing size.

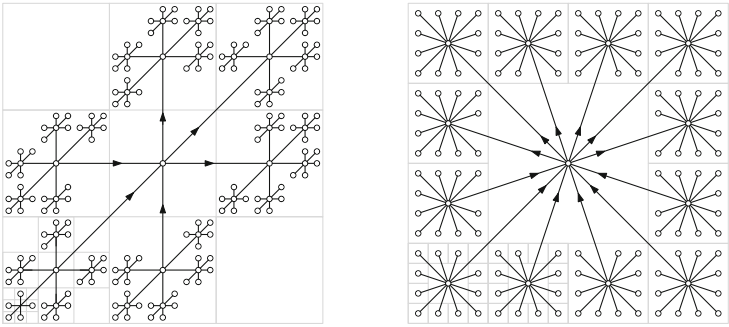


Fig. 3. Upward k-slope drawings of unordered tree $T_{3,3}$ with $k = 3$ and $T_{6,2}$ with $k = 6$, respectively, on the grid.

2 Trees

In this section, we consider upward planar k-slope drawings of directed trees. While our trees are in general not rooted, results for rooted trees can be derived or are partially already known [2,7]. For drawings of trees in the fixed and variable embedding scenario, the terms *ordered* tree, where a planar embedding is specified, and *unordered* tree, where it is not, are used. Note that naturally every unordered tree is upward planar, while an ordered tree is upward planar if and only if its embedding is bimodal. Not surprisingly, the upward planar slope number of an unordered directed tree T equals the maximum in- and outdegree of T; compare this to the planar slope number of $\lceil \Delta/2 \rceil$ of an unordered undirected tree with max. Degree Δ [20]. To show this, we draw T as subgraph of a larger, regular tree $T_{k,h}$ for $h \geq 1$ where every non-leaf vertex has in- and outdegree k and each leaf has distance h to a central vertex. To draw $T_{k,h}$ on a grid with k slopes, we adopt the strategy of Bachmaier et al. [1] for complete rooted trees; see Fig. 3. Alternatively, $T_{k,h}$ can be drawn with k uniform angles; see Fig. 4.

Theorem 1 ($\star$). *Let T be an unordered directed tree with maximum indegree and outdegree k. Then T admits an upward planar k-slope drawing on the regular grid and another upward planar k-slope drawing with uniform angles.*

Note that this recursive drawing procedure requires an exponential-size drawing area (or an exponential edge-length ratio). Different from ordered directed trees (see below), it is not clear whether exponential area is necessary for some unordered directed trees when restricting to k slopes. A $\Theta(n \log n)$ area suffices for an arbitrary number of slopes [22].

Next, let T be a bimodally ordered directed tree. With the drawing approach from unordered trees, it is clear that to determine the upward planar slope number of T it suffices to find a consistent k-slope assignment for T with minimal k. In this regard, note that the maximum in- and outdegree are natural lower bounds but that the choice of the (minimal) slope for an edge uv cannot be determined locally at u and v. For example, the edge vw in Fig. 5a is the third incoming edge at w

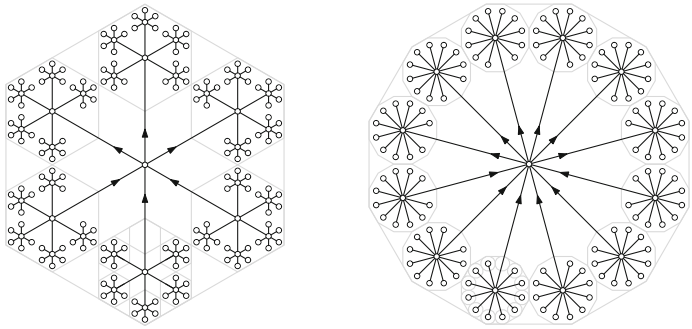


Fig. 4. Upward k-slope drawings of unordered tree $T_{3,3}$ with $k = 3$ and $T_{6,2}$ with $k = 6$, respectively, with uniform angles.

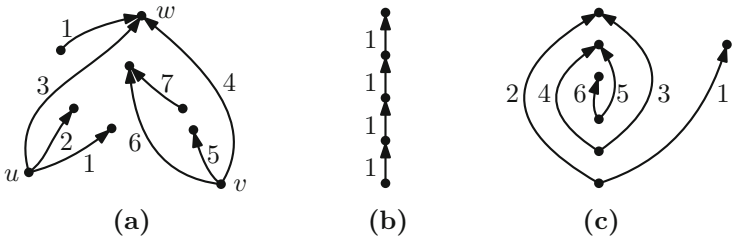


Fig. 5. **(a)** The minimal slope of an edge is not determined locally; **(b)** A directed path requires only one slope; **(c)** An alternating path requires $n - 1$ slopes.

but requires at least slope 4, since its preceding edge uw already requires slope 3 at u. This effect only appears along alternating intervals of incoming and outgoing edges. Hence we have the following observation – see also Fig. 5b and c.

Observation 2. *The upward planar slope number of ordered directed trees with n vertices, $n \geq 2$, is bounded within 1 and $n - 1$ and these bounds are tight.*

However, a simple greedy algorithm finds a consistent k-slope assignment for T, where k is minimal. The algorithm first identifies all edges that can have slope 1, e.g., if an edge uv is the sole incoming edge at v and the ccw first outgoing edge at u. Then, it gives any subsequent edge xy the maximum of the slope of its ccw preceding outgoing edge at x and its ccw preceding incoming edge at y plus one. Since linear time suffices for this and any additional bookkeeping, we get:

Theorem 3. *The upward planar slope number of an ordered directed tree can be determined in linear time.*

Corollary 1. *Let T be an ordered directed tree with maximum in-/outdegree k. We can decide in linear time whether T admits an upward planar k-slope drawing.*

Regarding the area requirement for k-slope drawings of ordered directed trees, we want to remark that Quapil and Jungeblut [40] have shown that there are ordered trees with a spiral structure that require exponential area for 3 slopes.

