Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 24, no. 1, pp. 1–19 (2020)
DOI: 10.7155/jgaa.00515

Order-preserving Drawings of Trees with Approximately Optimal Height (and Small Width)

Johannes Batzill[1] *Therese Biedl*[2]

[1]Karlsruhe Institute of Technology, Karlsruhe, Germany.
[2]David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, Ontario N2L 1A2, Canada.

Abstract

In this paper, we study how to draw trees so that they are planar, straight-line and respect a given order of edges around each node. We focus on minimizing the height, and show that we can always achieve a height of at most $2pw(T) + 1$, where $pw(T)$ (the so-called *pathwidth*) is a known lower bound on the height of the tree T. Hence our algorithm provides an asymptotic 2-approximation to the optimal height. The width of such a drawing may not be a polynomial in the number of nodes. Therefore we give a second way of creating drawings where the height is at most $3pw(T)$, and where the width can be bounded by the number of nodes. Finally we construct trees T that require height $2pw(T) + 1$ in all planar order-preserving straight-line drawings.

Submitted:	Reviewed:	Revised:	Accepted:
December 2018	May 2019	July 2019	October 2019
	Final:	Published:	
	October 2019	January 2020	

Article type:	Communicated by:
Regular paper	C.D. Tóth

Work done while JB was visiting Univ. of Waterloo. Research of TB supported by NSERC.
E-mail addresses: batzilljohannes@gmail.com (Johannes Batzill) biedl@uwaterloo.ca (Therese Biedl)

1 Introduction

Let T be a tree, i.e., a connected graph with n nodes and $n-1$ edges. Trees occur naturally in many applications, e.g., family trees, organizational charts, directory structures, etc. To be able to understand and study such trees, it helps to create a visualization, i.e., to draw the tree. This is the topic of this paper.

There are many different requirements that one could impose on tree-drawings: Does the drawing have to be *[strictly] upward* (parents are [strictly] above their children), *order-preserving* (a fixed cyclic order of edges at each node is respected), *straight-line* (edges are drawn as straight line segments), and do we care about minimizing the *area* or the number of *layers*? One could further distinguish by the maximum degree of the tree and by imposing further conditions on how edges can be drawn. All tree-drawing algorithms require that the drawing is *planar* (has no crossings), and nodes are placed at grid points.

In consequence, there are many results concerning how to draw trees. A good overview of results up till 2014 was given by Di Battista and Frati [6]. In a recent breakthrough paper, Chan [4] lowered the long-standing area-bounds for some of the drawing models. It has also only recently been shown that minimizing the area is NP-hard in some of the upward tree-drawing models [3, 1].

In this paper, we focus on the number of layers needed for planar, straight-line, order-preserving drawings of trees of arbitrary degrees, but we do not require drawings to be upward. (We often omit "planar, straight-line, order-preserving", as we study no other drawing-types except during the literature-review.) Formally, a drawing is said to be a *k-layer drawing* if all y-coordinates (possibly after translation) are in the range $\{1, \dots, k\}$; we also say that it has *height* k and *layers* $1, \dots, k$ (from top to bottom). We do not always require x-coordinates to be integers, but when we do, then the drawing has *width* w if all x-coordinates are in the range $\{1, \dots, w\}$.

It has been known since 1992 that any n-node tree has a drawing on $\log_2(n+1)$ layers [5, 6]. (This, and many of the papers listed below, bound the width, not the height, but since we do not require drawings to be upward this is the same after a 90° rotation.) This bound is tight for the complete ternary tree [5] and hence cannot be improved in terms of n. However, some trees can be drawn on significantly fewer layers. To this end, Suderman [12] showed that every tree T can be drawn on $\lceil \frac{3}{2} pw(T) \rceil$ layers, where $pw(T)$ denotes the *pathwidth* of a tree T (defined in Section 2). Since any tree T requires at least $pw(T)$ layers [7], Suderman hence gives an asymptotic $\frac{3}{2}$-approximation on the number of layers required by a tree. ("Asymptotic" means that up to a constant term his number of layers is within a factor of $\frac{3}{2}$ of the optimum.) Later Mondal et al. showed that the minimum number of layers required for a tree can be found in polynomial time [10].

All the results listed above were for *unordered* trees, i.e., the drawing algorithm is allowed to rearranged the subtrees around each node arbitrarily. In contrast to this, we study *order-preserving* drawings. Recall that this means that we are given an *ordered tree*, i.e., a fixed cyclic order of edges around each node, and the drawing must respect this. Garg and Rusu [8] showed that any ordered tree has an order-preserving upward drawing with $O(\log n)$ layers and area $O(n \log n)$; the number of layers can be seen to be at most $3 \log n$. In a recent paper [2] we showed that the number of layers can also be bound by $2rpw(T) - 1$ (where $rpw(T)$ is the so-called rooted pathwidth); this is at most $4pw(T) + 1$ and hence an asymptotic 4-approximation for the number of layers in an ordered tree-drawing.

In this paper, we give a different construction for order-preserving drawings of trees which is inspired by the approach of Suderman [12]. We show that every tree T has an order-preserving drawing on $2pw(T)+1$ layers; this is hence an asymptotic 2-approximation algorithm on the number of layers for order-preserving drawings. We also show that for some trees T, we cannot hope to do better, i.e., T needs $2pw(T) + 1$ layers.

In the construction that we give here, the width is potentially very large. We therefore give another (and in fact, much simpler) construction that draws a tree T on $3pw(T)$ layers and for which the width is n. Furthermore, our drawing is a so-called rectangle-of-influence drawing (see [9]). Since any tree has $pw(T) \leq \log_3(2n+1)$ [11], our results are never worse than the ones of Garg and Rusu, and frequently better.

2 Preliminaries

The *pathwidth* is a well-known graph-parameter, usually defined as the smallest k such that a super-graph of the graph is an interval graph that can be colored with $k + 1$ colors. For trees, there exists an equivalent simpler definition [12] given below. For a tree T and a path P, we use $T \setminus P$ to denote the forest obtained by deleting all vertices of P.

Definition 1 *The* pathwidth $pw(T)$ *of a tree* T *is 0 if* T *is a single node, and* $\min_P \max_{T' \subseteq T \setminus P} \{1 + pw(T')\}$ *otherwise. Here the minimum is taken over all paths* P *in* T*, and the maximum is taken over all subtrees* T' *of* $T \setminus P$*.*

A path where the minimum of Definition 1 is achieved is called a *main path*. Note that we may assume that a main path connects a leaf to a leaf, for otherwise making it longer gives another main path. In particular, if T is not a single-node tree, then the main path contains at least one edge.

