SIMPLE COMPACT MONOTONE TREE DRAWINGS *

ANARGYROS OIKONOMOU † AND ANTONIOS SYMVONIS ‡

Abstract. A monotone drawing of a graph G is a straight-line drawing of G such that every pair of vertices is connected by a path that is monotone with respect to some direction.

Trees, as a special class of graphs, have been the focus of several papers and, recently, He and He [6] showed how to produce a monotone drawing of an arbitrary n-vertex tree that is contained in a $12n \times 12n$ grid.

All monotone tree drawing algorithms that have appeared in the literature consider rooted ordered trees and they draw them so that (i) the root of the tree is drawn at the origin of the drawing, (ii) the drawing is confined in the first quadrant, and (iii) the ordering/embedding of the tree is respected. In this paper, we provide a simple algorithm that has the exact same characteristics and, given an n-vertex rooted tree T, it outputs a monotone drawing of T that fits on a $n \times n$ grid.

For unrooted ordered trees, we present an algorithms that produces monotone drawings that respect the ordering and fit in an $(n+1) \times (\frac{n}{2}+1)$ grid, while, for unrooted non-ordered trees we produce monotone drawings of good aspect ratio which fit on a grid of size at most $\lfloor \frac{3}{4}(n+2) \rfloor \times \lfloor \frac{3}{4}(n+2) \rfloor$.

Key words. Monotone tree drawings, graph drawing, grid drawing, area of drawing, algorithm.

1. Introduction. A *straight-line drawing* Γ of a graph G is a mapping of each vertex to a distinct point on the plane and of each edge to a straight-line segment between the vertices. A path $P = \{p_0, p_1, \ldots, p_n\}$ is *monotone* if there exists a line l such that the projections of the vertices of P on l appear on l in the same order as on P. A straight-line drawing Γ of a graph G is *monotone*, if a *monotone* path connects every pair of vertices.

Monotone graph drawing has gained the recent attention of researchers and several interesting results have appeared. Given a planar fixed embedding of a planar graph G, a planar monotone drawing of G can be constructed, but at the cost of some bends on some edges (thus no longer a straight-line drawing) [2]. In the variable embedding setting, there exists a planar monotone drawing of any planar graph without any bends [8].

One way to find a monotone drawing of a graph is to simply find a monotone drawing of one of its spanning trees. For that reason, the problem of finding monotone drawings of trees has been the subject of several recent papers, starting from the work by Angelini et al. [1] which introduced monotone graph drawings. Angelini et al. [1] provided two algorithms that used ideas from number theory and more specifically Stern-Brocot trees [11, 3], [4, Sect. 4.5]. The first algorithm used a grid of size $O(n^{1.6}) \times O(n^{1.6})$ (BFS-based algorithm) while the second one used a grid of size $O(n) \times O(n^2)$ (DFS-based algorithm). Later, Kindermann et al. [9] provided an algorithm based on Farey sequence (see [4, Sect. 4.5]) that used a grid of size $O(n^{1.5}) \times O(n^{1.5})$. He and He [7] gave an algorithm based on Farey sequence and reduced the required grid size to $O(n^{1.205}) \times O(n^{1.205})$, which was the first result that used less than $O(n^3)$ area. Recently, He and He [5] firstly reduced the grid size for a monotone tree drawing to $O(n \log(n)) \times O(n \log(n))$ and, in a sequel paper, to

*Funding: The work of Prof. Symvonis was supported by the iRead H2020 research grant (No. 731724).

†School of Electrical & Computer Engineering, National Technical University of Athens, Greece (ar.economou@outlook.com)

‡School of Applied Mathematical & Physical Sciences, National Technical University of Athens, Greece (symvonis@math.ntua.gr)

$O(n) \times O(n)$ [6]. Their monotone tree drawing uses a grid of size at most $12n \times 12n$ which turns out to be asymptotically optimal as there exist trees which require at least $\frac{n}{9} \times \frac{n}{9}$ area [6].

Our Contribution:
All monotone tree drawing algorithms that have appeared in the literature consider rooted ordered trees and they draw them so that (i) the root of the tree is drawn at the origin of the drawing, (ii) the drawing is confined in the first quadrant, and (iii) the embedding of the tree is respected. In this paper, we provide a simple algorithm that has the exact same characteristics and, given an n-vertex rooted tree T, it outputs a monotone drawing of T that fits on a $n \times n$ grid. Despite its simplicity, our algorithm improves the $12n \times 12n$ result of He and He [6].
By relaxing the drawing restrictions we can achieve smaller drawing area. More specifically, by carefully selecting a new root for the tree, which we draw it at the origin, we can produce a "two-quadrants" drawing that fits in an $(n+1) \times (\frac{n}{2}+1)$ grid. We note that the produced drawing respects the given embedding of the tree. By further relaxing this requirement, i.e., by allowing to change the order of the neighbors of a tree vertex around it, we can achieve a drawing of better aspect ratio and smaller area (compared to our $n \times n$ algorithm). More specifically, we describe a "four-quandrants" algorithm that draws an n-vertex tree on a grid of size at most $\lfloor \frac{3}{4}(n+2) \rfloor \times \lfloor \frac{3}{4}(n+2) \rfloor$.
The paper is organized as follows: Section 2 provides definitions and preliminary results. In Sections 3, 4 and 5 we present our one-, two- and four-quadrants algorithms, respectively. We conclude in Section 6. A preliminary version of this paper which included the one-quadrant algorithm for monotone tree drawings was presented in [10].

2. Definitions and Preliminaries. Let Γ be a drawing of a graph G and (u, v) be an edge from vertex u to vertex v in G. The slope of edge (u, v), denoted by $slope(u, v)$, is the angle spanned by a counter-clockwise rotation that brings a horizontal half-line starting at u and directed towards increasing x-coordinates to coincide with the half-line starting at u and passing through v. We consider slopes that are equivalent modulo 2π as the same slope. Observe that $slope(u, v) = slope(v, u) - \pi$. We only deal with planar monotone drawings of trees, as it was proved by Angelini et al. that every monotone drawing of tree is planar [1].
Let T be a tree rooted at a vertex r. Denote by T_v the subtree of T rooted at a vertex v. By $|T_v|$ we denote the number of vertices of T_v. Let v be a child of u. By T_v^u we denote the tree that consists of edge (u, v) and T_v. In the rest of the paper, we assume that all tree edges are directed away from the root. A rooted tree is said to be *ordered* if there is an order imposed on the children of each vertex. A drawing is said to *respect the ordering* of the tree (or the embedding) if the children of a vertex are drawn around it in the specified order.
When producing a grid drawing, it is common to refer to the *side-length* of the required grid and to its *dimensions*. We emphasize that we measure length (width/height) in units of distance, but when we denote the dimensions of a grid we use the number of grid points in each dimension. So, a grid of width w and height h fits in a $(w+1) \times (h+1)$ grid.

2.1. Slope-disjoint Tree Drawings. Angelini et al. [1] defined the notion of slope-disjoint tree drawings. Let Γ be a drawing of a rooted tree T. Γ is called a *slope-disjoint* drawing of T if the following properties are satisfied:

1. For every vertex $u \in T$, there exist two angles $a_1(u)$ and $a_2(u)$, with $0 <$

$a_1(u) < a_2(u) < \pi$ such that for every edge e that is either in T_u or that enters u from its parent, it holds that $a_1(u) < slope(e) < a_2(u)$.
2. For every two vertices $u, v \in T$ such that v is a child of u, it holds that $a_1(u) < a_1(v) < a_2(v) < a_2(u)$.
3. For every two vertices v_1, v_2 having the same parent, it holds that either $a_1(v_1) < a_2(v_1) < a_1(v_2) < a_2(v_2)$ or $a_1(v_2) < a_2(v_2) < a_1(v_1) < a_2(v_1)$.

The idea behind the definition of slope-disjoint tree drawings is that all edges in the subtree T_u as well as the edge entering u from its parent have slopes that *strictly* fall within the angle-range $\langle a_1(u), a_2(u) \rangle$ defined for vertex u. $\langle a_1(u), a_2(u) \rangle$ is called the *angle-range of* u with $a_1(u)$ and $a_2(u)$ being its *boundaries*. The convex angle formed between two half-lines with slopes $a_1(u)$ and $a_2(u)$ is denoted by $\phi_u = a_2(u) - a_1(u)$ and is called *angle-range length* of u.
Angelini et al. [1] proved the following theorems:

Theorem 1 (Angelini et al.[1]). *Every monotone drawing of a tree is planar.*

Theorem 2 (Angelini et al.[1]). *Every slope-disjoint drawing of a tree is monotone.*

In order to simplify the description of our algorithm, we extend the definition of slope-disjoint tree drawings to allow for angle-ranges of adjacent vertices (parent-child relationship) or sibling vertices (children of the same parent) to share angle-range boundaries.

Definition 3. *A tree drawing* Γ *of a rooted tree* T *is called a* non-strictly slope-disjoint *drawing if the following properties are satisfied:*
1. *For every vertex* $u \in T$*, there exist two angles* $a_1(u)$ *and* $a_2(u)$*, with* $0 \leq a_1(u) < a_2(u) \leq \pi$ *such that for every edge* e *that is either in* T_u *or enters* u *from its parent, it holds that* $a_1(u) < slope(e) < a_2(u)$.
2. *For every two vertices* $u, v \in T$ *such that* v *is a child of* u*, it holds that* $a_1(u) \leq a_1(v) < a_2(v) \leq a_2(u)$.
3. *For every two vertices* v_1, v_2 *with the same parent, it holds that either* $a_1(v_1) < a_2(v_1) \leq a_1(v_2) < a_2(v_2)$ *or* $a_1(v_2) < a_2(v_2) \leq a_1(v_1) < a_2(v_1)$.

In our extended definition, we allow for angle-ranges of adjacent vertices (parent-child relationship) or sibling vertices (children of the same parent) to share angle-range boundaries. Note that replacing the "$\leq$" symbols in our definition by the "$<$" symbol gives us the original definition of Angelini et al. [1] for the slope disjoint tree drawings.