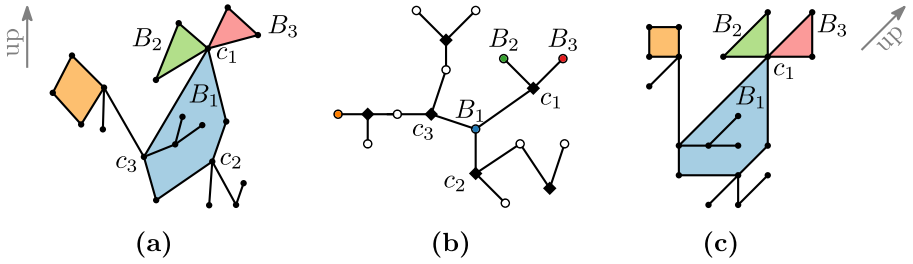


Fig. 6. **(a)** A cactus graph G; **(b)** Block-cut tree of G; **(c)** 3-slope drawing of G.

3 Cactus Graphs

In this section, we show that it can be decided in polynomial time whether a given cactus digraph admits an upward planar k-slope drawing, both in the fixed and the variable embedding scenario. We use a dynamic program on the block-cut tree of the cactus that computes combinable k-slope assignments for each block.

Recall that a block-cut tree $\mathcal{T}$ of a graph G has a vertex for each *block* (biconnected component) and each cut vertex of G and an edge between a block B and a cut vertex c if c is part of B; see Fig. 6b. Let G be a cactus graph. Note that in a block-cut tree $\mathcal{T}$ of G each block vertex is either a cycle or an edge – we thus distinguish between *cycle blocks* and *edge blocks*. The block-cut tree of G can be computed in linear time [41]. For G to admit an upward planar k-slope drawing, each block of $\mathcal{T}$ must be drawable under constraints imposed by other blocks. For example in Fig. 6a–c and $k = 3$, under a fixed embedding the two edges of the block B_1 incident to the cut vertex c_1 need the slopes $\nearrow$ and $\uparrow$ because of the blocks B_2 and B_3. Our strategy is thus as follows.

Algorithm. In the first phase we run a dynamic program (described below) on the blocks of $\mathcal{T}$ to find a consistent slope assignment for each block such that the blocks are combinatorially combinable. If successful, we enter the second phase, where we compute drawings of the blocks that are geometrically combinable. In the last phase we put all block drawings together.

Let G be a cactus and let $\mathcal{T}$ be the block-cut tree of G. We pick an arbitrary block vertex, say B', of $\mathcal{T}$ as root and direct all edges towards B_ℓ. As a result, each block vertex B (except B') has one outgoing edge towards a cut vertex c. We then say c is the *anchor* of B. Let B be a cycle block with anchor c and let e and e' be the edges of B incident to c. Suppose we have a slope assignment for B. Then the *anchor type* $t_c(B)$ of c for B is defined as the slopes of e and e' and if they are incoming or outgoing edges at c; see Fig. 7. For an edge block B with edge e, the *anchor type* $t_c(B)$ describes the slope of e and if e is incoming or outgoing at c. For cycle blocks and edge blocks, there are $2k \cdot (2k-1)$ and $2k$ different anchor types, respectively.

For a block vertex B with anchor c, a *feasible tuple* $\tau_B = \langle \phi_B, t_c(B) \rangle$ consists of a consistent k-slope assignment ϕ_B of B and an anchor type $t_c(B)$ of c. A

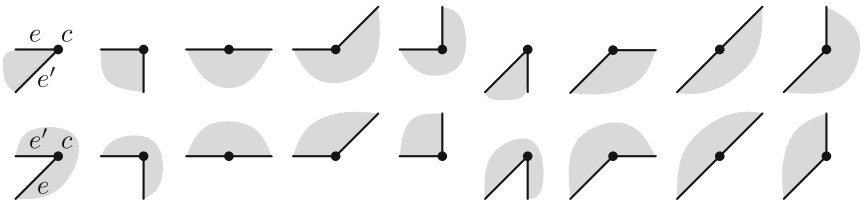


Fig. 7. A subset of the anchor types of a cycle block for $k = 3$.

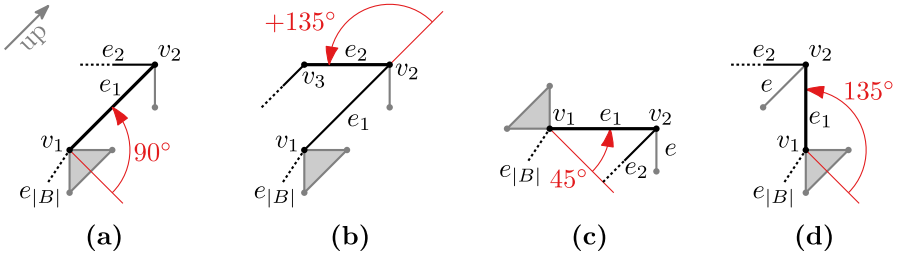


Fig. 8. Computing the possible slopes and rotations for each edge of a cycle.

feasible set for B is a maximal set of feasible tuples for B that have pairwise different anchor types. We process $\mathcal{T}$ in a post-order traversal. For each block we compute the feasible set based on the feasible tuples of its descendant blocks.