We draw trees by splitting them at a path, drawing subtrees recursively, and merging them. The following terminology is helpful. For a tree T and a strict sub-tree C, a *linkage-edge* is an edge e of T with exactly one end in C (called the *linkage-node*) and the other end not in C (called the *anchor-node*). Usually C will be a connected component of $T \setminus P$ for some path P, and then the linkage-edge of C is unique. An *external linkage-edge* of a tree T is an edge e that belongs to an (unspecified) super-tree T' of T and has exactly one end in T and the other in $T' - T$.

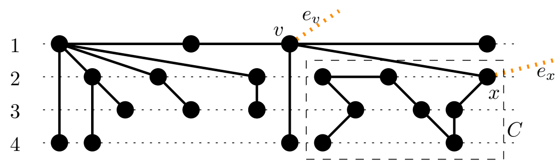

Figure 1: An HVA-drawing (defined in Section 3) on 4 layers. (v, x) is the linkage-edge of C, with v the anchor-node and x the linkage-node. The drawing is e_v-exposed (presuming the order in the supertree is respected), but not e_x-exposed since x is not unique among the rightmost nodes.

To be able to merge subtrees, we need to specify conditions on subtrees, concerning not only where linkage-nodes are placed, but also where external linkage-edges could be drawn such that edge-orders are respected.

Definition 2 *Let Γ be an order-preserving drawing of an ordered tree T, and let $e = (v, u)$ be an external linkage-edge of T with $v \in T$.*

We say that Γ is e-exposed *if v is in the top or bottom level, and after inserting e by drawing outward (up or down) from v, the drawing respects the edge-order at v in the super-tree of T that defined the external linkage-edge.*

We say that Γ is e-reachable *if v is drawn either as unique leftmost or as unique rightmost node, and after inserting e by drawing outward (left or right) from v, the drawing respects the edge-order at v in the super-tree of T that defined the external linkage-edge.*

See also Fig. 1. We sometimes use the terms top-e-exposed, bottom-e-exposed, left-e-reachable and right-e-reachable if we want to clarify the placement of node v.

Occasionally, we modify drawing Γ by doing linear transformations; this preserves planarity and makes it easier to merge Γ. We list below the ones that we use and which properties they preserve; all of them preserve the height of the drawing. The simplest transformation is a a *horizontal flip* (mirroring Γ across a vertical line), which reverses the orders of edges at all nodes, but preserves whether Γ is e-exposed or e-reachable. We sometimes do a *rotation by* $180°$, which preserves edge-orders and whether Γ is e-exposed or e-reachable, but converts a top-e-exposed drawing into a bottom-e-exposed one and vice versa.

We also sometimes *shrink* Γ *horizontally*, i.e., map any point (x, y) to $(\varepsilon x, y)$ for some small $\varepsilon > 0$. This preserves edge-orders and whether Γ is e-exposed or e-reachable. Note that this may make x-coordinates non-integral, which is not a problem since (with the exception of the last step of Theorem 3) we do not require integral x-coordinates. Another useful operation is a *skew*, where any point (x, y) of Γ is mapped to point $(x + \alpha y, y)$ for some constant α. This preserves whether Γ is e-exposed, but does not necessarily preserve e-reachability, because the end v of e that is required to be leftmost or rightmost may cease to be so after a skew.

Finally we explain the *reversal trick*, which will help cutting down on the number of cases that we need to consider. For a given tree T, let T^{rev} be the tree obtained from T by reversing *all* edge-orders at all nodes. Note that a horizontal flip of a drawing of T gives a drawing of T^{rev}. During the constructions described below, we will sometimes need that three neighbors w, w', w'' of a node v occur in clockwise order around v. This may or may not be true in T, but it always holds in one of T and T^{rev}. Therefore, if need be, we draw T^{rev} rather than T, and flip the final drawing horizontally to obtain the desired drawing of T.

3 3pw(T)-Layer HVA-Drawings

In this section, we construct special types of drawings of trees that we call *HVA-drawings*: Every edge is either Horizontal, Vertical, or connects Adjacent layers. We will see (in the proof of Theorem 3) that such drawings can be modified without affecting height or planarity to achieve small width. We construct such drawings using induction on the pathwidth; the following is the hypothesis.

Lemma 1 *Let T be an ordered tree, and let e be an external linkage-edge with end $v \in T$. Then T has an e-exposed HVA-drawing on $3pw(T) + 1$ layers. Moreover, if T has at least two nodes and a main path that ends at v, then it has such a drawing on $3pw(T)$ layers.*

We first give an outline of the idea of the proof of Lemma 1. Exactly as in Suderman's construction for his Lemma 7 [12], we split the tree twice along paths before recursing, choosing the paths such that they cover a main path and reach the node v specified in Lemma 1. All remaining subtrees then have pathwidth at most $pw(T) - 1$, are hence drawn at most three units smaller recursively, and can be merged into a drawing of these two paths. The main difference between our construction and Suderman's is that we must respect the order, both within the merged subtrees and near the external linkage-edge. This requires a more complicated drawing for the path, and more argumentation for why we have enough space to merge.

We phrase our main step ("how to merge subtrees of a path") as a lemma in terms of an abstract height-bound k, so that we can use it for different values of k. For one of these merges, it is necessary to allow one component to be one unit taller than the others; the crux to obtain the $3pw(T)$-bound is to realize that one such component can always be accommodated. Let $\chi(x)$ be an indicator function that is 1 if x is true and 0 otherwise.

Lemma 2 *Let T be an ordered tree with an external linkage-edge $e_1 = (v_0, v_1)$ with $v_1 \in T$. Let $P = v_1, \ldots, v_l$ (the "*draw-path*") be a path in T, and let C_S (the "*special component*") be one component of $T \setminus P$. Fix an integer $k \geq 1$.*

Assume that any component C' of $T \setminus P$ has an e'-exposed HVA-drawing on k' layers, where $k' = k + \chi(C'{=}C_S)$ and e' is the linkage-edge of C'. Then T has an e_1-exposed HVA-drawing on $k + 2$ layers.

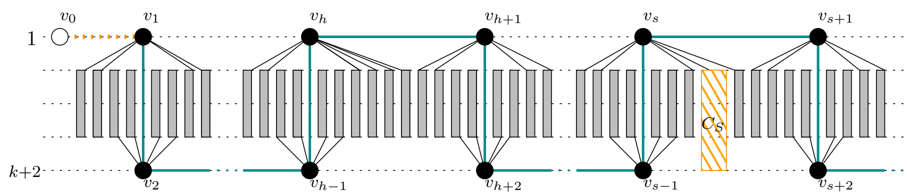

Figure 2: Merging at a draw-path P drawn as a battlement curve. In this and all following pictures the draw-path P is drawn turquoise (thick), and components that require special consideration are orange (patterned).