Lemma 4. *Every non-strictly slope-disjoint drawing of a tree* T *is also a slope-disjoint drawing.*

Proof. Intuitively, the theorem holds since we can always adjust (by a tiny amount) the angle-ranges of vertices that share an angle-range boundary so that, after the adjustment no two tree vertices share an angle-range boundary. Note that the actual drawing of the tree does not change. Only the angle-ranges are adjusted.
More formally, let Γ be a *non-strictly slope-disjoint* drawing of a tree T rooted at r. We show how to compute for every vertex u a new angle-range $\langle b_1(u), b_2(u) \rangle$ such that the current drawing of T with the new angle-range is *slope-disjoint*.
Let $e(u)$ be the edge that connects the parent of u to u in T, for $u \in T \backslash r$.
We make use of the following definitions:

$$\delta_1 = min_{u \in T \backslash r} \left(slope(e(u)) - a_1(u) \right)$$
$$\delta_2 = min_{u \in T \backslash r} \left(a_2(u) - slope(e(u)) \right)$$
$$\delta = min(\delta_1, \delta_2)$$

For any vertex $u \in T \backslash r$ it holds that:

$$\textit{slope}(e) - a_1(u) \geq \delta \tag{1}$$

$$a_2(u) - \textit{slope}(e) \geq \delta \tag{2}$$

By Property-1 of the non-strictly slope-disjoint drawing, we have that $\delta_1, \delta_2 > 0$ and, therefore, $\delta > 0$. By adding the two previous inequalities we get that,

$$(1) + (2) \Rightarrow a_2(u) - a_1(u) \geq 2\delta \text{ where } u \in T \backslash r \tag{3}$$

For any descendant v of the root r of T, by inductive use of Property-2 of the non-strictly slope-disjoint drawings, it holds that:

$$a_1(r) \leq a_1(v) \tag{4}$$

$$a_2(r) \geq a_2(v) \tag{5}$$

By subtracting (5) from (4) we get

$$(5) - (4) \Rightarrow a_2(r) - a_1(r) \geq a_2(v) - a_1(v) \overset{(3)}{\geq} 2\delta$$

Therefore, for any vertex $u \in T$ it holds:

$$a_2(u) - a_1(u) \geq 2\delta \tag{6}$$

Let the root r of T be at level-0, let u be a vertex in level-i, $i > 0$ and let h be the height of tree T. Define the slope-disjoint angle-ranges $\langle b_1(u), b_2(u) \rangle$ for each vertex $u \in T$ as follows:

$$b_1(u) = \begin{cases} a_1(r) & \text{if } u = r \\ a_1(u) + \delta \cdot \frac{i}{h+1} & \text{if } u \neq r \end{cases} \qquad b_2(u) = \begin{cases} a_2(r) & \text{if } u = r \\ a_2(u) - \delta \cdot \frac{i}{h+1} & \text{if } u \neq r \end{cases}$$

Firstly, we show that the new angle-range boundaries $b_1(\cdot)$, $b_2(\cdot)$ satisfy Property-2 of slope-disjoint drawings. Let u be a level-i vertex $\in T$ and v be its child. By the non-strictly slope-disjoint Property-2, it holds that:

$$\begin{aligned} a_1(u) \leq a_1(v) \Rightarrow a_1(u) + \delta \cdot \frac{i}{h+1} &< a_1(v) + \delta \cdot \frac{i+1}{h+1} \\ \Leftrightarrow b_1(u) &< b_1(v) \end{aligned}$$

Similarly, we have that $b_2(v) < b_2(u)$. We also have,

$$\begin{aligned} b_2(v) - b_1(v) &= a_2(v) - \delta \cdot \frac{i+1}{h+1} - \left(a_1(v) + \delta \cdot \frac{i+1}{h+1}\right) \\ &= (a_2(v) - a_1(v)) - 2\delta \cdot \frac{i+1}{h+1} \\ &\overset{(6)}{\geq} 2\delta - 2\delta \cdot \frac{i+1}{h+1} \\ &= 2\delta \cdot \left(1 - \frac{i+1}{h+1}\right) \\ &= 2\delta \cdot \frac{h-i}{h+1} \\ &> 0 \\ &\Rightarrow b_1(v) < b_2(v) \end{aligned}$$

The last inequality holds since vertex u has a child and, thus, u is at a level i such that $i < h$. Thus, Property-2 holds.
Secondly, we show that the new angle-range boundaries $b_1(\cdot)$, $b_2(\cdot)$ satisfy Property-3 of slope-disjoint drawings. Let v_1, and v_2 be two level-i vertices having the same parent. Then, by Property-3 of the non-strictly slope disjoint drawings we have that $a_1(v_1) < a_2(v_1) \leq a_1(v_2) < a_2(v_2)$ or $a_1(v_2) < a_2(v_2) \leq a_1(v_1) < a_2(v_1)$. The two cases are symmetric, so we only prove that $b_1(v_1) < b_2(v_1) < b_1(v_2) < b_2(v_2)$ when $a_1(v_1) < a_2(v_1) \leq a_1(v_2) < a_2(v_2)$. As proved for the case of Property-2, $b_1(v_1) < b_2(v_1)$ and $b_1(v_2) < b_2(v_2)$ and thus, it remains to prove that $b_2(v_1) < b_1(v_2)$. But we have that,

$$\begin{aligned} a_2(v_1) \leq a_1(v_2) &\Rightarrow a_2(v_1) - \delta \cdot \frac{i}{h+1} < a_1(v_2) + \delta \cdot \frac{i}{h+1} \\ &\Leftrightarrow b_2(v_1) < b_1(v_2) \end{aligned}$$

Finally, we turn our attention to Property-1 of slope-disjoint drawings. Angle-range boundaries $a_1(\cdot)$ and $a_2(\cdot)$ satisfy Property-1 of non-strictly slope-disjoint drawings and thus, for every vertex u at level i and for every edge e that belongs in T_u or that enters u from its parent inequality (1) holds. By definition, we have that $b_1(u) = a_1(u) + \delta \cdot \frac{i}{h+1}$ which implies

$$a_1(u) = b_1(u) - \delta \cdot \frac{i}{h+1} \tag{7}$$

$$\begin{aligned} (1) &\overset{(7)}{\Leftrightarrow} slope(e) - \left(b_1(u) - \delta \cdot \frac{i}{h+1}\right) \geq \delta \\ &\Leftrightarrow slope(e) - b_1(u) \geq \delta \cdot \left(1 - \frac{i}{h+1}\right) \\ &\Leftrightarrow slope(e) - b_1(u) \geq \delta \cdot \left(\frac{h+1-i}{h+1}\right) \\ &\Rightarrow slope(e) - b_1(u) > 0 \end{aligned}$$

The last inequality holds since $\delta > 0$ and $i < h+1$. The later is true since u is a level-i vertex where $i \leq h$.
In a similar way, we show that $b_2(u) - slope(e) > 0$ and we conclude that $b_1(u) < slope(e) < b_2(u)$. Thus, Property-1 of slope-disjoint drawing is also satisfied. ◻

THEOREM 5. *Every non-strictly slope-disjoint drawing of a tree is monotone and planar.*

Proof. By Lemma 4 every non-strictly slope-disjoint drawing of a tree T is slope-disjoint and by Theorem 1 and Theorem 2 it is monotone and planar. ◻

2.2. Locating Points on the Grid. Based on geometry, we now prove that it is always possible to identify points on a grid that satisfy several properties with respect to their location.

LEMMA 6. *Consider two angles θ_1, θ_2 with $0 \leq \theta_1 < \theta_2 \leq \frac{\pi}{4}$ and let $d = \lceil \frac{1}{\theta_2 - \theta_1} \rceil$. Then, edge e connecting the origin $(0,0)$ to point $p = (d, \lfloor tan(\theta_1) \cdot d + 1 \rfloor)$ satisfies $\theta_1 < slope(e) < \theta_2$.*

Proof. Let l_1 and l_2 be the half-lines from origin with slopes θ_1 and θ_2, respectively. Let a and b be the intersection points of l_1 and l_2 with line $x = d$, respectively. We prove that $|ab| > 1$, so a point of the grid must lie between a and b, since the x-coordinate is integer and line segment ab is parallel to y-axis as seen in Figure 1. From trigonometry, we know identities :

$$tan(a-b) = \frac{tan(a) - tan(b)}{1 + tan(a) \cdot tan(b)} \tag{8}$$

and

$$tan(a) > a \text{ , when } 0 < a < \frac{\pi}{2} \tag{9}$$

By (8), it holds that $tan(a) - tan(b) = tan(a-b) \cdot (1 + tan(a)tan(b))$ and thus, for $0 \leq a, b \leq \frac{\pi}{2}$ it holds:

$$tan(a) - tan(b) > tan(a-b), \text{ when } 0 \leq a, b \leq \frac{\pi}{2} \tag{10}$$

The coordinates of point a are $(d, tan(\theta_1) \cdot d)$ while the coordinates of point b are $(d, tan(\theta_2) \cdot d)$. Therefore,

$$\begin{aligned}
|ab| &= tan(\theta_2) \cdot d - tan(\theta_1) \cdot d \\
&= (tan(\theta_2) - tan(\theta_1)) \cdot d \\
&\overset{(10)}{>} tan(\theta_2 - \theta_1) \cdot d \\
&\overset{(9)}{\geq} (\theta_2 - \theta_1) \cdot d \\
&= (\theta_2 - \theta_1) \cdot \lceil \frac{1}{\theta_2 - \theta_1} \rceil \\
&\geq (\theta_2 - \theta_1) \cdot \frac{1}{\theta_2 - \theta_1} \\
&= 1
\end{aligned}$$

Given that $|ab| > 1$, the grid point $p = (d, \lfloor tan(\theta_1) \cdot d + 1 \rfloor)$ falls within the angular sector defined by half-lines l_1 and l_2 and satisfies the lemma. ∎

LEMMA 7. *Consider angles θ_1, θ_2 with $0 \leq \theta_1 < \theta_2 \leq \frac{\pi}{2}$ and let $d = \lceil \frac{1}{\theta_2 - \theta_1} \rceil$. Then, a grid point p such that the edge e that connects the origin $(0, 0)$ to p satisfies $\theta_1 < slope(e) < \theta_2$, can be identified as follows:*

$$\theta_2 - \theta_1 > \frac{\pi}{4} : \quad p = (1, 1)$$

$$\frac{\pi}{4} \geq \theta_2 - \theta_1 > arctan(\frac{1}{2}) : \begin{cases} p = (1, 2) & if \ \ \theta_1 \geq \frac{\pi}{4} \\ p = (1, 1) & if \ \ \frac{\pi}{4} > \theta_1 \geq arctan(\frac{1}{2}) \\ p = (2, 1) & if \ \ arctan(\frac{1}{2}) > \theta_1 \end{cases}$$

$$arctan(\frac{1}{2}) \geq \theta_2 - \theta_1 : \begin{cases} p = (d, \lfloor tan(\theta_1) \cdot d + 1 \rfloor) & if \ \ \frac{\pi}{4} \geq \theta_2 > \theta_1 \geq 0 \\ p = (1, 1) & if \ \ \theta_2 > \frac{\pi}{4} > \theta_1 \\ p = (\lfloor tan(\frac{\pi}{2} - \theta_2) \cdot d + 1 \rfloor, d) & if \ \ \theta_2 > \theta_1 \geq \frac{\pi}{4} \end{cases}$$

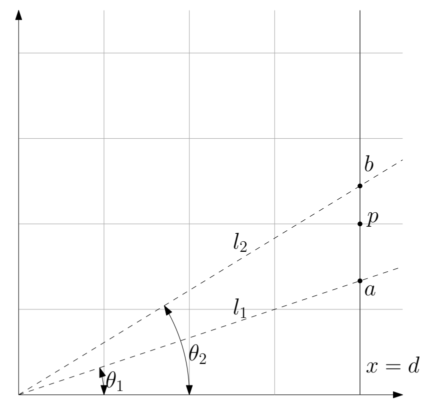

FIGURE 1. *Geometric representation of Lemma 6.*

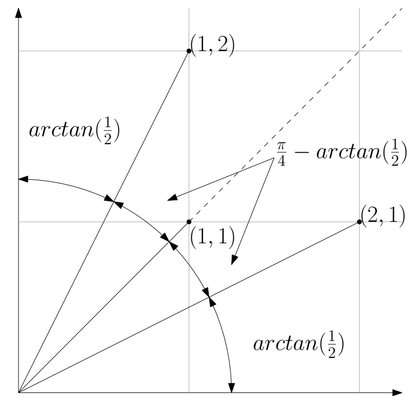

FIGURE 2. *Point, slopes angular sectors used in Lemma 7.*

Moreover, if $p = (x, y)$ is the identified point, it also holds that:

$$\max(x, y) \leq \frac{\pi}{2} \cdot \frac{1}{\theta_2 - \theta_1}$$

Proof. See Figure 2 for points, slopes and angular sectors relevant to Lemma 7. For each case, we show that the identified points in the statement of the lemma satisfy the "slope" ("$\theta_1 < slope(e) < \theta_2$") and the "length" ("$\max(x, y) < \ldots$") conditions.