Combinatorial Realization. Computing the feasible set of a cycle block B with anchor c works as follows. Let B be the cycle $(c = v_1, e_1, v_2, e_2, \ldots, v_{|B|}, e_{|B|}, v_1)$ – if an embedding is given, let the order be cw around the inner face. For every possible slope of e_1, we walk around B once and store all tuples of possible slopes and how far we rotated from the start. We start with e_1 and consider the $\mathcal{O}(1)$ feasible tuples of descendant blocks of B anchored at v_1 and v_2. In the example of Fig. 8a for $k = 3$, assuming a fixed embedding, the edge e_1 can only have slope $\nearrow$ and we have thus rotated 90° (starting from the original x-axis). For this tuple $(\nearrow, 90°)$, the edge e_2 in Fig. 8b has also only one possible slope, namely $\rightarrow$, and the rotation increases by 135°. However, in the variable embedding scenario, e_1 can also have slopes $\rightarrow$ and $\uparrow$, see Fig. 8c and d. In general, for an edge e_i, $i \in \{2, \ldots, |B|\}$, we consider for all tuples of e_{i-1} how e_i can proceed; again we consider the feasible tuples of descendant blocks of B at v_i and v_{i+1}. For each found tuple of e_i we store a pointer to the tuple(s) of e_{i-1} it is based on.

When we handle $e_{|B|}$, we reject all tuples that do not result in a 2π rotation if the embedding is given or with $\pm 2\pi$ if no embedding is given. This ensures that the cycle has a geometric realization [11]. Combining the slope of e_1 and $e_{|B|}$ as well as whether the rotation is $+2\pi$ or -2π yields an anchor type of B at c. We backtrack from the tuple of $e_{|B|}$ to find a consistent slope assignment of B.

Since the edge e_{i-1} can have at most $\mathcal{O}(k|B|)$ possible rotation values, which imply a slope each, we can compute all possible tuples of e_i in $\mathcal{O}(k|B|)$ time.

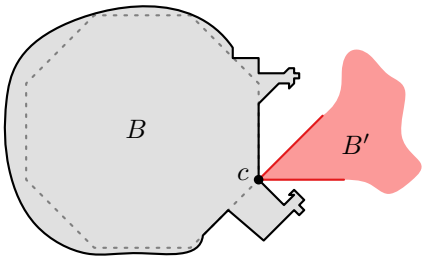


Fig. 9. When drawing a single block, we make sure that the anchor point c lies at a $2k$-gon edge within the algorithm by Culberson and Rawlins [11]; here $k = 4$.

Thus, a single feasible tuple of the whole block B can be computed in $\mathcal{O}(k|B|^2)$ time and all $\mathcal{O}(k)$ feasible tuples of B in $\mathcal{O}(k^2|B|^2)$ time.

Geometric Realization. Suppose we have found a consistent k-slope assignment for every cycle. In the variable embedding scenario, we now know whether and how cycles nest. We thus re-root $\mathcal{T}$ such that the root block lies on the outer face. Next, we describe how to obtain a drawing of a cycle block B as a polygon that does not intersect the edges of its parent block B' at its anchor point c.

We describe this only for the uniform angles setting and leave it as an open question for the regular grid setting. Given any sequence σ of rational angles (i.e., a rational number times π) that sum up to $\pm 2\pi$, Culberson and Rawlins [11] describe an algorithm that outputs a polygon with σ as turning angles. Internally, their so-called *Turtlegon* algorithm works as follows. It defines a base angle α as the greatest common divisor of π and all angles in σ; in our case this is π/k. Larger angles are split into sequences of $\pm\alpha$ resulting in a new angle sequence σ'. W.l.o.g. let σ' contain more angles $+\alpha$ than $-\alpha$. Using some of the αs, their algorithm draws a regular $(2\pi/\alpha)$-gon (in our case $2k$-gon). To accommodate additional angles in between, it inserts exponentially shrinking detours at the corners of the $(2\pi/\alpha)$-gon. In the end, we get the original larger angles from merging the smaller angles [11].

The difficulty for us when employing this $\mathcal{O}(k|B|)$ time algorithm, is to ensure that the edges of the parent block B' can reach the anchor point c without intersecting the polygon of B. This might be impossible if c lies within a spiral inside a detour. However, we can avoid this if we let an incident edge of c be a side of the $2k$-gon (this is always possible because we can pick an appropriate set of α angles of σ' for the $2k$-gon) and if we let each detour edge shrink by a sufficiently large factor (e.g., $k|B|$); see Fig. 9.

The running time of this step is in $\mathcal{O}(k|B|)$. Since each vertex is in at most k blocks, we have that $\sum_{i=1}^{\ell} |B_i| \leq kn$. Hence, the total running time is in $\mathcal{O}(k^2 n)$.

Putting Blocks Together. We start with a drawing of the root block. We then recursively draw each child (in a BFS-like order) such that its anchor point coincides with the corresponding vertex of the parent polygon and scale down

the drawing of the child block such that the appended polygon does not intersect the existing drawing. Note that it always suffices to scale down each child to the size of the minimum distance of any two vertices within in the parent polygon. We can determine vertex pairs of minimum and maximum distance for a block B in $\mathcal{O}(|B| \log |B|)$ time and then place and scale each polygon in linear time.

The total running time is dominated by the dynamic program, which runs in $\mathcal{O}(k^2|B|^2)$ time for one block B and, hence, in $\mathcal{O}(k^4n^2)$ time for all blocks.