Proof: We start by drawing path P as a *battlement curve* on $k+2$ layers. Draw (v_1, v_2) as a vertical line segment connecting the top and bottom layer, and then alternate horizontal edges and vertical edges such that all vertices are on the top and bottom layer, the curve is x-monotone, and v_1, v_2 are leftmost. See also Fig. 2. We have a choice whether v_1 is in the top or bottom layer, and make this choice such that the anchor-node v_s of the special component C_S is drawn in the top layer. Either way, v_1 is in the top or bottom layer, and so e_1 is exposed as long as we merge components while respecting edge-orders.

We think of the battlement curve as being extended at both ends with nodes v_0 and v_{l+1}, v_{l+2}. This is done only to avoid having to describe special cases below when anchor-vertices are v_1 or v_l; the added edges are not included in the final drawing.

For any component C' of $T \setminus P$, the order of edges at its anchor-node v_j forces on which side of the battlement curve C' should be inserted. More precisely, C' should be placed below the battlement curve if and only if at the anchor-node v_j of C' the ccw order of edges around v_j contains $\langle (v_j, v_{j-1})$, the linkage-edge of C', $(v_j, v_{j+1})\rangle$ as a subsequence. Using the reversal-trick, if need be, we can hence ensure that the special-component C_S should be placed below the battlement curve. Also recall that we want an e_1-exposed drawing, and used edge $e_1 = (v_0, v_1)$ as extension of the battlement curve. Components with anchor-node v_1 will be placed below/above the battlement-curve so that the edge-order is correct relative to e_1. Since e_1 is drawn horizontally, we could therefore draw it outward from v instead and satisfy the edge-order condition of "e_1-exposed".

Now we explain how to merge the drawing Γ' of component C'. Let us first assume that Γ' has height at most k, as is the case for all components except C_S. Say Γ' must be added below the battlement curve (adding it above the battlement curve is symmetric). The anchor-node v_j of C' is incident to a region below the battlement curve, say this is the region below the horizontal edge (v_h, v_{h+1}) for some $h \in \{j-2, j-1, j, j+1\}$.

Consider Fig. 2. The linkage-edge e' of C' is exposed in Γ'. If $v_j = v_h$ or $v_j = v_{h+1}$, then rotate Γ', if needed, such that it is top-e'-exposed, so the linkage-node of C' is in the top layer of Γ'. Place Γ' in the k layers below the top one, then e' connects two adjacent layers and we obtain an HVA-drawing. (We assume for this and all later merging-steps that Γ' has been shrunk horizontally sufficiently so that this fits.) If $v_j = v_{h-1}$ or $v_j = v_{h+2}$, then rotate Γ', if needed, such that it is bottom-e'-exposed, so the linkage-node of C' is in the bottom layer of Γ'. Place Γ' in the k layers above the bottom one, then e' connects two adjacent layers and we obtain an HVA-drawing. If more than one component is adjacent to v_j, then place these components in the order dictated by the edge order at v_j. One easily verifies planarity, that we have an HVA-drawing, and that the drawing is order-preserving.

It remains to explain how to deal with the special component C_S whose drawing may use $k+1$ layers. We ensured that the anchor-node v_s of C_S is drawn in the top layer, and C_S should be placed below the battlement curve. We can hence insert it as in the first case above: the bottom layer of the region below (v_s, v_{s+1}) is free to be used for the drawing of C_S. See Fig. 2. □

Now we are ready to prove Lemma 1, i.e., to build an e-exposed HVA-drawing of a tree T.

Proof: We proceed by induction on $pw(T)$. In the base case, $pw(T) = 0$, so T is a single node that can be drawn on $1 = 3pw(T) + 1$ layers; the external linkage-edge is exposed automatically. For the induction step, $pw(T) \geq 1$. Let P_m be a main path of T. We have three cases.

In the first case, P_m begins at the node v that is the end of e in T. Apply Lemma 2 with draw-path $P := P_m$, external linkage-edge $e_1 := e$ and $k = 3pw(T) - 2$. (We have no need for a special component C_S in this case.) Any component C' of $T \setminus P$ has pathwidth at most $pw(T)-1$, and hence by induction can be drawn on $3(pw(T)-1)+1 = 3pw(T)-2 = k$ layers with its linkage-edge exposed. Therefore, T can be drawn on $k+2 = 3pw(T)$ layers as desired.

In the next case, P_m contains v, but does not end at v. Removing an edge (v, s) incident to v from P_m splits it into two paths X and S, named such that X ends at v and S ends at s. See Fig. 3(a). Apply Lemma 2 with draw-path $P := X$, external linkage-edge $e_1 := e$, $k = 3pw(T) - 1$, and special component C_S as the component of $T \setminus P$ that contains s. Any component $C' \neq C_S$ of $T \setminus P$ has pathwidth at most $pw(T) - 1$, and hence by induction can be drawn on $3(pw(T) - 1) + 1 = 3pw(T) - 2 < k$ layers with its linkage-edge exposed. (We can pad the drawing with an empty layer suitably to achieve that the height is exactly k.) The special component C_S may have pathwidth $pw(T)$, but can use S as its main path. Since S ends at the linkage-node s of C_S, therefore by induction C_S can be drawn on $3pw(T) = k + 1$ layers with its linkage-edge (s, v) exposed. So the lemma can be applied, and T can be drawn on $k + 2 = 3pw(T) + 1$ layers as desired.

In the final case, P_m does not contain v. Let C_v be the component of $T \setminus P_m$ that contains v, and let x be the anchor-node of C_v. Let R be the path in T from v to x. Removing an edge (s, x) incident to x from P_m splits it into two paths X and S, named such that X ends at x and S ends at s. See Fig. 3(b).

Apply Lemma 2 with draw-path $P := R \cup S$, external linkage-edge $e_1 := e$, $k = 3pw(T) - 1$, and special component C_S as the component of $T \setminus P$ that

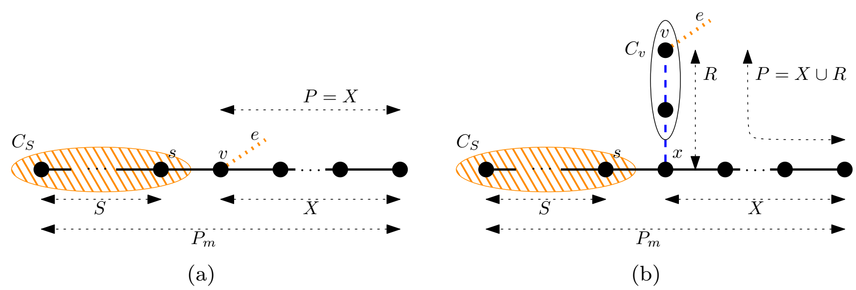

Figure 3: Finding the draw-path if node v is not an end of the main path P_m (black, thick). (a) v is in P_m, and we use X as the draw-path. (b) v is in a component C_v of $T \setminus P_m$, and the draw-path consists of X and path R (blue, thick dashed) that leads from P_m to v. To avoid cluttering we omit subtrees.