Case-1: $\theta_2 - \theta_1 > \frac{\pi}{4}$. Point $(1, 1)$ is the identified point. In this case, the edge e from the origin $(0, 0)$ to $(1, 1)$ has slope $\frac{\pi}{4}$. For the "slope" condition, given that $0 \leq \theta_1 < \theta_2 \leq \frac{\pi}{2}$ and $\theta_2 - \theta_1 > \frac{\pi}{4}$, it is enough to show that $\theta_1 < \frac{\pi}{4} < \theta_2$ which implies that $\theta_1 < slope(e) < \theta_2$. If $\theta_1 > \frac{\pi}{4}$ we have that,

$$\begin{aligned}\theta_2 - \theta_1 > \frac{\pi}{4} \Leftrightarrow \theta_2 &> \theta_1 + \frac{\pi}{4} \\ &> \frac{\pi}{4} + \frac{\pi}{4} \\ &= \frac{\pi}{2}\end{aligned}$$

A clear contradiction. So, $\theta_1 < \frac{\pi}{4}$. In a similar way we can show that $\theta_2 > \frac{\pi}{4}$. For the "length" condition, we have to show that,

$$max(1, 1) = 1 \leq \frac{\pi}{2} \cdot \frac{1}{\theta_2 - \theta_1}$$

This is true since,

$$\begin{aligned}0 \leq \theta_1 < \theta_2 \leq \frac{\pi}{2} &\Rightarrow \theta_2 - \theta_1 \leq \frac{\pi}{2} \\ &\Leftrightarrow \frac{1}{\theta_2 - \theta_1} \geq \frac{1}{\frac{\pi}{2}} \\ &\Leftrightarrow \frac{\pi}{2} \frac{1}{\theta_2 - \theta_1} \geq \frac{\pi}{2} \cdot \frac{1}{\frac{\pi}{2}} = 1\end{aligned}$$

Case-2: $\frac{\pi}{4} \geq \theta_2 - \theta_1 > arctan(\frac{1}{2})$. We first establish the "slope" condition.

For the case where $arctan(\frac{1}{2}) > \theta_1$ the identified point is $(2, 1)$. We note that the slope of the edge e from the origin $(0, 0)$ to $(2, 1)$ is $slope(e) = arctan(\frac{1}{2})$. Then, by the assumption we have:

$$\begin{aligned}\theta_2 - \theta_1 > arctan\left(\frac{1}{2}\right) \Leftrightarrow \theta_2 &> \theta_1 + arctan\left(\frac{1}{2}\right)\\ &\geq arctan\left(\frac{1}{2}\right)\end{aligned}$$

It follows that $\theta_1 < arctan(\frac{1}{2}) < \theta_2 \Rightarrow \theta_1 < slope(e) < \theta_2$.
For the case where $\frac{\pi}{4} > \theta_1 \geq arctan(\frac{1}{2})$ the identified point is $(1, 1)$. We note that the slope of the edge e from the origin $(0, 0)$ to $(1, 1)$ is $slope(e) = \frac{\pi}{4}$. By the assumption, and by taking into account that $arctan(\frac{1}{2}) > \frac{\pi}{8}$, we have:

$$\begin{aligned}\theta_2 - \theta_1 > arctan\left(\frac{1}{2}\right) \Leftrightarrow \theta_2 &> \theta_1 + arctan\left(\frac{1}{2}\right)\\ &\geq arctan\left(\frac{1}{2}\right) + arctan\left(\frac{1}{2}\right)\\ &= 2 \cdot arctan\left(\frac{1}{2}\right)\\ &> 2 \cdot \frac{\pi}{8}\\ &= \frac{\pi}{4}\end{aligned}$$

It follows that $\theta_1 < \frac{\pi}{4} < \theta_2 \Rightarrow \theta_1 < slope(e) < \theta_2$.
For the case where $\theta_1 \geq \frac{\pi}{4}$ the identified point is $(1, 2)$. We note that the slope of the edge e from the origin $(0, 0)$ to $(1, 2)$ is $slope(e) = arctan(2)$. We want to establish that $\theta_1 < arctan(2) < \theta_2$. This can be easily proved by taking into account that $arctan(2) = \frac{\pi}{2} - arctan(\frac{1}{2})$ as well as that $2 \cdot arctan(\frac{1}{2}) > \frac{\pi}{4}$.
For the "length" condition, it is enough to show that:

$$max(max(2, 1), max(1, 1), max(1, 2)) = 2 \leq \frac{\pi}{2} \cdot \frac{1}{\theta_2 - \theta_1}$$

This is true since,

$$\begin{aligned}\theta_2 - \theta_1 &\leq \frac{\pi}{4}\\ \Leftrightarrow \frac{1}{\theta_2 - \theta_1} &\geq \frac{1}{\frac{\pi}{4}}\\ \Leftrightarrow \frac{\pi}{2}\frac{1}{\theta_2 - \theta_1} &\geq \frac{\pi}{2} \cdot \frac{1}{\frac{\pi}{4}}\\ &= 2\end{aligned}$$

Case-3: $arctan(\frac{1}{2}) \geq \theta_2 - \theta_1$**.** We first establish the "slope" condition. In the case where $\frac{\pi}{4} \geq \theta_2 > \theta_1 \geq 0$, by Lemma 6 the identified point immediately satisfies the "slope" condition. The same holds for the symmetric case where $\theta_2 > \theta_1 \geq \frac{\pi}{4}$. Finally, in the case where $\theta_2 > \frac{\pi}{4} > \theta_1$ the slope condition trivially holds since the edge from the origin $(0, 0)$ to $(1, 1)$ has slope $\frac{\pi}{4}$.

For the "length" condition, we note that in all three cases we have that,

$$\begin{aligned}\max(x,y) \le d &= \lceil \frac{1}{\theta_2-\theta_1} \rceil \\ &< \frac{1}{\theta_2-\theta_1}+1\end{aligned}$$

But it also holds:

$$\frac{1}{x}+1 \le \frac{\pi}{2}\cdot\frac{1}{x}\text{ , where } 0 < x \le \frac{\pi}{2}-1 \tag{11}$$

And since $\theta_2-\theta_1 \le arctan\left(\frac{1}{2}\right) < \frac{\pi}{2}-1$ we get that,

$$max(x,y) < \frac{1}{\theta_2-\theta_1}+1 \overset{(11)}{\le} \frac{\pi}{2}\cdot\frac{1}{\theta_2-\theta_1}$$

The lemma is now proved. ∎

3. One-Quadrant "Traditional" Monotone Drawing of Rooted Ordered Trees. In this Section, we describe an algorithm that builds a monotone drawing of an n-vertex tree on a grid of size at most $n \times n$. We refer to this algorithm as "traditional" since it satisfies all drawing conventions followed by all algorithms that have appeared in the literature, that is, it take as input a *rooted ordered tree* T and produces a monotone drawing of T where (i) the root of T is drawn at the origin of the drawing, (ii) the drawing of T is confined in the first quadrant, and (iii) the order of the children of each node of T is respected. The algorithm produces a non-strictly slope-disjoint tree drawing which, by Theorem 5, is monotone and planar.
***
In order to describe a non-strictly slope-disjoint tree drawing, we need to identify for each vertex u of the tree a grid point to draw u as well as to assign to it two angles $a_1(u)$, $a_2(u)$, with $a_2(u) > a_1(u)$. For every tree node, the identified grid point and the two angles should be such that the three properties of the non-strictly slope-disjoint drawings are satisfied.
The basic idea behind our algorithm is to split in a balanced way the angle-range $\langle a_1(u), a_2(u)\rangle$ of vertex u to its children based on the size of the subtrees rooted at them. The following strategy formalizes this idea.

Strategy 1. *Let u be a non-leaf vertex of an n-vertex rooted tree T such that we already have assigned values for $a_1(u)$ and $a_2(u)$, with $a_1(u) < a_2(u)$. Let $v_1, v_2, \ldots, v_m$, $m \ge 1$ be the children of u. We assign angle-range for the children of u in the following way:*

$$a_1(v_i) = \begin{cases} a_1(u) & \text{if } \ i=1 \\ a_2(v_{i-1}) & \text{if } \ 1 < i \le m \end{cases}$$

$$a_2(v_i) = a_1(v_i) + (a_2(u)-a_1(u))\cdot\frac{|T_{v_i}|}{|T_u|-1}, \quad 1 \le i \le m$$

The following lemma proves that Strategy 1 satisfies Property-2 and Property-3 of the non-strictly slope-disjoint drawings.