Theorem 4. *Let G be an upward planar (or plane) cactus graph with maximum in- and outdegree k. It can be constructively tested in $\mathcal{O}(k^4n^2)$ time whether G admits an upward planar k-slope drawing in the uniform angles setting.*

For the regular grid setting, we cannot use the algorithm by Culberson and Rawlins [11] because we have irrational multiples of π as turning angles. For a sequence of general turning angles, the algorithm by Hartley [24] computes a polygon realizing that sequence. However, it is not immediately clear how to guarantee that the edges of the parent polygon at the anchor point are not intersected. For general polygons, we believe that we can iteratively shrink the spikes to resolve potential intersections. Since such a procedure involves some more technicalities, we leave it as an open question for now.

4 Outerplanar and Planar Graphs

In this section, we show that for any constant $k \geq 3$ deciding whether an upward planar (for $k = 3$, outerplanar) digraph admits an upward planar k-slope drawing is NP-hard. Except for $k = 4$, this hardness holds true regardless of whether we prescribe an embedding or not. However, it remains open if the problem is also NP-complete. Containment in NP is not immediately clear, since it is open whether some graphs require irrational (or super-polynomial precise) coordinates for any k-slope drawing. We first describe our NP-hardness reduction for embedded outerplanar graphs for 3 slopes. Afterwards, we show how this extends to the variable embeddings and to larger k.

We reduce from Planar Monotone 3-SAT [4], an NP-complete version of 3-SAT, where the three literals of each clause are all either negated or unnegated – from now on called *negative* and *positive* clauses, respectively. Moreover, the incidence graph[2] has a planar drawing where the vertices are rectangles, the edges are vertical straight-line segments, the variables are arranged on a horizontal line, the positive clauses are above, and the negative clauses are below this line; see Fig. 11a. For a given formula F and a rectangular drawing of its incidence graph, we construct a corresponding upward outerplanar digraph G_F, which can only be drawn upward planar with 3 slopes if F is satisfiable. Our construction follows ideas of Nöllenburg [36] and Kraus [32] and utilizes the following observations.

[2] The incidence graph of a SAT formula has a vertex for each variable and clause and an edge for each occurrence of a variable in a clause between the corresponding vertices.

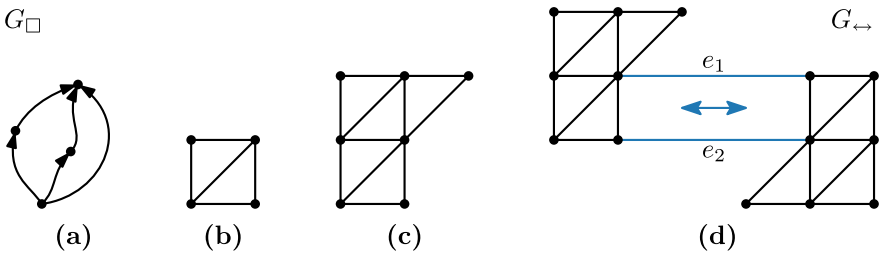


Fig. 10. **(a)** The digraph $G_\square$ admits only an upward 3-slope drawing as square **(b)**; **(c)** By combining copies of $G_\square$ and triangles we can build larger rigid structures; **(d)** Upward planar 3-slope drawing of the digraph $G_\leftrightarrow$.

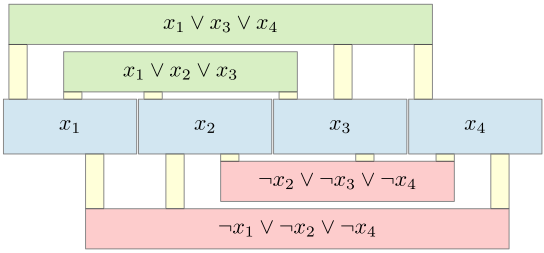
Up to scaling and mirroring diagonally, $G_\square$ in Fig. 10a admits an upward planar 3-slope drawing only as an outerplanar square as in Fig. 10b. We can attach multiple squares (and triangles) to each other as in Fig. 10c. The drawing of such a bigger digraph is unique up to scaling and mirroring diagonally. If the squares form a tree, the drawing is outerplanar. We refer to these squares as *unit squares*, since, once set, the side lengths for all attached squares are the same. To allow a certain small degree of freedom, we exploit the following.

Lemma 1 (⋆). *In any upward planar 3-slope drawing of $G_\leftrightarrow$ (see Fig. 10d)*

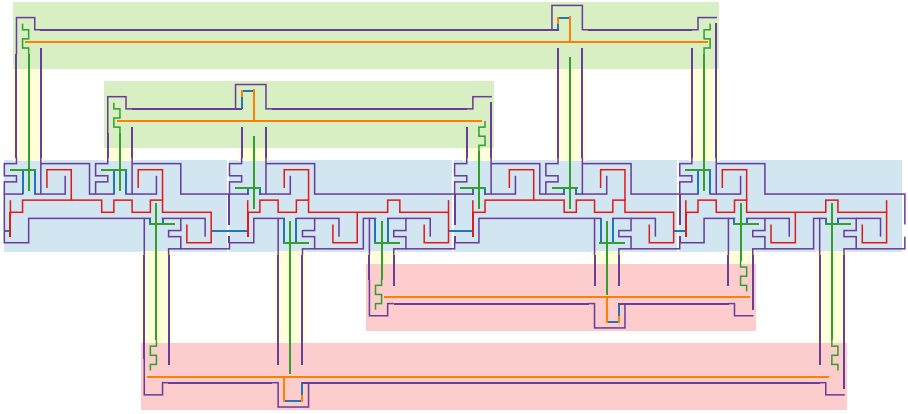
- *the edges e_1 and e_2 are parallel and have the same arbitrary length $\ell > 0$,*
- *all edges are oriented as in Fig. 10d up to mirroring along a diagonal axis,*
- *and all vertical and horizontal edges (besides e_1 and e_2) have the same lengths, as well as all diagonal edges.*

With this construction kit of useful (sub)graphs in hand, we build a graph whose upward planar drawings represent the satisfying truth assignments for F. The **high-level construction** is depicted in Fig. 11b. We construct, for each variable x_i, a specific digraph – the *variable gadget for x_i*. Similarly, for each clause c_j, there is a specific digraph – the *clause gadget for c_j*. All gadgets mainly consist of chains of $G_\square$s. For a drawing, this enforces a rigid frame structure built from unit squares. We glue all variable gadgets together in a row and connect variable and clause gadgets by *edge gadgets* such that the composite graph remains upward outerplanar (see Fig. 11b) and $G_\square$s are drawn as unit squares.