contains s. Any component $C' \neq C_S$ of $T \setminus P$ has pathwidth at most $pw(T)-1$, because it is either a component of $T \setminus P_m$ (hence has smaller pathwidth by definition of a main path) or a subtree of C_v (hence $pw(C') \leq pw(C_v) < pw(T)$ since C_v is a component of $T \setminus P_m$). Therefore any component $C' \neq C_S$ of $T \setminus P$ can by induction be drawn on $3(pw(T)-1)+1 = 3pw(T)-2 < k$ layers with its linkage-edge exposed. The special component C_S may have pathwidth $pw(T)$, but can use S as its main path. Since S ends at the linkage-node s of C_S, therefore by induction C_S can be drawn on $3pw(T) = k+1$ layers with its linkage-edge (s, x) exposed. So the lemma can be applied, and T can be drawn on $k+2 = 3pw(T)+1$ layers as desired. □

Theorem 3 *Any ordered tree T has an order-preserving planar straight-line HVA-drawing with height at most* $\max\{1, 3pw(T)\}$ *and width at most* $|V(T)|$.

Proof: If T has pathwidth 0 then it is a singleton node and the bound is obvious, so assume $pw(T) \geq 1$. Fix a main path of T, and insert a dummy external-linkage-edge at its end. The resulting tree has the same pathwidth. Now apply Lemma 1 to obtain an order-preserving planar straight-line HVA-drawing Γ of the required height.

It remains to argue the width, for which we need a small detour. A *rectangle-of-influence drawing* (see e.g. [9]) is a straight-line drawing in the plane such that for any edge (u, w), the minimum axis-aligned rectangle $R(u, w)$ containing u and w is either the line segment $\overline{uw}$, or its interior contains no other nodes of the drawing. It is well-known that in a rectangle-of-influence drawing we can change the x-coordinates without affecting planarity, as long as relative orders are preserved.

Observe that any HVA-drawing is a rectangle-of-influence drawing, because any edge (u, w) is either horizontal or vertical (then $R(u, w)$ is the line segment $\overline{uw}$) or (u, w) connects adjacent layers (then the interior of $R(u, w)$ consists of points that are between layers and hence contains no other nodes).

So modify the obtained drawing Γ into drawing Γ' as follows. Enumerate all x-coordinates of nodes as $x_1, \ldots, x_W$ with $x_1 < x_2 < \cdots < x_W$, and then assign $x(w) := i$ if node w had x-coordinate x_i. Keep y-coordinates unchanged. Clearly the relative orders of coordinates have been preserved, so Γ' is planar since Γ was planar. Also all edges are again horizontal, vertical or connect adjacent layers, and the height is unchanged. The width is $W \leq |V(T)|$, which gives the result. □

4 (2pw(T) + 1)-Layer Drawings of Ordered Trees

We now improve the number of layers, at the cost of not having an upper bound on the width. Our construction is very similar to the one of Suderman for his Lemma 19 [12], except that we must be more careful when merging subtrees so that the order is preserved. There are two key differences to the construction from the previous section:

1. We split three times along paths, and achieve that the resulting subtrees have pathwidth at most $pw(T) - 2$.

2. In the top-level split, we do *not* require that the draw-path P begins at the node v at which the external linkage-edge e attaches.

The second change makes the top-level split much more efficient, but means that when recursing in the sub-tree C_v that contains v, we now must consider *two* linkage-edges: the external linkage-edge e and the linkage-edge from C_v to P. (We make one exposed and the other reachable.) This will complicate the induction hypothesis (which is expressed in the following lemma) significantly.

Lemma 3 *Let T be an ordered tree and e be an external linkage-edge.*
(a) T has a drawing on $2pw(T) + 1$ layers that is e-exposed.
(b) Let e' be a second external linkage-edge that has no common end with e. Then T has a drawing on $2pw(T) + 2$ layers that is e-exposed and e'-reachable.

This lemma will be proved by induction on the pathwidth. For the induction step, we need to merge components into a drawing of a path. Since this will be done repeatedly with different paths, we phrase this merging-step as a lemma using as height-bound an abstract constant k. This lemma is quite similar to Lemma 2, but has more complicated conditions that are illustrated in Fig. 4.

Lemma 4 *Let T be an ordered tree with an external linkage-edge $e_1 = (v_1, v_0)$ with $v_1 \in T$. Let the draw-path $P = v_1, \ldots, v_l$ be a path of T starting at v_1. Let $e_v = (v, u)$ be some other external linkage-edge with $v \in T$. Fix some $k \geq 1$.*

Assume that every component C' of $T \setminus P$ that is not C_v (defined below) can be drawn on k layers with its linkage-edge exposed. Assume further that one of the following conditions holds (see also Fig. 4):

I. $v \in P$ and $v \neq v_1$. [No component C_v is needed in this case.]

II. $v \notin P$*, v is not adjacent to a vertex of P, and the component* C_v *of* $T \setminus P$ *that contains v has a drawing on* $k+1$ *layers that is* e_v*-exposed and* e_C*-reachable, where* e_C *is the linkage-edge of* C_v*.*

III. $v \notin P$*, v is adjacent to a vertex of P, and every component* C'' *of* $C_v \setminus \{v\}$ *(where* C_v *as before is the component of* $T \setminus P$ *containing v) has a drawing on k layers such that the edge connecting* C'' *to v is exposed.*

Then T has a drawing on $k+2$ *layers that is* e_v*-exposed and* e_1*-reachable.*

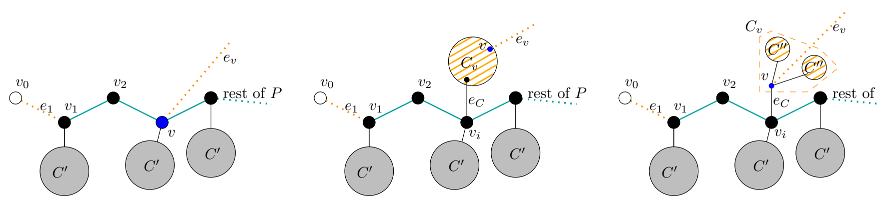

Figure 4: Notations for the three conditions for Lemma 4. The component C_v of $T \setminus P$ that contains v is orange (rising pattern).

Proof: The first step is to draw P on $k+2$ layers as a zig-zag-curve[1] between the top and the bottom layer, with v_1 leftmost. With this e_1 ends at the unique leftmost node and hence is reachable as long as we merge components suitably. For ease of description, we think of the zig-zag-line as extended further left and right with vertices v_0 and v_{l+1}; these will not be in the final drawing.