Lemma 8. *Let u be a vertex of the rooted tree T such that we already have assigned values for $a_1(u)$ and $a_2(u)$, with $a_1(u) < a_2(u)$. Let $v_1, v_2, \ldots, v_m$, $m \ge 1$, be the* *children of u in T. If we assign values for angle-ranges of the children of u according to Strategy 1, then Property-2 and Property-3 of the non-strictly slope-disjoint drawings are satisfied.*

Proof. For Property-3, we have to show that for every k, l, $1 \leq k < l \leq m$, it holds: $a_1(v_k) < a_2(v_k) \leq a_1(v_l) < a_2(v_l)$. For any j, $1 \leq j \leq m$, we have that,

$$\begin{aligned} a_2(v_j) &= a_1(v_j) + (a_2(u) - a_1(u)) \cdot \frac{|T_{v_j}|}{|T_u| - 1} \\ &> a_1(v_j) \end{aligned}$$

The last inequality holds since, by assumption, $a_2(u) - a_1(u) > 0$ and because the size of a rooted tree is always positive. Therefore,

$$\begin{aligned} a_1(v_1) &< a_2(v_1) \\ &= a_1(v_2) \\ &< a_2(v_2) \\ &= a_1(v_3) \\ &\vdots \\ &< a_2(v_{m-1}) \\ &= a_1(v_m) \\ &< a_2(v_m) \end{aligned}$$

So, for any k, l, $1 \leq k < l \leq m$, it holds that $a_1(v_k) < a_2(v_k) \leq a_1(v_l) < a_2(v_l)$ and, thus, Property-3 holds.
For Property-2, since we proved that $a_1(v_1) < a_2(v_1) \leq a_1(v_2) < \ldots < a_2(v_{m-1}) \leq a_1(v_m) < a_2(v_m)$, it is sufficient to show that $a_1(u) \leq a_1(v_1)$ and $a_2(v_m) \leq a_2(u)$. The first part trivially holds since $a_1(v_1) = a_1(u)$ by definition. For the second part, by using repeatedly the assignment for a_1 and a_2 provided in the statement of the lemma we get that,

$$a_2(v_m) = a_1(v_1) + (a_2(u) - a_1(u)) \frac{\sum_{i=1}^{m} |T_{v_i}|}{|T_u| - 1}$$

Since the subtree rooted at u, consists of the root vertex u and the subtrees rooted at u's children, it holds that $|T_u| = \sum_{i=1}^{n} |T_{v_i}| + 1$. It follows that $a_2(v_m) = a_2(u)$ and Property-2 is satisfied. ◻

Observation 1. *If a vertex u has only one child, say v_1, then the angle assignment Strategy 1 assigns $a_1(v_1) = a_1(u)$ and $a_2(v_1) = a_2(u)$, which means that the child "inherits" the angle-range of its parent.*

Algorithm 1 describes our monotone tree drawing algorithm. It consists of three steps: Procedure AssignAngles which assigns angle-ranges to the vertices of the tree according to Strategy 1, Procedure DrawVertices which assigns each tree vertex to a grid point according to Lemma 7 and Procedure BalancedTreeMonotoneDraw which assigns the root to point $(0, 0)$ with angle-range $\left\langle 0, \frac{\pi}{2} \right\rangle$ and initiates the drawing of the tree.

Lemma 9. *The drawing produced by Algorithm 1 is monotone and planar.*

Algorithm 1 One-Quadrant Monotone Rooted Ordered Tree Drawing

procedure BalancedTreeMonotoneDraw
Input: An n-vertex tree T rooted at vertex r.
Output: A monotone drawing of T on a grid of size at most $n \times n$.
    $a_1(r) \leftarrow 0,\ a_2(r) \leftarrow \frac{\pi}{2}$
    AssignAngles$(r,\ a_1(r),\ a_2(r))$
    Draw r at $(0, 0)$
    DrawVertices(r)

procedure AssignAngles(u, a_1, a_2)
Input: A vertex u and the boundaries of the angle-range $\langle a_1, a_2 \rangle$ assigned to u.
Action: It assigns angle-ranges to the vertices of T_u.
    for each child v_i of u **do**
        Assign $a_1(v_i),\ a_2(v_i)$ as described in Strategy 1.
        AssignAngles$(v_i,\ a_1(v_i),\ a_2(v_i))$

procedure DrawVertices(u)
Input: A vertex u where u has already been drawn of the grid and angle-ranges have been defined for all vertices of T_u.
Action: It draws the vertices of T_u.
    for each child v_i of u **do**
        Find a valid pair (x, y) as described in Lemma 7 where
            $\theta_1 \leftarrow a_1(u)$ and $\theta_2 \leftarrow a_2(u)$
        If u is drawn at (u_x, u_y), draw v_i at $(u_x + x, u_y + y)$
        DrawVertices(v_i)

Proof. The angle-range assignment of Strategy 1 satisfies Property-2 and Property-3 of the non-strictly slope disjoint drawing as proved in Lemma 8. In addition, the assignment of the vertices to grid points satisfies Property-1 of the non-strictly slope disjoint drawing as proved in Lemma 7. Thus, the produced drawing by Algorithm 1 is non-strictly slope disjoint and, by Theorem 5, it is monotone and planar. ∎

It remains to establish a bound on the grid size required by Algorithm 1. Our proof uses induction on the number of tree vertices having more than one child.

Lemma 10. *Let T be a rooted tree and $u \in T$ be a vertex. Consider $\phi_u = a_2(u) - a_1(u)$ as assigned by Algorithm 1. Then the side-length of the grid which Algorithm 1 uses for the drawing of the subtree T_u rooted at u is bounded by:*

$$(|T_u| - 1)\frac{\pi}{2}\frac{1}{\phi_u}$$

Proof. We use induction on the number of vertices having at least two children. Let i be the number of vertices of $\in T_u$ with at least two children.

Base Case (i=0): In that case, T_u is just a path and by Observation 1, Algorithm 1 assigns to every vertex the same angle-range. From this observation, for any vertex $v \in T_u$, it holds that $a_2(v) - a_1(v) = a_2(u) - a_1(u) = \phi_u$, therefore by Lemma 7 we have that each edge expands our grid at most by:

$$\frac{\pi}{2}\frac{1}{\phi_u}$$

Since the tree has $|T_u|$ vertices, we expand the grid $|T_u|-1$ times, therefore the side-length of the grid required for the drawing of tree T_u is:

$$(|T_u|-1)\frac{\pi}{2}\frac{1}{\phi_u}$$

The base case is now settled.

Induction Step: We assume that for any rooted subtree which contains at most i vertices with at least two children each, the statement holds. We prove that for any subtree rooted at vertex u with $i+1$ vertices in T_u having at least two children each, the statement also holds.

At first we prove that the only case of interest is when the subtree is rooted at a vertex with at least two children. Let's assume T_u is the union of a path starting from u and ending at v where each vertex has exactly one child except v and the subtree rooted at v. The number of vertices in T_v having at least two children is $i+1$ by assumption since the vertices in the path between u and v have exactly one child. If the statement holds for v we have, by Observation 1, $a_2(v)=a_2(u)$ and $a_1(v)=a_1(u)$, and thus,

$$\phi_v = a_2(v)-a_1(v) = a_2(u)-a_1(u) = \phi_u \tag{12}$$

The side-length of the required grid for T_v is,

$$(|T_v|-1)\frac{\pi}{2}\frac{1}{\phi_v} \overset{(12)}{=} (|T_v|-1)\frac{\pi}{2}\frac{1}{\phi_u}$$

Also, the side-length of grid required for the path from u to v, having $|T_u|-|T_v|$ vertices, is

$$(|T_u|-|T_v|)\frac{\pi}{2}\frac{1}{\phi_u}$$

So, the total side-length of the required grid is:

$$(|T_v|-1)\frac{\pi}{2}\frac{1}{\phi_u}$$

Therefore, it is enough to only consider the case where the root u of the subtree T_u has at least two children.

Let u be a vertex $\in T$ such that u has at least two children and T_u has $i+1$ vertices with at least two children each. Let $v_1, v_2, \dots, v_m$ be the children of u and observe that the largest grid devoted to any of the trees[1]. $T^u_{v_j}, 1 \le j \le m$, determines the side-legth of the grid drawing of T_u since the subtrees rooted at the children of u are drawn completely inside non-overlapping (but possibly touching) angular sectors. The above statement holds because all the grids that are used for the subtrees have the same origin (u) and all angular sectors

[1] Recall that by T^u_v, where v is a child of u, we denote the tree that consists of edge (u,v) and T_v

lies in the first quadrant since Algorithm 1 assigns to the root angle-range $\left\langle 0, \frac{\pi}{2}\right\rangle$. Therefore, the side-length of the grid required in order to draw T_u is equal to the maximum of the grid side-lengths required to draw any of $T^u_{v_j}$. For any vertex v_j, since vertex u has at least two children, it holds that the number of vertices in T_{v_j} having at least two children each is less or equal to i, and therefore the induction hypothesis applies. Thus, T_{v_j} is drawn on a grid with side-length bounded by,

$$(|T_{v_j}| - 1)\frac{\pi}{2}\frac{1}{\phi_{v_j}}$$

For the edge connecting u to v_j, by Lemma 7 we require a grid of side-length bounded by,

$$\frac{\pi}{2}\frac{1}{\phi_{v_j}}$$

Therefore, the required grid has total side-length bounded by:

$$|T_{v_j}|\frac{\pi}{2}\frac{1}{\phi_{v_j}}$$

Since we applied Strategy 1, it holds that

$$\phi_{v_j} = \frac{|T_{v_j}|}{|T_u| - 1}\phi_u \tag{13}$$

Thus, the required total grid side-length required can be restated as:

$$\begin{aligned} |T_{v_j}|\frac{\pi}{2}\frac{1}{\phi_{v_j}} &\overset{(13)}{=} |T_{v_j}|\frac{\pi}{2}\frac{1}{\frac{|T_{v_j}|}{|T_u|-1}\phi_u} \\ &= (|T_u| - 1)\frac{\pi}{2}\frac{1}{\phi_u} \end{aligned}$$

Therefore, the statement holds for the induction step. This completes the proof of the lemma. ∎

THEOREM 11. *Given a rooted n-vertex Tree T, Algorithm 1 produces a monotone grid drawing using a grid of size at most $n \times n$.*

Proof. The monotonicity of the drawing follows directly from Lemma 9. By applying Lemma 10 to the root of the tree which is assigned angular sector $< 0, \frac{\pi}{2} >$ (and thus, $\theta_2 = \frac{\pi}{2}$ and $\theta_1 = 0$) we get that in the worst case the drawing of T uses a grid of side-length that is smaller or equal to:

$$(n-1)\,\frac{\pi}{2}\frac{1}{\frac{\pi}{2}} = n - 1$$

Therefore, the required grid is of size at most $n \times n$. ∎

Figures 3-5 present drawings produced by Algorithm 1. Figure 3 shows the drawing of a 5-layer complete binary tree (31 vertices). While Theorem 11 indicates that a grid of size 31×31 may be required, the binary tree is drawn on a 23×22 grid. Figure 4 shows the drawing of a path (15 vertices). All paths that are rooted at one of their endpoints are drawn along the main diagonal of a grid with side-length matching the bound stated in Theorem 11. Finally, Figure 5 shows a drawing of tree (out of all 10-vertex rooted trees) that requires maximum area (when produced by Algorithm 1. We have drawn all 10-vertex rooted trees and have identified non-path trees that require a grid of the dimensions stated in Themorem 11.

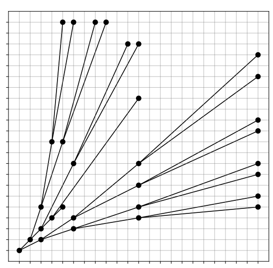

FIGURE 3. *A full binary tree (31 vertices) as drawn by Algorithm 1. Grid size:* 23×22.

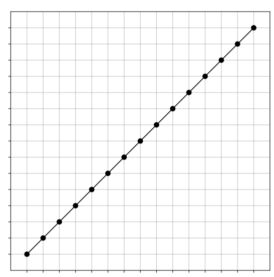

FIGURE 4. *A path (15 vertices) as drawn by Algorithm 1. Grid size:* 15×15.