A **variable gadget** is depicted in Fig. 12. Its base structure is the (violet) frame composed of chains of unit squares. The core element is the (red) central chain of unit squares (with a few side-arms), which has one degree of flexibility, namely, moving as a whole to the left or to the right without leaving the frame structure of the gadget. It looks and behaves a bit like a pipe cleaning brush that is stuck inside the frame but can be moved a bit back and forth. Hence, we call it a *brush*. It is connected via a $G_\leftrightarrow$ to the brush of the previous variable gadget (see Fig. 12a/d) and the first brush is connected to the frame via a $G_\leftrightarrow$. This allows only a horizontal shift of the brushes, but no vertical movement relative



(a) Rectangular incidence graph drawing of a PLANAR MONOTONE 3-SAT formula F.



(b) Outerplanar drawing of the digraph G_F obtained from (a). Chains of unit squares are drawn as straight line segments. The variable/clause/edge gadgets occupy the areas of their corresponding rectangles. Here, x_1 and x_4 are set to false (brush on the left), while x_2 and x_3 are set to true (brush on the right)

Fig. 11. Schematic example for our NP-hardness reduction.

to its anchor point at the frame structure. Note that the horizontal position in any variable gadget is independent of those in all other gadgets. If the brush is positioned to the very left (right), the corresponding variable is set to false (true).

For each occurrence of a variable in a positive clause, we have a construction as depicted in Fig. 12b. There, a long chain of (green) $G_{\square}$s – from now on called *bolt* – is attached to the frame structure via two $G_{\hookrightarrow}$s, which allow only a vertical, but no horizontal shift. The bolt has on its left side an arm, which can only be placed in one of two pockets of the frame. It can always be placed in the upper pocket, which pushes the bolt outwards with respect to the variable gadget (into an edge and then a clause gadget). It can only be placed in the lower pocket if the brush is shifted to the very right (i.e. set to true) – then the bolt can "fall" into a cove of the brush. For each occurrence of a variable in a negative clause, we have this construction upside-down, such that the bolt can be pulled into the variable gadget only if the brush is shifted to the very left (i.e. set to false).

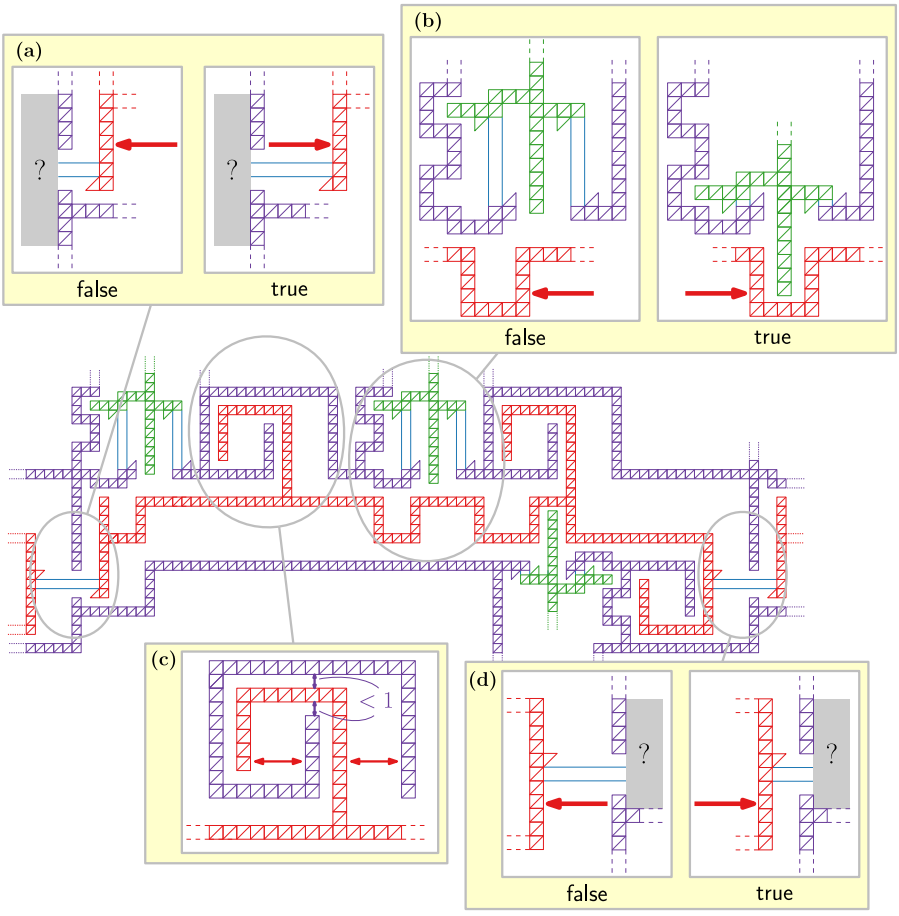


Fig. 12. A variable gadget, which is contained in two positive and one negative clauses. The brush is positioned to the left and, thus, the variable is set to false. (Color figure online)

Note that, to maintain outerplanarity of the whole construction, the frame structure is not contiguous, but connected by $G_{\leftrightarrow}$s and the arms of the bolts. Hence, the frame structure decomposes into many components that have fixed relative horizontal positions and their unit squares have the same side lengths. However, the components can shift up and down relative to each other. To keep this vertical shift small enough not to affect the correct functioning of our reduction, we use, for each such component, the construction depicted in Fig. 12c. The chain of brushes has no vertical flexibility and serves as a base ground for an "anchor" of the frame. The frame can move less than one unit up or down unless it violates planarity. If the frame would be shifted up enough to be completely above the brush, it would get in conflict with the adjacent bolt.