We have the choice of placing v_1 in the top or in the bottom layer, and make this choice as follows. Define v_i to be v if $v \in P$ and define v_i to be the anchor-node of C_v if $v \notin P$. Choose the placement of v_1 such that v_i is in the top layer.

The following details the *standard-method* of merging a component C' anchored at $v_j \in P$. See also Fig. 5. Assume that v_j is in the top layer; the other case is symmetric. Assume that the linkage-edge of C' was top-exposed in the drawing Γ' of C'; else rotate Γ' by 180° to make it so. Scan the edge-order around v_j to find the two incident path edges (v_j, v_{j+1}) and (v_j, v_{j-1}). If the linkage-edge of C' appears clockwise between these two, then place Γ' below edge (v_j, v_{j+1}), else place it above (v_j, v_{j+1}). In both cases, we do not use the top layer for Γ', and can hence connect to the linkage-node of C' while preserving planarity and edge-orders since the linkage-edge was top-exposed. If multiple components are anchored at v_j, then we all place them in this region, in the order as dictated by the edge-order at v_j.

Now we show how to make the drawing e_v-exposed while preserving e_1-reachability. We distinguish cases depending on which condition applies.

[1]Using a zig-zag-curve allows more flexibility in placing components, but means that we will not have an HVA-drawing.

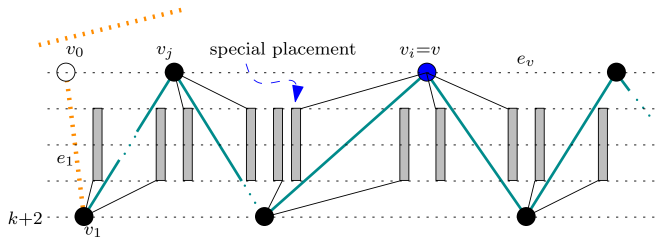

Figure 5: Adding components to a zig-zag path for Lemma 4. The component marked with an arrow needs to be placed to the left of v_i so that the edge-order at e_v is respected.

Condition I. We know that $v = v_i$ for some $i > 1$ and v_i is in the top layer. After applying the reversal-trick, if needed, we may assume that the clockwise order at v_i in the super-tree contains $\langle (v, v_{i-1}), e_v, (v, v_{i+1})\rangle$ as a subsequence. Therefore, drawing e_v upward from v_i makes it top-exposed as long as we merge components suitably.

Merge all components not anchored at v_i with the standard-method. For a component C' anchored at $v_i = v$, the placement must be such that the order including edge e_v is also respected. This is done as follows (see also Fig. 5): Determine where the linkage-edge of C' falls in the clockwise order around v. If it is between e_v and (v_i, v_{i+1}), or between (v_i, v_{i+1}) and (v_i, v_{i-1}), then place C' with the standard-method. But if it is between (v_i, v_{i-1}) and e_v, then place the drawing of C' in the region above edge (v_i, v_{i-1}) (and to the right of any components anchored at v_{i-1} that may also have been placed there). By $i > 1$, this does not place anything to the left of v_1, and so v_1 continues to be e_1-reachable.

Condition II or III. Recall that the anchor-node v_i of C_v is drawn in the top layer. Apply the reversal-trick, if needed, to ensure that e_C appears between (v_i, v_{i+1}) and (v_i, v_{i-1}) in clockwise order around v_i.

We merge the drawings of subtrees anchored at v_i as follows. If Condition II holds, then assume (after possible rotation) that the drawing Γ_v of C_v is bottom-e_v-exposed. Insert Γ_v in the region below (v_i, v_{i+1}). This is possible (after skewing Γ_v as needed) without crossing, since the end of e_C in C_v is the unique leftmost or rightmost node of Γ_v. See Fig. 6. If Condition III holds, then place v on the bottom layer, in the region below edge (v_i, v_{i+1}), and connect it to v_i. This makes e_v bottom-exposed, as long as we are careful when placing components of $C_v \setminus \{v\}$. For each such component C'', we have a drawing Γ'' on k layers where the linkage-edge from C'' to v is exposed. Rotate Γ'', if needed, to make this edge bottom-exposed, and then place Γ'' in the k layers above v, either left or right of edge (v_i, v), as dictated by the edge-order around v. See Fig. 6.

We merge all other components C' of $T \setminus P$ with the standard-method. This includes any other components that may be anchored at v_i; for those we place them so that they are left/right of C_v as dictated by the edge-order, but still remain in the region above or below (v_i, v_{i+1}) so that v_1 is the unique leftmost node. □

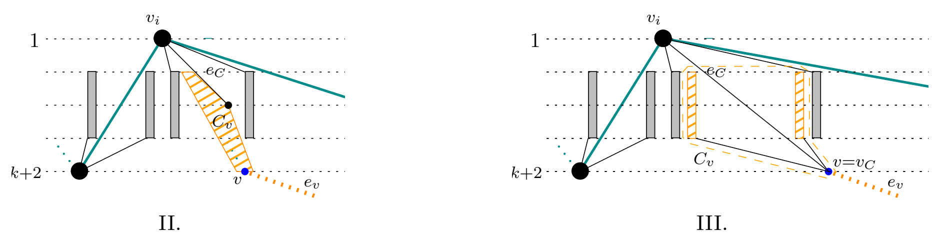

Figure 6: Merging component C_v (orange, rising pattern), depending on which condition holds.

We are now ready to give the proof of Lemma 3.

Proof: Recall that we are given a tree T with an external linkage-edge e that ends at $v \in T$, and possibly a second external-linkage e' that ends at $v' \in T$ with $v \neq v'$. We want to find drawings that are e-exposed and (perhaps) e'-reachable.

We proceed by induction on $pw(T)$. In the base case, $pw(T) = 0$, so T is a single node and drawing T on a single layer satisfies Claim (a). Claim (b) is vacuously true since any two external linkage-edges would have the (unique) node of T in common.

For the induction step let $pw(T) \geq 1$ and let $P_m = v_1, \dots, v_l$ be a main path of T. Any component C' of $T \setminus P_m$ has pathwidth at most $pw(T)-1$ and hence can be drawn using induction. For some components we will create different drawings later to accommodate external linkage-edges.

Induction step for Claim (a): In this case no edge e' has been specified; we artificially insert one as follows. Since we may assume that P_m has at least one edge, at least one end v' of P_m is not node v; insert a dummy external-linkage edge e' here and note that it shares no end with e as required. The goal is to apply Lemma 4 using path P_m, $k = 2pw(T) - 1$, $e_v := e$, and $e_1 = e'$. For this, first observe that any component C' of $T \setminus P_m$ has smaller pathwidth than T, hence can be drawn by induction on at most $2pw(T) - 1 \leq k$ layers with its linkage-edge exposed. It remains to argue that one of the conditions holds.