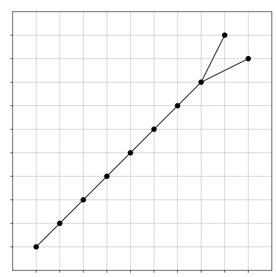

FIGURE 5. *A non-path tree (10 vertices) with maximum required area when drawn by Algorithm 1. Grid size:* 10×10.

4. Two-Quadrants Monotone Unrooted Ordered Tree Drawing. In this Section, we examine monotone drawings for unrooted ordered trees. Our approach is to carefully select a vertex r and designate it as the root of the tree. The produced tree drawing occupies the first two quadrants, with respect to the location of its root r which is drawn at the origin. We note that the drawing respects the initial embedding of the tree, that is, the order of the neighbors of each vertex around it is maintained. The ability to choose a vertex r and to designate it as the root of the tree, in addition to the use of the first two quadrants, allows us to reduce the used grid to at most $n \times \frac{n}{2}$.

We first describe how to select the vertex to be designated as the root of the tree. A desirable property of the root node, given the nature of our algorithm, is that its children have as much balanced subtrees (with respect to their number of vertices) as possible.

Let T be an unrooted tree. Let $r \in T$ be a vertex such that if we root T at r then for any child v of r, the size of subtree T_v is $|T_v| \leq \frac{n}{2}$. We refer to r as a *gravity root* of T. Therefore, if an n-vertex tree T is rooted at a *gravity root* vertex r there is no vertex $u \in T \backslash r$ such that $|T_u| > \frac{n}{2}$. Also, if $n \geq 3$, r has at least two children.

We now show that every tree has a gravity root.

LEMMA 12. *Let T be an n-vertex unrooted tree. Algorithm 2 always succeeds in identifying a gravity root of T .*

Proof. At each iteration, Algorithm 2 gets a step closer to finding a gravity root. Denote by $T_{lcc}(r)$ the largest connected component of $T \backslash r$. By definition, vertex r is a gravity root if $|T_{lcc}(r)| \leq \frac{n}{2}$. We show that at each iteration of Algorithm 2 the value of $|T_{lcc}(r)|$ decreases; this continues until a gravity root is reached.

Algorithm 2 Identify a gravity root

procedure GRAVITYROOTFINDER(T)
  Input: A unrooted tree T.
  Output: A gravity root vertex r.
    $r \leftarrow$ An arbitary vertex $u \in T$
    while r is not a gravity root **do**
      $u \leftarrow$ the vertex connected to r which lies in the largest connected component of $T \backslash r$.
      $r \leftarrow u$

Assume that r is not a gravity root. Then, $|T_{lcc}(r)| \geq \frac{n+1}{2}$. Let u be the neighbor of r in $T_{lcc}(r)$. Since Algorithm 2 selects vertex u as the root for next iteration, it is enough to show that $|T_{lcc}(u)| < |T_{lcc}(r)|$.
Note that the connected component of $T \backslash u$ that contains r has size less or equal than $n - \frac{n+1}{2} = \frac{n-1}{2}$. Thus, if u is not a gravity root then the next candidate gravity root will be a neighbor of u in $T_{lcc}(r)$. Thus, $T_{lcc}(u)$ will be a proper subtree of $T_{lcc}(r)$, and therefore, $|T_{lcc}(u)| < |T_{lcc}(r)|$.
We conclude that the value of $|T_{lcc}(r)|$, where r is the candidate gravity root in Algorithm 2 decreases with each iteration until a gravity root is selected. ∎

By rooting a tree at a *gravity root*, we can obtain a monotone drawing with bounded angle-range length for any subtree rooted at a child of the root. This is formalized in the Lemma that follows. Let function $odd() : \mathbb{N} \rightarrow \{0, 1\}$ evaluate to 1 when its parameter is odd, otherwise it evaluates to 0.

LEMMA 13. *Let T be an n-vertex tree rooted at a* gravity root r. *Let $\langle \theta_1, \theta_2 \rangle$ be the angle-range of r. Strategy 1 assigns at each vertex $u \in T \backslash r$ angle-range of length at most $\frac{\theta_2 - \theta_1}{2} \frac{n - odd(n)}{n-1}$.*

Proof. Let T be an n-vertex tree rooted at a *gravity root* r. Since T is rooted at a gravity root then, for any child u of r it holds that $|T_u| \leq \frac{n}{2}$. Furthermore, if n is odd then it holds that $|T_u| \leq \frac{n-1}{2}$ since the size of a subtree must be an integer. By making use of the *odd*() function, we have that for any child u of r it holds that $|T_u| \leq \frac{n - odd(n)}{2}$.
By Strategy 1, we assign to each child u of r an angle-range of length:

$$\begin{aligned} a_2(u) - a_1(u) &= (\theta_2 - \theta_1) \frac{|T_u|}{n-1} \\ &\leq (\theta_2 - \theta_1) \frac{\frac{n - odd(n)}{2}}{n-1} \\ &= \frac{\theta_2 - \theta_1}{2} \frac{n - odd(n)}{n-1} \end{aligned}$$

We complete the proof by noticing that the observation holds not only for the children of r but also for any other vertex of $T \backslash r$. This is due to the fact that Strategy 1 always assigns to a vertex of T an angle-range of length equal or smaller to that of its parent. ∎

OBSERVATION 2. *Let T be an n-vertex tree, $n > 2$, rooted at a gravity root r. Then, r has at least two children.*

Proof. If we assume that r has only one child, say u, then $|T_u| = |T| - 1 > \frac{|T|}{2}$ (for $n > 2$), a contradiction since we assumed r is a gravity root. ∎

In our "two-quadrant" algorithm we again use Strategy 1 for angle assignment but, we now assign the gravity root of the input tree T angle-range $\langle 0, \pi \rangle$ instead of $\langle 0, \frac{\pi}{2} \rangle$. Consequently, in order to assign grid points to tree vertices we need to extend Lemma 7 to cover the case where a vertex has angle-range boundary $\theta_2 > \frac{\pi}{2}$.

LEMMA 14. *Consider angles β_1, β_2 with $0 \leq \beta_1 < \beta_2 \leq \pi$. Then, a grid point p such that the edge e that connects the origin $(0,0)$ to p satisfies $\beta_1 < slope(e) < \beta_2$, can be identified as follows:*

$$p = \begin{cases} (0,1) & \text{if } \beta_1 < \frac{\pi}{2} < \beta_2 \\ (x,y) & \text{if } \beta_2 \leq \frac{\pi}{2}, \text{ where } (x,y) \text{ is a valid pair according to Lemma 7} \\ & \text{where } \theta_1 \leftarrow \beta_1 \text{ and } \ \theta_2 \leftarrow \beta_2 \\ (-x,y) & \text{if } \beta_1 \geq \frac{\pi}{2}, \text{ where } (x,y) \text{ is a valid pair according to Lemma 7} \\ & \text{where } \theta_1 \leftarrow \pi - \beta_2 \text{ and } \theta_2 \leftarrow \pi - \beta_1 \end{cases}$$

Proof. We prove the lemma by taking cases depending on the value of β_1 and β_2.
Case-1: $\beta_1 < \frac{\pi}{2} < \beta_2$. It is clear that $\beta_1 < \frac{\pi}{2} < \beta_2 \Leftrightarrow \beta_1 < slope(e) < \beta_2$.
Case-2: $\beta_2 \leq \frac{\pi}{2}$. From Lemma 7 it holds that $\theta_1 < slope(e) < \theta_2 \Leftrightarrow \beta_1 < slope(e) < \beta_2$.
Case-3: $\beta_1 \geq \frac{\pi}{2}$. Let e' be the edge that connects the origin to (x,y). Note that $slope(e) = \pi - slope(e')$. From Lemma 7 it holds that:

$$\begin{aligned} &\theta_1 < slope(e') < \theta_2 \\ \Leftrightarrow &\pi - \beta_2 < slope(e') < \pi - \beta_1 \\ \Leftrightarrow &\beta_1 < \pi - slope(e') < \beta_2 \\ \Leftrightarrow &\beta_1 < slope(e) < \beta_2 \end{aligned}$$

□

Algorithm 3 describes our "two-quadrants" balanced monotone unrooted-tree drawing algorithm. It consists of three procedures: Procedure AssignAngles (same as in Algorithm 1) which assigns angle-ranges to the vertices of the tree according to Strategy 1, Procedure ExpandedDrawVertices which assigns each tree vertex to a grid point according to Lemma 14 and Procedure UnrootedTreeMonotoneDraw which assigns a vertex as the root, draws it to point $(0,0)$ with angle-range $\langle 0, \pi \rangle$ and initiates the drawing of the tree.
The following observation highlights the connection between Algorithm 3 and Algorithm 1.

OBSERVATION 3. *Let v be a vertex that has been assigned angle-range $\langle a_1(v), a_2(v) \rangle$ and let u be its parent which is drawn at grid point (u_x, u_y). Algorithm 3 draws T_v in the following way:*
-$a_2(v) \leq \frac{\pi}{2}$: *Algorithm 3 draws T_v in the first quadrant, in exactly the same way as Algorithm 1 does.*
-$a_1(v) \geq \frac{\pi}{2}$: *Algorithm 3 draws T_v in the second quadrant as the reflex drawing (with respect to line $l : x = u_x$) of the drawing Algorithm 1 produces for T_u if we reverse the order of the children for each vertex $x \in T_v$.*
-$a_1(v) < \frac{\pi}{2} < a_2(v)$: *Algorithm 3 draws v at the Y axis. Since all children are assigned non-overlapping angle-ranges, at most one child includes $\frac{\pi}{2}$ in its angle-range and, according to the two previous points, the other children are either drawn at the first or second quadrant.*

Algorithm 3 Two-Quadrants Monotone Tree Drawing algorithm

procedure UnrootedTreeMonotoneDraw
Input: An n-vertex unrooted tree T.
Output: A monotone drawing of T on a grid of size at most $n \times \frac{1}{2}n$.
    $r \leftarrow$ GravityRootFinder(T) (Finds a gravity root as described in
        Algorithm 2)
    $a_1(r) \leftarrow 0,\ a_2(r) \leftarrow \pi$
    AssignAngle($r,\ a_1(r),\ a_2(r)$)
    Draw r at $(0, 0)$
    ExpandedDrawVertices(r)

procedure AssignAngles(u, a_1, a_2)
Input: A vertex u and the boundaries of the angle-range $\langle a_1, a_2 \rangle$ assigned to u.
Action: It assigns angle-ranges to the vertices of T_u.
    for each child v_i of u **do**
        Assign $a_1(v_i),\ a_2(v_i)$ as described in Strategy 1.
        AssignAngles($v_i,\ a_1(v_i),\ a_2(v_i)$)

procedure ExpandedDrawVertices(u)
Input: A vertex u where u has already been drawn of the grid and angle-ranges have been defined for all vertices of T_u.
Action: It draws the vertices of T_u.
    for each child v_i of u **do**
        Find a valid pair (x, y) as described in Lemma 14 where
            $\beta_1 \leftarrow a_1(u)$ and $\beta_2 \leftarrow a_2(u)$
        If u is drawn at (u_x, u_y), draw v_i at $(u_x + x, u_y + y)$
        ExpandedDrawVertices(v_i)

By combining Lemma 13 with Lemma 14, we obtain an upper bound on the length of an edge in the drawing produced by Algorithm 3.