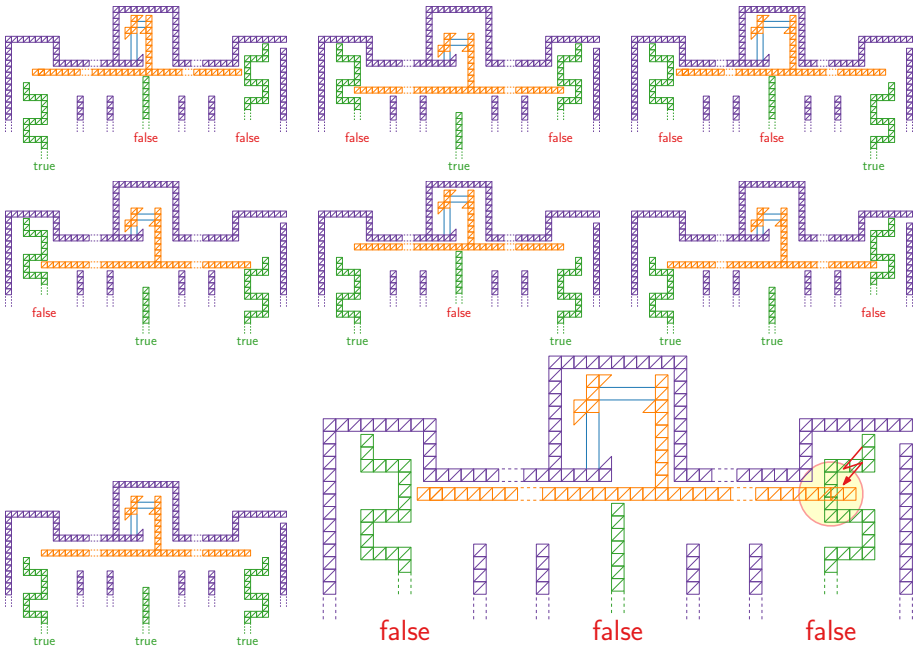


Fig. 13. Positive clause gadget in 8 configurations. (Color figure online)

An **edge gadget** consists of only three straight chains – two frame segments and a bolt in the middle. Their purpose is to synchronize the distance of the clause gadgets to the variable gadgets and to preserve the size of the unit squares. Several edge gadgets are depicted on yellow background color in Fig. 11b.

A **clause gadget** for a positive clause is depicted in Fig. 13. Within a frame, which is connected at six points to the frames of three edge gadgets, there is a horizontal (orange) bar, which is attached via two $G_{\leftrightarrow}$s to the frame – one $G_{\leftrightarrow}$ allows a horizontal, the other allows a vertical shift. It resembles a crane that can move up and extend its arm, while it holds the horizontal bar on a vertical (orange) rope. The three bolts from the corresponding variable gadgets reach into the clause gadget. The lengths of these bolts is chosen such that, if they are pushed out of their variable gadget and into the clause gadget, they only slightly fit inside the gadget. Depending on whether each of the bolts is pushed into the clause gadget or pulled out of it, we have eight possible configurations (with sufficiently small vertical slack). They represent the eight possible truth assignments to a clause. In Fig. 13, we illustrate that in each configuration, we can accommodate the horizontal bar in an upward planar 3-slope drawing of the clause gadget – except for the case when all three bolts push into the clause gadget, which represents the truth assignment false to all contained variables. A negative clause gadget uses the same construction, but mirrored vertically.

Note that, since we have only connected $G_{\square}$s and $G_{\leftrightarrow}$s, the planar embedding of the constructed graph is unique up to mirroring along a diagonal axis. Therefore, our reduction holds true also for the variable embedding scenario and we conclude:

Theorem 5 ($\star$). *Deciding whether an upward outerplanar digraph admits an upward planar 3-slope drawing is NP-hard with and without a given embedding.*

Next, we describe how to extend our NP-hardness reduction to more than 3 slopes. There, however, we use only upward planar instead of upward outerplanar graphs. Observe that, if we fix the embedding and give up outerplanarity, we can add dummy leaves to each vertex to occupy all but the originally used slopes. Since any 3 slopes can be projected to $\{\uparrow, \nearrow, \rightarrow\}$ and we block all other slopes, our arguments work for all sets of k slopes and the reduction remains correct.

Last, we show that our NP-hardness reduction remains applicable for $k > 4$ in the variable embedding setting. This leaves $k = 4$ in the variable embedding setting as the only open case. Again, we do this be extending the graph such that it has only planar embedding up to mirroring along a diagonal axis.