If $v \in P_m$ then Condition I holds (we know $v \neq v'$ since e and e_v have no end in common). If $v \notin P_m$, then let C_v be the component of $T \setminus P_m$ that contains v and let e_C and v_C be its linkage-edge and linkage-node. We know that C_v has pathwidth at most $pw(T) - 1$. If $v_C \neq v$, then apply induction (Claim (b)) to get a drawing of C_v on $2pw(T) = k + 1$ layers that is e_v-exposed and e_C-reachable. So Condition II applies. Otherwise ($v_C = v$) any component C'' of $C_v \setminus \{v\}$ has pathwidth at most $pw(C_v) \leq pw(T) - 1$, and by induction hence has a drawing on $2pw(T) - 1 = k$ layers such that the edge from C'' to v is exposed. So Condition III holds.

We can hence apply Lemma 4 and get an e-exposed drawing of height $k+2 = 2pw(T) + 1$ as desired.

Induction step for Claim (b): Recall that $P_m = v_1, \ldots, v_l$ is a main path of T and v and v' are the ends of edges that should be exposed and reachable, respectively. We now split T along a draw-path derived from P and v' and v; this draw-path is *different* from what we used in Section 3 (and in particular, begins at v' rather than v).[2]

Fig. 7 illustrates the following definitions. If $v' \in P_m$, then set $s := v'$ and let R be an empty path. Otherwise, let s be the anchor-node of the component of $T \setminus P_m$ that contains v', and let R be the path from v' to s in T. If $v \in P_m$, then set $y := v$. Otherwise, let y be the anchor-node of the component of $T \setminus P_m$ that contains v. Note that we well may have $s = y$.

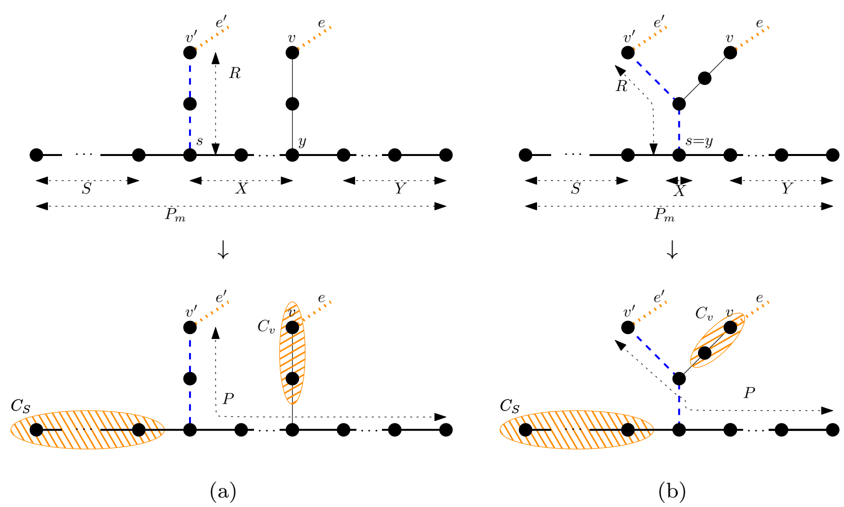

Figure 7: Splitting the tree to obtain path P. The main path P_m is thick black, the path R from P_m to v' is blue (dashed). To avoid cluttering we omit subtrees. (a) The subtrees of $T \setminus P_m$ containing v' and v are anchored at different nodes of P_m. (b) The subtrees of $T \setminus P_m$ containing v' and v are anchored at the same node of P_m.

Let X be the sub-path of P_m between s and y (inclusive); it may be a single vertex $s{=}y$. Let S and Y be the (possibly empty) two components of $P_m \setminus X$, named such that S is anchored at s and Y is anchored at y. Define the draw-path P to be $(P_m \setminus S) \cup R$. Put differently, P is the path in T that connects v' to one of the ends of P_m, where the end of P_m is chosen such that the component

[2] This choice of paths is the same as in Suderman, Lemma 23, though we combine the drawings of the subtrees quite differently to maintain edge orders.

C_v of $T \setminus P$ that contains v does *not* contain vertices of P_m, and therefore is guaranteed to have pathwidth at most $pw(T) - 1$.

The goal is to apply Lemma 4 using P as the draw-path. However, if S (the "rest" of P_m) is non-empty, then this is not straightforward, because the component C_S of $T \setminus P$ that contains S has pathwidth $pw(T)$ and so cannot necessarily be drawn small enough.

Case 1: $S = \emptyset$. Use Lemma 4 with P as the draw-path, $e_1 := e'$, $e_v := e$, and $k = 2pw(T)$.[3] We must argue that this is feasible. First, any component C' of $T \setminus P$ has pathwidth at most $pw(T) - 1$ since S is empty and so P covers the entire main path P_m. So C' has by induction (Claim (a)) a drawing on $2pw(T) - 1 \leq k$ layers with its linkage-edge exposed.

If $v \in P$ then Condition I holds (we know $v \neq v'$ since e and e_v have no end in common). If $v \notin P$ then let C_v be the component of $T \setminus P$ that contains v, and let e_C and v_C be its linkage-edge and linkage-node. If $v \neq v_C$, then use induction (Claim (b)) to obtain a drawing of C_v on $2pw(C_v) + 2 \leq 2pw(T) \leq k + 1$ layers such that e_v is exposed and e_C is reachable. So Condition II holds. Finally if $v = v_C$, then any component C'' of $C_v \setminus \{v\}$ has pathwidth at most $pw(C_v) \leq pw(T) - 1$ and by induction (Claim (a)) C'' can be drawn on $2pw(T) - 1 \leq k$ layers such that edge from C'' to v is exposed. So Condition III holds. Hence regardless of the location of v we obtain a drawing of T on $k + 2 = 2pw(T) + 2$ layers with e' reachable and e exposed.

Case 2: S is non-empty, and component C_S "belongs to the taller side" (defined below). Construct a drawing of $T \setminus C_S$ as in Case 1. We say that C_S *belongs to the taller side* if the anchor-node v_S of C_S is in the top [bottom] layer and the clockwise [counter-clockwise] order of edges around v_s contains $\langle (v_s, v_{s+1})$, the linkage-edge of C_S, $(v_s, v_{s-1}) \rangle$ as a subsequence. Put differently, belonging to the taller side means that the drawing of C_S needs to be put into a region that has $2pw(T) + 1$ levels that can be used for inserting drawings. Construct a drawing Γ_S of C_S with its linkage-edge exposed on $2pw(T) + 1$ layers using induction (Claim (a)). We can insert Γ_S with the standard-method for merging components since C_S belongs to the taller side. See Fig. 8.