Lemma 15. *Let T be an n-vertex tree rooted at a* gravity root *r. Let v be a vertex in $T \backslash r$ with angle-range $\langle \theta_1, \theta_2 \rangle$ and let u be its parent. For the vector $e = (x, y)$ that connects u to v, as drawn by Algorithm 3, it holds:*

$$max(|x|, y) \leq \frac{\pi}{2} \frac{1}{\theta_2 - \theta_2} \frac{n - odd(n)}{n - 1}$$

Proof. First we note that, according to Lemma 14, the y-coordinate is always positive but the sign of the x-coordinate depends on the angle-range of v, as noted in Observation 3.

-$\theta_1 < \frac{\pi}{2} < \theta_2$: The vector that connects u to v is $e = (0, 1)$. Therefore, $max(|x|, y) = 1$. By Lemma 13, and since v is not the tree root, v has angle-range length at most $\frac{\pi}{2} \cdot \frac{n - odd(n)}{n-1}$. Therefore:

$$
\begin{aligned}
&\frac{\pi}{2}\frac{n-odd(n)}{n-1} \geq \theta_2-\theta_1\\
\Rightarrow&\frac{\pi}{2}\frac{n-odd(n)}{n-1}\frac{1}{\theta_2-\theta_1} \geq 1\\
\Rightarrow&\frac{\pi}{2}\frac{n-odd(n)}{n-1}\frac{1}{\theta_2-\theta_1} \geq max(x,y)
\end{aligned}
$$

-Otherwise: When $\theta_2 \leq \frac{\pi}{2}$ or $\theta_1 \geq \frac{\pi}{2}$, the grid point assignment is made according to Lemma 14 which, in turn, makes use of Lemma 7. By applying Lemma 7 and by noticing that $\frac{n-odd(n)}{n-1} \geq 1$, the bound is guaranteed. ◻

LEMMA 16. *The drawing produced by Algorithm 3 is monotone and planar.*

Proof. The angle-range assignment of Strategy 1 satisfies Property-2 and Property-3 of the non-strictly slope disjoint drawing as proved in Lemma 8. In addition, the assignment of the vertices to grid points satisfies Property-1 of the non-strictly slope disjoint drawing as proved in Lemma 14. Thus, the produced drawing by Algorithm 3 is non-strictly slope disjoint and, by Theorem 5, it is monotone and planar. ◻

It remains to establish a bound on the grid size required by Algorithm 3. We consentrate on trees of at least 3 vertices, since it is trivial to draw a tree with two vertices. Our proof uses induction on the number of tree vertices having more than one child.

LEMMA 17. *Let T be an n-vertex tree, $n > 2$, rooted at a* gravity root *r and Γ be the drawing of T produced by Algorithm 3. Let $u \in T$ be a vertex which, in Γ, is drawn on the Y-axis and consider $\phi_u = a_2(u) - a_1(u)$ as assigned by Algorithm 3. Let Γ_u^R and Γ_u^L be the partial drawings of T_u that lie in the first and second quadrant[2], respectively. Then, each of Γ_u^R and Γ_u^L uses a grid of side-length bounded by:*

$$(|T_u|-1)\frac{\pi}{2}\frac{n-odd(n)}{n-1}\frac{1}{\phi_u}$$

Proof. We firstly observe a property that plays a key role in the proof. All vertices $u \in T$ that are drawn by Algorithm 3 on the Y-axis satisfy, by construction, that $a_1(u) < \frac{\pi}{2} < a_2(u)$. This is due to the fact that a vertex is drawn on the Y-axis only if its placement was determined based on the first case of Lemma 14.
Secondly, we establish an inequality that holds for any vertex $u \in T \backslash r$. Given that Algorithm 3 assigns to the gravity root r angle-range $\langle 0, \pi \rangle$ and since u is not the gravity root, by Lemma 13 the angle range $\phi(u)$ of u satisfies $\phi(u) \leq \frac{\pi}{2}\frac{n-odd(n)}{n-1}$. Thus,

$$1 \leq \frac{\pi}{2}\frac{n-odd(n)}{n-1}\frac{1}{\phi(u)} \tag{14}$$

Similar to the proof of Lemma 10, we employ induction on the number of vertices having at least two children. We also make use of the "edge-length bound" provided by Lemma 15. Let i be the number of vertices in T_u with at least two children.
Base Case (i=0): In this case, T_u is just a path and, by Observation 1, Algorithm 3 assigns to every vertex of T_u the same angle-range. Since $a_1(u) < \frac{\pi}{2} < a_2(u)$,

[2] The positive Y-axis in considered to be part of both the first and the second quadrant. So, vertices that are drawn on the Y-axis appear in both Γ_u^R and Γ_u^L

for any vertex $v \in T_u$ the vector that connects v to its parent is $e = (0, 1)$. Therefore, by Algorithm 3, T_u is drawn on the Y-axis and has length $|T_u| - 1$. Thus, both Γ_u^R and Γ_u^L consist of only a path of length $|T_u| - 1$ which is drawn on the Y-axis. By Observation 2, u is not the gravity root and, thus, we can make use of (14). It immediately follows that each of Γ_u^R and Γ_u^L uses a grid of side-length bounded by:

$$(|T_u| - 1)\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_u}$$

The base case is now settled.

Induction Step: We prove the bound only for the grid side-length of Γ_u^R as the case for Γ_u^L is symmetric.

We first establish that the only case of interest is when u has two or more children. If u has only one child, say v, then, by Observation 2 u is not the gravity root. By Observation 1, v inherits the angle range of its parent and, thus, $\frac{\pi}{2}$ is contained within v's angle-range. Moreover, $\phi(u) = \phi(v)$. By Algorithm 3, the vector that connects u to v is $e = (0, 1)$. If we assume that the induction hypothesis holds for v, then the grid side-length of the Γ_u^R is bounded by the grid side-length of Γ_v^R plus the length of the vector that connects u to v. Therefore, the grid side-length of Γ_u^R is bounded by:

$$\begin{aligned}
&(|T_v| - 1)\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_v} + 1 \\
=&(|T_u| - 2)\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_u} + 1 \\
\overset{(14)}{\leq}&(|T_u| - 2)\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_u} + \frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_u} \\
=&(|T_u| - 1)\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_u}
\end{aligned}$$

Therefore, the only case of interest is when u has at least two children.

Let $u \in T$ be a vertex such that u is drawn by Algorithm 3 on the Y-axis, u has at least two children, and T_u has $i + 1$ vertices with at least two children. Let $v_1, v_2, \ldots, v_m$ be the children of u such that the drawing of T_{v_j}, $1 \leq j \leq m$, lies on the first quadrant. By Observation 3, the angle-range of any v_j must be in the form of $\langle a_1(v_j), a_2(v_j)\rangle$ where $a_2(v_j) \leq \frac{\pi}{2}$ or $a_1(v_j) < \frac{\pi}{2} < a_2(v_j)$. We note that the largest grid (wrt its side-length) on the first quadrant devoted to any tree[3] $T_{v_j}^u, 1 \leq j \leq m$, determines the grid side-length of Γ_u^R since the subtrees rooted at children of u are drawn completely inside non-overlapping (but possibly touching) angular sectors. The above statement holds because all the grids that are used for the subtrees share as common origin vertex u and we only care about all angular sectors that at least partially lie in the first quadrant. Therefore, the grid size required to draw T_u is the maximum of the grid sizes required to draw any of $T_{v_j}^u$.

For any vertex v_j with angle-range $\langle a_1(v_j), a_2(v_j)\rangle$, if $a_2(v_j) \leq \frac{\pi}{2}$, i.e., T_{v_j} lies entirely in the first quadrant, then, the statement holds from Lemma 10 and by noticing that $\frac{n - odd(n)}{n-1} \geq 1$. For the vertex v_j (there exists at most

[3]Recall that by T_v^u when v is a child of u, we denote the tree that consists of edge (u, v) and T_v

one such vertex) that $a_1(v_j) < \frac{\pi}{2} < a_2(v_j)$, the number of vertices in T_{v_j} with at least two children is less or equal to i, therefore the induction hypothesis holds for $\Gamma^R_{v_j}$. Therefore, the statement holds for the first quadrant for any T_{v_j} which is drawn on a grid with grid-length side bounded by,

$$(|T_{v_j}| - 1)\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_{v_j}}$$

For the edge connecting u to v_j, by Lemma 15 we require a grid of side-length bounded by,

$$\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_{v_j}}$$

Therefore, the total required grid has side-length bounded by:

$$|T_{v_j}|\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_{v_j}}$$

Since we employ Strategy 1, it holds that:

$$\phi_{v_j} = \frac{|T_{v_j}|}{|T_u| - 1}\phi_u \tag{15}$$

Thus, the bound on the side-length of the total required grid can be restated as:

$$\begin{aligned} |T_{v_j}|\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_{v_j}} &\stackrel{(15)}{=} |T_{v_j}|\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\frac{|T_{v_j}|}{|T_u|-1}\phi_u} \\ &= (|T_u| - 1)\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\phi_u} \end{aligned}$$

Therefore, the statement holds for the induction step. The proof of the lemma is complete. ∎

We can now state our main result regarding "two-quadrant" drawings.

THEOREM 18. *Given a rooted n-vertex tree T, Algorithm 3 produces a monotone grid drawing using a grid of size at most:*

$$\begin{aligned} n \times \left(\frac{n+1}{2}\right) &\qquad \textit{when } n \textit{ is odd} \\ (n+1) \times \left(\frac{n}{2} + 1\right) &\qquad \textit{when } n \textit{ is even} \end{aligned}$$

Proof. The monotonicity of the drawing follows directly from Lemma 16. By applying Lemma 17 with the gravity root r, where Algorithm 3 assigns $a_1(r) = 0$ and $a_2(r) = \pi$, we get that in the worst case the drawing of T that consists of G^R_r on the first quadrant and G^L_r on the second quadrant, uses for each one a grid of side-length that is smaller or equal to:

$$(n - 1)\,\frac{\pi}{2}\frac{n - odd(n)}{n - 1}\frac{1}{\pi} = \frac{n - odd(n)}{2}$$

The total width of the grid that Algorithm 3 draws T is the sum of the width of G_r^R and G_r^L. The total height of the grid that Algorithm 3 draws T is the maximum height of G_r^R and G_r^L. Given that a grid of width w and height h is an $(w+1)\times(h+1)$ grid[4], the size of the total grid used by Algorithm 3 is bounded by:

$$\begin{aligned}&\left(2\left(\frac{n-odd(n)}{2}\right)+1\right)\times\left(\frac{n-odd(n)}{2}+1\right)\\ =&\,(n+1-odd(n))\times\left(\frac{n-odd(n)}{2}+1\right)\end{aligned}$$

Therefore, when n is odd the grid size is bounded by $n\times\frac{n+1}{2}$ while, when n is even it is bounded by is $(n+1)\times\left(\frac{n}{2}+1\right)$. ∎

Figures 6-8 present drawings produced by Algorithm 3. Compare Figures 3 and 4 to Figures 6 and 7, respectively, as they depict drawings of the same trees. Figure 6 shows the drawing of a 5-layer complete binary tree (31 vertices). While Theorem 18 indicates that a grid of size 31×16 may be required, the binary tree is drawn on a 23×12 grid. Figure 7 shows the drawing of a path (15 vertices). The drawing matches the bound stated in Theorem 18. Finally, Figure 8 shows a drawing of a non-path tree (out of all 10-vertex rooted trees) that requires maximum area (when produced by Algorithm 3. We have drawn all 10-vertex rooted trees and have identified non-path trees that require a grid of the dimensions stated in Themorem 11.