Assume for now that k is an odd number; in the full version [30], we consider otherwise. From the given k slopes, we pick the 3 middle slopes to host the graph of the hardness construction described before. For simplicity, we visualize these 3 middle slopes again as $\{\uparrow, \nearrow, \rightarrow\}$ and the other slopes in quadrants II and IV around a vertex. The key idea is to occupy the unused slopes at each vertex by *fans* and *beaters* as depicted in Fig. 14 instead of simple leaves. Fans are appended to the outside of each vertex if the angle that has been formed in the old construction is $\geq 180°$. For each other remaining slope at each vertex, we add a beater. This is a graph obtained from the wheel graph W_{2k+1} of which one spoke e^* is broken free. This enforces an order on the spokes and, hence, we prescribe the slope of e^*. Note that the whole beater could be mirrored leaving two possible slopes for e^*. However, this is unproblematic since in our construction the "mirrored" slope is also occupied by a beater or a fan. In the full version [30], we prove that this suffices to enforce a desired embedding. Though we lost upward outerplanarity, note that the underlying undirected graph remains outerplanar.

Theorem 6 ($\star$). *Deciding whether an upward planar digraph with maximum in- and outdegree k admits an upward planar drawing with k slopes is NP-hard for $k \geq 3$ if a bimodal embedding is given and is NP-hard for $k \in \mathbb{N}^+ \setminus \{1, 2, 4\}$ if no embedding is given. This holds true for all choices of k slopes.*

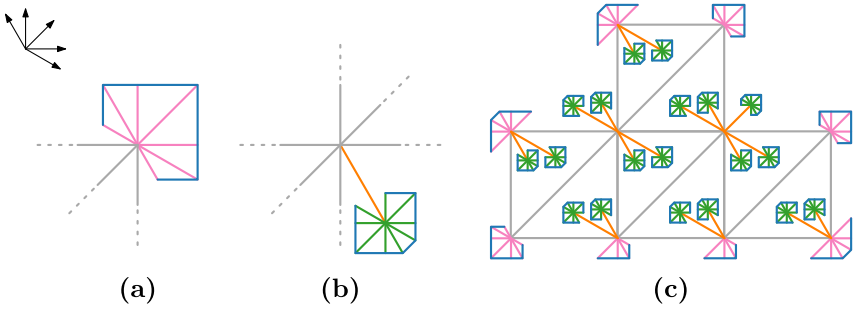


Fig. 14. Example for $k = 5$ slopes: **(a)** A fan and **(b)** a beater. **(c)** We add fans and beaters to each vertex of the graph such that all unused slopes are occupied.

References

1. Bachmaier, C., Brandenburg, F.J., Brunner, W., Hofmeier, A., Matzeder, M., Unfried, Thomas: Tree drawings on the hexagonal grid. In: Tollis, I.G., Patrignani, M. (eds.) GD 2008. LNCS, vol. 5417, pp. 372–383. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00219-9_36
2. Bachmaier, C., Matzeder, M.: Drawing unordered trees on k-grids. J. Graph Algorithms Appl. **17**(2), 103–128 (2013). https://doi.org/10.7155/jgaa.00287
3. Bekos, M.A., Di Giacomo, E., Didimo, W., Liotta, G., Montecchiani, F.: Universal slope sets for upward planar drawings. In: Biedl, T., Kerren, A. (eds.) GD 2018. LNCS, vol. 11282, pp. 77–91. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-04414-5_6
4. de Berg, M., Khosravi, A.: Optimal binary space partitions for segments in the plane. Int. J. Comput. Geom. Appl. **22**(3), 187–206 (2012). https://doi.org/10.1142/s0218195912500045
5. Bertolazzi, P., Di Battista, G., Liotta, G., Mannino, C.: Upward drawings of triconnected digraphs. Algorithmica **12**(6), 476–497 (1994). https://doi.org/10.1007/BF01188716
6. Bertolazzi, P., Di Battista, G., Mannino, C., Tamassia, R.: Optimal upward planarity testing of single-source digraphs. SIAM J. Comput. **27**(1), 132–169 (1998). https://doi.org/10.1137/S0097539794279626
7. Brunner, W., Matzeder, M.: Drawing ordered (k - 1)–any treees (k–grids. In: Brandes, U., Cornelsen, S. (eds.) GD 2010. LNCS, vol. 6502, pp. 105–116. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-18469-7_10
8. Brückner, G., Krisam, N.D., Mchedlidze, T.: Level-planar drawings with few slopes. In: Archambault, D., Tóth, C.D. (eds.) GD 2019. LNCS, vol. 11904, pp. 559–572. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-35802-0_42
9. Chan, H.: A parameterized algorithm for upward planarity testing. In: Albers, S., Radzik, T. (eds.) ESA 2004. LNCS, vol. 3221, pp. 157–168. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30140-0_16
10. Crescenzi, P., Di Battista, G., Piperno, A.: A note on optimal area algorithms for upward drawings of binary trees. Comput. Geom. **2**(4), 187–200 (1992). https://doi.org/10.1016/0925-7721(92)90021-J