Case 3: S is non-empty and C_S does not belong to the taller side. In this case we need a special construction to accommodate C_S.[4] Let v_s be the anchor-node of S. Let T^- be the tree that results from removing from T the component C_S, as well as all components of $T \setminus P$ that are anchored at v_s. We first construct a drawing of T^- on $2pw(T) + 2$ layers as in Case 1. Assume that v_s is in the top level; the other case is symmetric. We know that C_S does not belong to the taller side, so it should normally be placed above edge (v_s, v_{s+1}) to preserve edge-orders. (In the special case that $v_s = v$, it may have to be placed above

[3]For Case 1, $k = 2pw(T) - 1$ would have been enough, but later cases build on top of this and then require $k = 2pw(T)$.

[4]One might be tempted to appeal to the reversal-trick here to ensure that C_S belongs to the taller side. However, the reversal-trick was already used in Lemma 4 to ensure that C_v belongs to the taller side of v_i, and this is used here as a subroutine. We cannot apply the reversal-trick for two different subtrees of one main path.

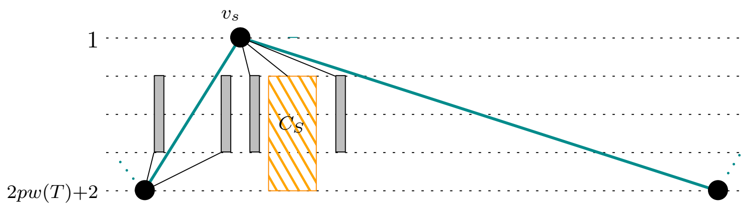

Figure 8: Inserting component C_S (orange, falling pattern) if it belongs to the taller side.

edge (v_s, v_{s-1}) instead to preserve edge-orders for e_v; this can be handled in a symmetric fashion.)

Observe that S is a main path of C_S. We draw S as a zig-zag-curve alternating between layer 1 and layer $2pw(T)+1$, going rightwards from v_s. See Fig. 9 Any component C'' of $C_S \setminus S$ has pathwidth at most $pw(T)-1$, and can hence be drawn inductively (Claim (a)) on $2pw(T)-1$ layers with its linkage-edge exposed. We can hence merge these components in the regions around S, exactly as in Lemma 4. Finally, we must merge a component C' anchored at v_s. If this component came (in the clockwise order around v_s) before the linkage-edge of C_S, then path S now blocks the connection to where we would normally place C'. (All other components at v_s can be merged with the standard-construction.) We know that C' can be drawn with $2pw(T)-1$ layers. Since the linkage-node of C_S is placed on layer $2pw(T)+1$, we can place C' in the $2pw(T)-1$ layers below the top-row and above the linkage-edge and connect it to v_s without violating planarity and respecting edge-orders.

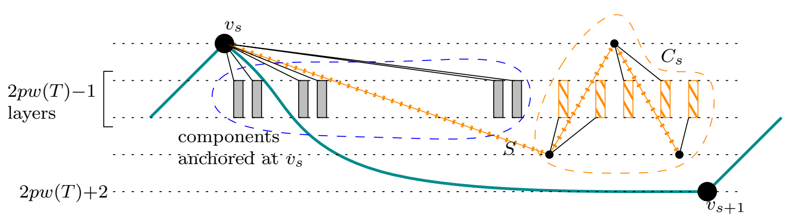

Figure 9: The special construction for component C_S if it does not belong to the taller side. We draw (v_s, v_{s+1}) slightly curved to avoid having to scale too much.

This special construction for C_S does *not* interfere with the (potentially special) construction for component C_v (presuming $v \notin P$), because we had ensured (by using the reversal-trick, if needed) that C_v belongs to the taller side. So either C_v is in a different region altogether, or C_v is anchored at v_{s+1}, and we can easily keep these drawings separate. This finishes the proof of

Lemma 3. □

By applying Lemma 3(a) with an arbitrary dummy-edge as external linkage-edge, we hence obtain:

Theorem 4 *Any tree T has a planar straight-line order-preserving drawing on $2pw(T)+1$ layers.*

Note that we make no claims on the width of the drawing. In fact, in order to fit drawings of components within the regions underneath zig-zag-lines, we may have to scale these components horizontally (or equivalently, widen the zig-zags significantly).

5 2pw(T) + 1 Layers is Tight

We can show that the bound in Theorem 4 is tight. Define an ordered tree T_i recursively as follows. T_0 consists of a single node. T_i for $i > 0$ consists of a path v_1, v_2, v_3 and 12 copies of T_{i-1}, three attached at each of v_1, v_3 and three attached on each side of the path at v_2. (It does not matter which node of T_{i-1} is used as linkage-node for these attachments.) See also Fig. 10.

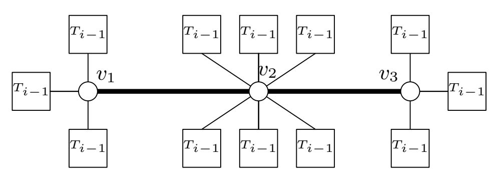

Figure 10: Tree T_i has pathwidth i but requires $2i+1$ layers in an order-preserving planar drawing.

By using v_1, v_2, v_3 as main path, one sees that $pw(T_i) \leq i$. We can now show the lower bound on the height, which holds even if the drawing is not straight-line.

Theorem 5 *Any planar order-preserving drawing of T_i has at least $2i+1 \geq 2pw(T_i)+1$ layers.*

Proof: We prove this by induction on i; the case $i = 0$ is trivial since the single-node tree T_0 requires 1 layer. So assume that $i > 0$ and we already know that T_{i-1} requires at least $2i-1$ layers by induction. We need a helper-lemma.

Lemma 5 *Let H_i be the tree that consists of a single node v with three copies of T_{i-1} attached. Then H_i requires at least $2i$ layers.*

Proof: Assume to the contrary that H_i could be drawn on $2i-1$ layers. For each copy of T_{i-1}, we require $2i-1$ layers. Hence each copy of T_{i-1} gives rise to a *blocking path* that connects the topmost and bottommost layer and stays within that copy of T_{i-1}. Add a node v' above the drawing connected to the three top ends of the three blocking paths, and a node v'' below the drawing connected to the three bottom ends of the three blocking paths. See also Fig. 11(a). Also observe that v is connected (via a path within that copy of T_{i-1}) to each of the three blocking paths. Therefore the three blocking paths, together with $\{v, v', v''\}$, give a planar drawing of a subdivision of $K_{3,3}$, an impossibility. □

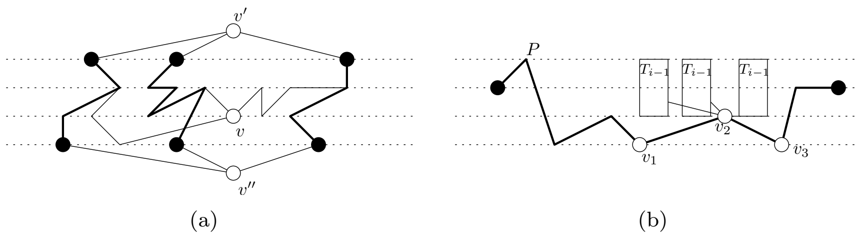

Figure 11: (a) We can construct a planar drawing of $K_{3,3}$. (b) If v_2 is not in the top row, then the path P forces a copy of H_i to be drawn within $2i-1$ layers.