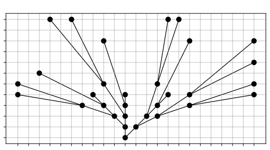

FIGURE 6. *A full binary tree (31 vertices) as drawn by Algorithm 3. Grid size:* 23×12.

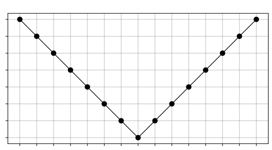

FIGURE 7. *A path (15 vertices) as drawn by Algorithm 3. Grid size:* 15×8.

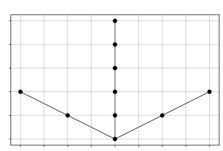

FIGURE 8. *A non-path tree (10 vertices) with maximum required area when drawn by Algorithm 3. Grid size:* 9×6.

5. Four-Quadrants Unrooted Monotone Tree Drawing. In this Section, we provide an algorithm that construct "four-quadrants" drawings of good aspect-ratio for unrooted trees. Algorithm 4, which combines Algorithm 1 and Algorithm 3, yields monotone drawings of n-vertex trees on an $\lfloor\frac{3}{4}(n+2)\rfloor\times\lfloor\frac{3}{4}(n+2)\rfloor$ grid. The main idea of the algorithm is that we first locate a gravity root and partition the subtrees rooted at it into two groups as balanced as possible and, finally, draw the subtrees in each group into two disjoint areas. We emphasize that we consider "non-ordered" trees, i.e., our algorithm will not respect (if given) the embedding of the tree.

LEMMA 19. *Let T be an n-vertex tree rooted at a* gravity root r. *Then, we can identify two subtrees T_1 and T_2 of T of at most $\frac{2n+1}{3}$ vertices each, such that $T_1\cup T_2=T$ and $T_1\cap T_2=r$.*

[4]Recall that we measure length (width/height) in units of distance but, when we denote the dimensions of a grid we use the number of grid points in each dimension.

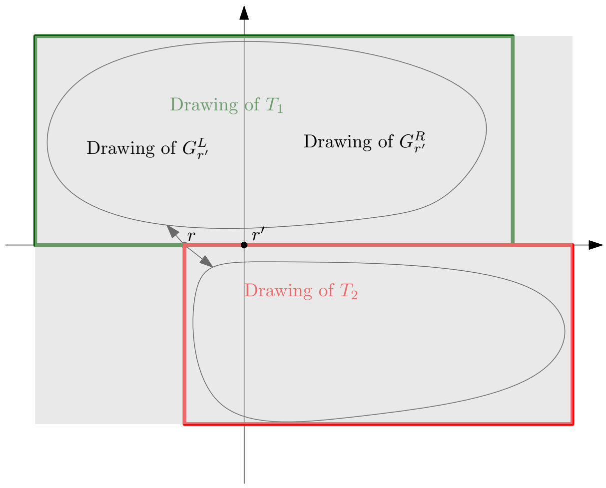

FIGURE 9. *Example of how does Algorithm 4 places T_1 and T_2.*

Proof. Since we must have that $T_1 \cup T_2 = T$ and $T_1 \cap T_2 = r$, it follows that one of the wanted subtrees, say T_1, is formed by r and some of the subtrees rooted at its children, while the other, say T_2, is formed by r and the subtrees rooted at its remaining children. Given that T is rooted at a *gravity root*, the size of each subtree rooted at a child of r is bounded by $\frac{n}{2}$. Let m be the maximum size of a subtree rooted at a child of r, where $m \leq \frac{n}{2}$. We consider cases depending on the value of m.

$\boldsymbol{\frac{n-1}{3} \leq m \leq \frac{n}{2}}$**:** T_1 in formed by r and the subtree of size m that is rooted at a child of r. T_2 is formed by r and the subtrees rooted at the remaining children of r. T_2 is of size $n-m$. Since $\frac{n-1}{3} \leq m \leq \frac{n}{2} \Leftrightarrow \frac{n}{2} \leq n-m \leq \frac{2n+1}{3}$, the size of each subtree is bounded by $\frac{2n+1}{3}$.

$\boldsymbol{m < \frac{n-1}{3}}$**:** In this case, we form T_1 and T_2 as follows: Initially, both T_1 and T_2 consist of the gravity root r. We then consider the subtrees rooted at the children of r in increasing order of their size. At any given step, we insert the currently examined subtree to the smaller of T_1 or T_2 by attaching it to r. At the end of this procedure, the difference in size between the T_1 and T_2 is at most the size of the biggest subtree rooted at a child of r, that is, at most m. Therefore the size of the largest of T_1 and T_2 is bounded by $\frac{n-m}{2} + m < \frac{n+m-1}{2} < \frac{2n+1}{3}$. ∎

Algorithm 4 describes at a high level our *four-quadrant* monotone tree drawing algorithm. Let T be the input tree with gravity root r. Let T_1 and T_2, $|T_1| \geq |T_2|$, be the two subtrees of T according to Lemma 19. We draw the tree in two steps. In the first step, we draw T_1 according to Algorithm 3. In doing so, we take special care to place the path from the gravity root r' of T_1 to r on the X-axis with the appropriate change in the embedding of T_1. In the second step, we draw T_2 according to Algorithm 1. Then, we combine the drawing of T_1 with the reflect on the x axis of the drawing of T_2. The way we combine the two drawings is demonstrated in Figure 9. The drawing produced is monotone and its grid size is bounded by $\lfloor \frac{3}{4}(n+2) \rfloor \times \lfloor \frac{3}{4}(n+2) \rfloor$.
The next strategy explains how to change the embedding of T_1 in order to place the path from the gravity root r' of T_1 to r on the x-axis and to the left of r'.

STRATEGY 2. *In T_1, place each vertex in the path from the gravity root r' of T_1 to r as* *the last child of its parent. In that way, Strategy 1 (which is employed by Algorithm 3) assigns each vertex from r' to r angle-ranges in the from of $\langle \theta_1, \pi \rangle$. Moreover, each edge in the path from the gravity root r' of T_1 to r lies on the X-axis, that is, it is assigned slope π.*

Observe, that when we apply Strategy 2 and draw the tree based on Algorithm 3, Theorem 18 which bounds the drawing area still holds. This is due to the facts that (i) each edge e that connects a vertex u (on the path from r' to r) to its child with slope $slope(e) = \pi$ is drawn at the boundary of the the angle-range of u, and (ii) the length of such an edge e is 1, i.e., the least length possible. Of course, the drawing remains monotone as the following lemma indicates.

Algorithm 4 Four-Quadrants Monotone Tree Drawing algorithm

```
procedure 4QuadrantTreeMonotoneDraw
Input: An n-vertex unrooted tree T.
Output: A four-quadrant monotone drawing of T on a grid of size at most
   ⌊3/4 (n + 2)⌋ × ⌊3/4 (n + 2)⌋.

    Find T1 and T2 according to Lemma 19, where |T1| ≥ |T2|.
    Draw T1 according to Algorithm 3 with the modification of Strategy 2.
    Draw T2 according to Algorithm 1.
    Combine the drawing of T1 with the reflect on the X-axis drawing of T2.
```

Lemma 20. *The drawing produced by Algorithm 4 is planar and monotone.*

Proof. We prove the lemma by showing that the unique simple path that connects two arbitrary vertices u, v of tree T is monotone with respect to some direction. This will imply the monotonicity of the drawing of T and, by Theorem 1, its planarity.
Consider the drawing of an arbitrary tree T produced by Algorithm 4, and let u, v be two arbitrary vertices of T.

Case 1: $u \in T_2$ and $v \in T_2$. By Lemma 9 the drawing of T_2 is monotone. Given that the simple path from u to v is entirely contained in T_2, the path is monotone.

Case 2: $u \in T_1$ and $v \in T_1$. If T_1 was drawn by Algorithm 3 (as it is described in Section 4) then, by Lemma 16, the drawing of T_1 would be monotone. Thus, the simple path from u to v would also be monotone since it is entirely contained in T_1. However, Algorithm 4 additionally applies Strategy 2 when drawing T_1, and thus, we have to ensure that the changes in the drawing due to Strategy 2 do not affect its monotonicity.
Let r and r' be the gravity roots of T and T_1, respectively. The only edges that violate the non-strictly slope-disjoint property of the produced drawing are those that enter nodes on the path from r' to r. Recall that, by Strategy 2 all these edges lie on the X-axis and have slope π. Let $e = (u, v)$ be such an edge in T_1 from vertex u to its child v of slope π.
This is a violation to Property-1 of non-strictly slope disjoint drawings (see Definition 3). Property-1 requires that every edge e from a vertex u to any of its children has a slope that falls within the angle-range of u and does not take the boundary values, that is, $a_1(u) < slope(e) < a_2(u)$. In our case, by Strategy 2 we have that $a_2(u) = \pi$ and $slope(e) = a_2(u) = \pi$. Therefore, for edge $e = (u, v)$ it holds that $a_1(u) < slope(e) \leq a_2(u)$.
We can rotate the whole drawing clockwise around the gravity root r' of T_1

by an arbitrarily small amount $\epsilon > 0$. If we denote by $slope(e)$ the slope of the edge e in the original drawing and by $slope'(e)$ the slope of the edge e in the new rotated drawing, it holds that $slope'(e) = slope(e) - \epsilon$.
For any vertex u with angle-range in the form of $\langle a_1(u), a_2(u)\rangle$, for any edge $e = (u, v)$ that connects u to its child v, since $\epsilon > 0$ is arbitrarily small, it holds that:

$$\begin{aligned}
&a_1(u) < slope(e) \leq a_2(u) \\
\Rightarrow &a_1(u) - \epsilon < slope'(e) \leq a_2(u) - \epsilon \\
\Rightarrow &a_1(u) < slope'(e) < a_2(u)
\end{aligned}$$

So, for the slightly rotated drawing, all the properties of non-strictly slope disjoint drawings are satisfied and, by Theorem 5, the drawing of T_1 is monotone and planar.

if $u \in T_1$ and $v \in T_2$: The simple path from vertex u to vertex v is the concatenation of the simple path from u to the gravity root r of T and of the simple path from r to v. If we consider r as the origin, the edges from u and r lie inside the first two quadrants, with the exception of the edges between r and the gravity root r' of T_1 which lie on the X-axis, while the edges between r and v lie inside the fourth quadrant.
It is easy to observe that the combined path is monotone with respect to a line with slope $\frac{\pi}{2} + \epsilon$ where $\epsilon > 0$ is arbitrarily small. Crucial to this observation is that the two drawing do not overlap. Indeeed, the drawing of T_2, as it is drawn with Algorithm 1, it lies entirely in the fourth quadrant (with respect to r) and non of its vertices lies on the X-axis.