11. Culberson, J.C., Rawlins, G.J.E.: Turtlegons: generating simple polygons for sequences of angles. In: Proceedings of 1st Symposium on Computational Geometry (SoCG 1985), pp. 305–310. ACM (1985)
12. Czyzowicz, J.: Lattice diagrams with few slopes. J. Comb. Theory Ser. A **56**(1), 96–108 (1991). https://doi.org/10.1016/0097-3165(91)90025-C
13. Czyzowicz, J., Pelc, A., Rival, I.: Drawing orders with few slopes. Discrete Math. **82**(3), 233–250 (1990). https://doi.org/10.1016/0012-365X(90)90201-R
14. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice-Hall, Hoboken (1999)
15. Di Battista, G., Tamassia, R.: Algorithms for plane representations of acyclic digraphs. Theor. Comput. Sci. **61**(2), 175–198 (1988). https://doi.org/10.1016/0304-3975(88)90123-5
16. Di Giacomo, E., Liotta, G., Montecchiani, F.: Drawing outer 1-planar graphs with few slopes. J. Graph Algorithms Appl. **19**(2), 707–741 (2015). https://doi.org/10.7155/jgaa.00376
17. Di Giacomo, E., Liotta, G., Montecchiani, F.: Drawing subcubic planar graphs with four slopes and optimal angular resolution. Theor. Comput. Sci. **714**, 51–73 (2018). https://doi.org/10.1016/j.tcs.2017.12.004
18. Di Giacomo, E., Liotta, G., Montecchiani, F.: 1-bend upward planar slope number of SP-digraphs. Comput. Geom. **90**, 101628 (2020). https://doi.org/10.1016/j.comgeo.2020.101628
19. Didimo, W., Giordano, F., Liotta, G.: Upward spirality and upward planarity testing. SIAM J. Discrete Math. **23**(4), 1842–1899 (2010). https://doi.org/10.1137/070696854
20. Dujmović, V., Eppstein, D., Suderman, M., Wood, D.R.: Drawings of planar graphs with few slopes and segments. Comput. Geom. **38**(3), 194–212 (2007). https://doi.org/10.1016/j.comgeo.2006.09.002
21. Dujmović, V., Suderman, M., Wood, D.R.: Graph drawings with few slopes. Comput. Geom. **38**(3), 181–193 (2007). https://doi.org/10.1016/j.comgeo.2006.08.002
22. Frati, F.: On minimum area planar upward drawings of directed trees and other families of directed acyclic graphs. Int. J. Comput. Geom. Appl. **18**(03), 251–271 (2008). https://doi.org/10.1142/S021819590800260X
23. Garg, A., Tamassia, R.: On the computational complexity of upward and rectilinear planarity testing. SIAM J. Comput. **31**(2), 601–625 (2001). https://doi.org/10.1137/S0097539794277123
24. Hartley, R.I.: Drawing polygons given angle sequences. Inf. Process. Lett. **31**(1), 31–33 (1989)
25. Healy, P., Lynch, K.: Two fixed-parameter tractable algorithms for testing upward planarity. Int. J. Found. Comput. Sci. **17**(05), 1095–1114 (2006). https://doi.org/10.1142/S0129054106004285
26. Jelínek, V., Jelínková, E., Kratochvíl, J., Lidický, B., Tesař, M., Vyskočil, T.: The planar slope number of planar partial 3-trees of bounded degree. Graphs Comb. **29**(4), 981–1005 (2013). https://doi.org/10.1007/s00373-012-1157-z
27. Keszegh, B., Pach, J., Pálvölgyi, D.: Drawing planar graphs of bounded degree with few slopes. SIAM J. Discrete Math. **27**(2), 1171–1183 (2013). https://doi.org/10.1137/100815001
28. Kindermann, P., Meulemans, W., Schulz, A.: Experimental analysis of the accessibility of drawings with few segments. J. Graph Algorithms Appl. **22**(3), 501–518 (2018). https://doi.org/10.7155/jgaa.00474
29. Klawitter, J., Mchedlidze, T.: Upward planar drawings with two slopes. Arxiv report (2021), http://arxiv.org/abs/2106.02839

30. Klawitter, J., Zink, J.: Upward planar drawings with three slopes. Arxiv report (2021). http://arxiv.org/abs/2103.06801
31. Knauer, K., Micek, P., Walczak, B.: Outerplanar graph drawings with few slopes. Comput. Geom. **47**(5), 614–624 (2014). https://doi.org/10.1016/j.comgeo.2014.01.003
32. Kraus, R.: Level-außenplanare zeichnungen mit wenigen steigungen. Bachelor Thesis, University of Würzburg (2020)
33. Lenhart, W., Liotta, G., Mondal, D., Nishat, R.I.: Planar and plane slope number of partial 2-trees. In: Wismath, S., Wolff, A. (eds.) GD 2013. LNCS, vol. 8242, pp. 412–423. Springer, Cham (2013). https://doi.org/10.1007/978-3-319-03841-4_36
34. Mukkamala, P., Pálvölgyi, D.: Drawing cubic graphs with the four basic slopes. In: van Kreveld, M., Speckmann, B. (eds.) GD 2011. LNCS, vol. 7034, pp. 254–265. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-25878-7_25
35. Mukkamala, P., Szegedy, M.: Geometric representation of cubic graphs with four directions. Comput. Geom. **42**(9), 842–851 (2009). https://doi.org/10.1016/j.comgeo.2009.01.005
36. Nöllenburg, M.: Automated drawing of metro maps. Diploma Thesis, University of Karlsruhe (TH) (2010)
37. Nöllenburg, M., Wolff, A.: Drawing and labeling high-quality metro maps by mixed-integer programming. IEEE Trans. Visual. Comput. Graph. **17**(5), 626–641 (2011). https://doi.org/10.1109/TVCG.2010.81
38. Pach, J., Pálvölgyi, D.: Bounded-degree graphs can have arbitrarily large slope numbers. Electron. J. Comb. **13**(1), N1 (2006)
39. Papakostas, A.: Upward planarity testing of outerplanar dags (extended abstract). In: Tamassia, R., Tollis, I.G. (eds.) GD 1994. LNCS, vol. 894, pp. 298–306. Springer, Heidelberg (1995). https://doi.org/10.1007/3-540-58950-3_385
40. Quapil, V.: Bachelor thesis on slope drawings. In: Jungeblut, P (ed.) Karlsruhe Institute of Technology (2021)
41. Tarjan, R.: Depth-first search and linear graph algorithms. SIAM J. Comput. **1**(2), 146–160 (1972). https://doi.org/10.1137/0201010
42. Wade, G.A., Chu, J.H.: Drawability of complete graphs using a minimal slope set. Comput. J. **37**(2), 139–142 (1994). https://doi.org/10.1093/comjnl/37.2.139