Now we give the induction step of the proof of Theorem 5. Since T_i contains H_i, by Lemma 5 it requires at least $2i$ layers. Assume for contradiction that we have a drawing Γ of T_i on exactly $2i$ layers. Let P be a path that connects a leftmost node in Γ to a rightmost node in Γ (breaking ties arbitrarily). Any subtree that is node-disjoint from P must not intersect it by planarity and hence must be drawn either within the bottommost $2i-1$ layers or within the topmost $2i-1$ layer. See also Fig. 11(b).

Observe that P must contain path v_1, v_2, v_3, for otherwise we have a copy of H_i at one of v_1, v_3 that is node-disjoint from P and would be drawn in $2i-1$ layers, which is impossible. Now consider the layer that v_2 is on. Since we have $2i \geq 2$ layers, one of the top and bottom layer does not contain v_2, say v_2 is not on the bottom layer. Since path P uses v_1, v_2, v_3, and since the drawing is order-preserving, there must be three copies of T_{i-1} that are attached at v_2 and above path P, hence in the top $2i-1$ layers. Vertex v_2 together with these three copies forms an H_i, and since it is vertex-disjoint from P (except at v_2, but v_2 is not in the bottom layer either), it is drawn in $2i-1$ layers. This contradicts Lemma 5, so no drawing Γ of T_i on $2i$ layers can exist. □

6 Remarks

In this paper, we studied planar straight-line order-preserving drawings of trees that use few layers. Inspired by techniques of Suderman [12], we gave two constructions. The first one is a 3-approximation for the height and the width is bounded by n. The second is an asymptotic 2-approximation for the height, with no bound on the width. We also showed that '2' is tight if one uses the pathwidth for lower-bounding the height.

Our constructions are algorithmic, and the bottleneck for its run-time is the extraction of main paths. It is known how to compute the pathwidth of a tree in linear time [11]. It is not hard to see that for a rooted tree, this bottom-up dynamic programming algorithm to compute the pathwidth stores sufficient information that we can find a main path P_m, and the path R from the root to the nearest node on P_m, in time $O(|P_m \cup R|)$ time. Our algorithm can be viewed as traversing a rooted tree top-down (the root is vertex v in Lemma 1 and vertex v' in Lemma 3). In each recursion we exactly need to find a main path P_m and the path R that leads to it from the root; this hence takes time $O(|P_m \cup R|)$. Also, $P_m \cup R = P \cup S$, where P is the draw-path, and S is the draw-path used when recursing in the special component C_S of $T \setminus P$. Hence we next need to find main paths only in subtrees of $T \setminus (P_m \cup R)$. Since all other steps of the recursion can also be done in $O(|P_m \cup R|)$ time, the overall run-time is linear, presuming that we can handle arbitrarily small coordinates in constant time.

As for open problems, all our constructions (and all the ones by Suderman) rely on path decompositions, and hence yield only approximation algorithms to the height of tree-drawings. The algorithm for optimum-height (unordered) tree-drawings [10] uses an entirely different, direct approach. Is there a poly-time algorithm that finds optimum-height ordered tree-drawings?

References

[1] H. A. Akitaya, M. Löffler, and I. Parada. How to fit a tree in a box. In T. Biedl and A. Kerren, editors, *Graph Drawing and Network Visualization - 26th International Symposium, GD 2018*, volume 11282 of *LNCS*, pages 361–367. Springer, 2018. doi:10.1007/978-3-030-04414-5_26.

[2] T. Biedl. Ideal drawings of rooted trees with approximately optimal width. *Journal of Graph Algorithms and Applications*, 21(4):631–648, 2017. doi:10.7155/jgaa.00432.

[3] T. Biedl and D. Mondal. On upward drawings of trees on a given grid. In F. Frati and K. Ma, editors, *Graph Drawing and Network Visualization - 25th International Symposium, GD 2017*, volume 10692 of *LNCS*, pages 318–325. Springer, 2017. doi:10.1007/978-3-319-73915-1_25.

[4] T. M. Chan. Tree drawings revisited. In B. Speckmann and C. D. Tóth, editors, *34th International Symposium on Computational Geometry, SoCG 2018*, volume 99 of *LIPIcs*, pages 23:1–23:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018. doi:10.4230/LIPIcs.SoCG.2018.23.

[5] P. Crescenzi, G. Di Battista, and A. Piperno. A note on optimal area algorithms for upward drawings of binary trees. *Comput. Geom.*, 2:187–200, 1992. doi:10.1016/0925-7721(92)90021-J.

[6] G. Di Battista and F. Frati. A survey on small-area planar graph drawing. *CoRR*, abs/1410.1006, 2014. URL: http://arxiv.org/abs/1410.1006.

[7] S. Felsner, G. Liotta, and S. Wismath. Straight-line drawings on restricted integer grids in two and three dimensions. *Journal of Graph Algorithms and Applications*, 7(4):363–398, 2003. doi:10.7155/jgaa.00075.

[8] A. Garg and A. Rusu. Area-efficient order-preserving planar straight-line drawings of ordered trees. *Int. J. Comput. Geometry Appl.*, 13(6):487–505, 2003. doi:10.1142/S021819590300130X.

[9] G. Liotta, A. Lubiw, H. Meijer, and S. Whitesides. The rectangle of influence drawability problem. *Comput. Geom.*, 10(1):1–22, 1998. doi:10.1016/S0925-7721(97)00018-7.

[10] D. Mondal, M. J. Alam, and M. S. Rahman. Minimum-layer drawings of trees - (extended abstract). In N. Katoh and A. Kumar, editors, *Algorithms and Computation - 5th International Workshop, WALCOM 2011*, volume 6552 of *LNCS*, pages 221–232. Springer, 2011. doi:10.1007/978-3-642-19094-0_23.

[11] P. Scheffler. *A Linear Algorithm for the Pathwidth of Trees*, pages 613–620. Physica-Verlag HD, 1990. doi:10.1007/978-3-642-46908-4_70.

[12] M. Suderman. Pathwidth and layered drawings of trees. *Int. J. Comput. Geometry Appl.*, 14(3):203–225, 2004. doi:10.1142/S0218195904001433.