From the three cases above, we conclude that the produced drawing is monotone and planar. This completes the proof. □

THEOREM 21. *Given an n-vertex Tree T, Algorithm 4 draws T in a grid of size at most* $\lfloor \frac{3}{4}(n+2) \rfloor \times \lfloor \frac{3}{4}(n+2) \rfloor$.

Proof. Let r and r' be the vertices used by Algorithm 4 as the gravity roots of T and T_1, respectively. Based on the modification of the drawing of T_1 by Strategy 2, r lies in the second quadrant if we assume r' as the origin node. Furthermore, T_2 is drawn in the fourth quadrant if we assume r as the origin node. From Figure 9, it is clear that the worst case grid size for the combined drawing is realized when r' coincides with r.
By Theorem 18, the grid side-length of subdrawings $\Gamma^R_{r'}$ in the first quadrant and $\Gamma^L_{r'}$ in the second quadrant is bounded by $\frac{|T_1|}{2}$ while in the fourth quadrant, according to Theorem 11, the side-length is at most $|T_2| - 1$.
Therefore, the grid width is $max(|T_1|, \frac{|T_1|}{2} + |T_2| - 1)$ and the grid height is $\frac{|T_1|}{2} + |T_2| - 1$.
We consider two cases depending on whether $\frac{|T_1|}{2} > |T_2| - 1$.

Case 1: $\frac{|T_1|}{2} > |T_2| - 1$. It this case, it is clear that the both the width and the height of the drawing are bounded by $|T_1|$. Given that the gravity root r is included in both T_1 and T_2, we have that:

$$|T_1| + |T_2| = n + 1 \tag{16}$$

From the assumption, we have:

$$
\begin{aligned}
&\frac{|T_1|}{2} > |T_2| - 1 \\
\Rightarrow\ &\frac{|T_1| + |T_2|}{2} > \frac{3}{2}|T_2| - 1 \\
\overset{(16)}{\Rightarrow}\ &\frac{n+1}{2} > \frac{3}{2}|T_2| - 1 \\
\Rightarrow\ &\frac{1}{3}n + 1 > |T_2|
\end{aligned}
$$

Furthermore, we also have that:

$$
\begin{aligned}
|T_2| &= n + 1 - |T_1| \\
&\overset{(Lemma\ 19)}{\geq} n + 1 - \left(\frac{2}{3}n + \frac{1}{3}\right) \\
&\geq \frac{1}{3}n + \frac{2}{3}
\end{aligned}
$$

Therefore, since $\frac{n}{3} + \frac{2}{3} \leq |T_2| < \frac{n}{3} + 1$, the only integer that satisfies this set of inequalities is $|T_2| = \frac{1}{3}n + \frac{2}{3}$. So, $|T_1| = n + 1 - \left(\frac{1}{3}n + \frac{2}{3}\right) = \frac{2}{3}n + \frac{1}{3}$. Thus, the required grid is of size at most:

$$
\left(\frac{2}{3}n + \frac{4}{3}\right) \times \left(\frac{2}{3}n + \frac{4}{3}\right)
$$

This grid, for any $n \geq 1$, fits in a grid of dimensions:

$$
\left(\frac{3}{4}(n+2)\right) \times \left(\frac{3}{4}(n+2)\right)
$$

Therefore, the statement holds.

Case 2: $\frac{|T_1|}{2} \leq |T_2| - 1$. In this case, the grid side-length is:

$$
\begin{aligned}
& \frac{|T_1|}{2} + |T_2| - 1 \\
= \ & \frac{|T_1| + |T_2|}{2} + \frac{|T_2|}{2} - 1 \\
\overset{(16)}{=} \ & \frac{n+1}{2} + \frac{|T_2|}{2} - 1 \\
\overset{|T_2| \leq |T_1|}{\leq} \ & \frac{n+1}{2} + \frac{n+1}{4} - 1 \\
= \ & \frac{3n-1}{4}
\end{aligned}
$$

Therefore, in this case the required grid is of size

$$
\left(\frac{3}{4}n + \frac{3}{4}\right) \times \left(\frac{3}{4}n + \frac{3}{4}\right)
$$

which, obviously, fits in a grid of size

$$
\left(\frac{3}{4}(n+2)\right) \times \left(\frac{3}{4}(n+2)\right)
$$

Since the bound for the grid size must be integer, the floor of the bound also bounds the grid size. Therefore, as stated in the lemma, the required grid is of size:

$$\left\lfloor \frac{3}{4}(n+2) \right\rfloor \times \left\lfloor \frac{3}{4}(n+2) \right\rfloor$$

$\square$

Figures 10-12 present drawings produced by Algorithm 4. In the drawings, we indicate by a solid square (rhombus) the gravity root of tree T (resp., T_1). Figure 10 shows a drawing of aspect-ratio equal to one for a 5-layer complete binary tree (31 vertices). While Theorem 21 indicates that a grid of size 24×24 may be required, the binary tree is drawn on a 17×17 grid. Figure 11 shows the drawing of a path (15 vertices). The drawing matches the bound stated in Theorem 18. Finally, Figure 8 shows a drawing of a non-path tree (out of all 10-vertex rooted trees) that requires maximum area (when produced by Algorithm 3. We have drawn all 10-vertex rooted trees and have identified non-path trees that require the maximum area. While Themorem 21 indicates that an 9×9 grid may be used for a tree of 10 vertices, the drawing of maximum area uses a grid of size 8×7.

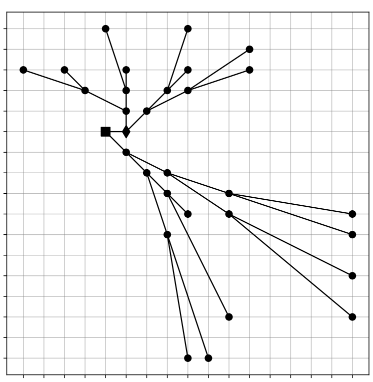

FIGURE 10. *A full binary tree (31 vertices) as drawn by Algorithm 3. Grid size:* 17×17.

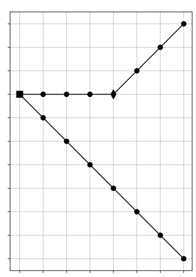

FIGURE 11. *A path (15 vertices) as drawn by Algorithm 3. Grid size:* 8×11.

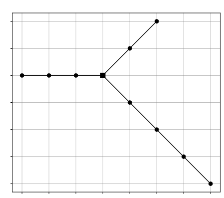

FIGURE 12. *A non-path tree (10 vertices) with maximum required area when drawn by Algorithm 3. Grid size:* 8×7.

6. Conclusion and Open Problems. We have described three algorithms that produce monotone drawings of trees. The algorithm that has the best aspect ratio produces a monotone drawing of an n-vertex tree on a grid of size at most $\lfloor \frac{3}{4}(n+2) \rfloor \times \lfloor \frac{3}{4}(n+2) \rfloor$. The following problems on monotone tree drawings are worth studying:

1. He and He [6] described a tree that requires for its monotone drawing a grid of size at least $\frac{n}{9} \times \frac{n}{9}$. Can this bound be improved? Is there a tree that requires a larger grid for its monotone drawing?
2. The angular resolution of the produced drawing has not been studied. Is there a trade-off between the angular resolution and the grid size of the monotone drawing?

REFERENCES

[1] P. Angelini, E. Colasante, G. Di Battista, F. Frati, and M. Patrignani, *Monotone drawings of graphs*, Journal of Graph Algorithms and Applications, 16 (2012), pp. 5–35, https://doi.org/10.7155/jgaa.00249.

[2] P. Angelini, W. Didimo, S. Kobourov, T. Mchedlidze, V. Roselli, A. Symvonis, and S. Wismath, *Monotone drawings of graphs with fixed embedding*, Algorithmica, 71 (2015), pp. 233–257, https://doi.org/10.1007/s00453-013-9790-3.

[3] A. Brocot, *Calcul des rouages par approximation, nouvelle methode*, Revue Chronometrique, 6 (1860), pp. 186–194.

[4] R. L. Graham, D. E. Knuth, and O. Patashnik, *Concrete Mathematics: A Foundation for Computer Science*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd ed., 1994.

[5] D. He and X. He, *Nearly optimal monotone drawing of trees*, Theoretical Computer Science, 654 (2016), pp. 26–32, https://doi.org/10.1016/j.tcs.2016.01.009.

[6] D. He and X. He, *Optimal monotone drawings of trees*, SIAM J. Discrete Math., 31 (2017), pp. 1867–1877, https://doi.org/10.1137/16M1080045, https://doi.org/10.1137/16M1080045.

[7] X. He and D. He, *Compact monotone drawing of trees*, in Computing and Combinatorics - 21st International Conference, COCOON 2015, Beijing, China, August 4-6, 2015, Proceedings, 2015, pp. 457–468, https://doi.org/10.1007/978-3-319-21398-9_36.

[8] M. I. Hossain and M. S. Rahman, *Good spanning trees in graph drawing*, Theoretical Computer Science, 607 (2015), pp. 149–165, https://doi.org/10.1016/j.tcs.2015.09.004.

[9] P. Kindermann, A. Schulz, J. Spoerhase, and A. Wolff, *On monotone drawings of trees*, in Graph Drawing - 22nd International Symposium, GD 2014, Würzburg, Germany, September 24-26, 2014, Revised Selected Papers, 2014, pp. 488–500, https://doi.org/10.1007/978-3-662-45803-7_41.

[10] A. Oikonomou and A. Symvonis, *Simple compact monotone tree drawings*, in Graph Drawing and Network Visualization - 25th International Symposium, GD 2017, Boston, MA, USA, September 25-27, 2017, Revised Selected Papers, F. Frati and K. Ma, eds., vol. 10692 of Lecture Notes in Computer Science, Springer, 2017, pp. 326–333, https://doi.org/10.1007/978-3-319-73915-1_26.

[11] M. Stern, *Ueber eine zahlentheoretische funktion*, Journal fur die reine und angewandte Mathematik, 55 (1858), pp. 193–220.