On Monotone Drawings of Trees*

Philipp Kindermann[1], André Schulz[2], Joachim Spoerhase[1], and Alexander Wolff[1]

[1] Lehrstuhl für Informatik I, Universität Würzburg, Germany
http://www1.informatik.uni-wuerzburg.de/en/staff
[2] Institut für Mathematische Logik und Grundlagenforschung, Universität Münster, Germany
andre.schulz@uni-muenster.de

Abstract. A crossing-free straight-line drawing of a graph is *monotone* if there is a monotone path between any pair of vertices with respect to *some* direction. We show how to construct a monotone drawing of a tree with n vertices on an $O(n^{1.5}) \times O(n^{1.5})$ grid whose angles are close to the best possible angular resolution. Our drawings are *convex*, that is, if every edge to a leaf is substituted by a ray, the (unbounded) faces form convex regions. It is known that convex drawings are monotone and, in the case of trees, also crossing-free.

A monotone drawing is *strongly monotone* if, for every pair of vertices, the direction that witnesses the monotonicity comes from the vector that connects the two vertices. We show that every tree admits a strongly monotone drawing. For biconnected outerplanar graphs, this is easy to see. On the other hand, we present a simply-connected graph that does not have a strongly monotone drawing in any embedding.

1 Introduction

A natural requirement for the layout of a connected graph is that between any source vertex and any target vertex, there should be a source–target path that approaches the target according to some distance measure. A large body of literature deals with problems of this type; various measures have been studied. For example, in a *greedy drawing* it is possible to decide locally where to go, by selecting in the current vertex any neighbor closer to the target. In a *monotone* drawing, the distance between vertices (on the desired source-target path) is measured with respect to their projections on *some* line, which may be different for any source–target pair. In *strongly monotone* drawings, that line is always the line from source to target, and in *upward* drawings, the line is always the vertical line, directed upwards.

In this paper, we focus on monotone and strongly monotone drawings of trees with additional aesthetic properties such as convexity or small area. Given a tree, we call the edges incident to the leaves *leaf edges* and all other edges *interior edges*. We direct all edges away from the root. Given a straight-line drawing of a tree, we substitute each leaf edge by a ray whose initial part coincides with the edge. The embedding of the tree defines a combinatorial embedding of the tree, that is the order of the edges around every vertex. The faces are then specified by this combinatorial embedding as leaf-to-leaf paths. If the faces of the augmented drawing are realized as convex nonoverlapping

* This research was supported by the ESF EuroGIGA project GraDR (DFG grant Wo 758/5-1).

C. Duncan and A. Symvonis (Eds.): GD 2014, LNCS 8871, pp. 488–500, 2014.

(unbounded) polygonal regions, then we call the original drawing a *convex drawing*. If every region is *strictly convex* (that is, all interior angles are strictly less than π), we also call the drawing *strictly convex*. Note that a convex drawing is also monotone [4,2], but a monotone drawing is not necessarily convex. Strict convexity forbids vertices of degree 2. In this paper, when we talk about (strongly) monotone drawings, this always includes the planarity requirement. Otherwise, as Angelini et al. [2] observed, drawing any spanning tree of the given graph in a (strongly) monotone way and inserting the remaining edges would yield a (strongly) monotone drawing.

Previous Work. While any 3-connected plane graph has a greedy drawing in the Euclidean plane [10] (even without crossing [7]), this is, unfortunately, not true for trees. Nöllenburg and Prutkin [11] gave a complete characterization for the tree case, which shows that no tree with a vertex of degree 6 or more admits a greedy drawing. Alamdari et al. [1] have studied a subclass of greedy drawings, so-called *self-approaching drawings* which require that there always is a source–target path such that the distance decreases for any triplet of intermediate points on the *edges*, not only for the vertices.

Carlson and Eppstein [6] study convex drawings of trees. They give linear-time algorithms that optimize the angular resolution of the drawings, both for the fixed- and the variable-embedding case. They observe that convexity allows them to pick edge lengths arbitrarily, without introducing crossings.

For monotone drawings, Angelini et al. [2] studied the variable-embedding case. They showed that any n-vertex tree admits a straight-line monotone drawing on a grid of size $O(n^{1.6}) \times O(n^{1.6})$ (using a BFS-based algorithm) or $O(n) \times O(n^2)$ (using a DFS-based algorithm). They also showed that any biconnected planar graph has a monotone drawing (using exponential area). Further, they observed that not every planar graph admits a monotone drawing if its embedding is fixed. They introduced the concept of *strong monotonicity* and showed that there is a drawing of a planar triangulation that is not strongly monotone. Hossain and Rahman [9] improve some of the results of Angelini et al. by showing that every connected planar graph admits a monotone drawing of size $O(n) \times O(n^2)$ and that such a drawing can be computed in linear time.

Both the BFS- and the DFS-based algorithms of Angelini et al. precompute a set of $n-1$ vectors in decreasing order of slope. For this, they use two different partial traversals of the so-called Stern–Brocot tree, an infinite tree whose vertices are in bijection with the irreducible positive rational numbers. Such numbers can be seen as *primitive* vectors in 2d, that is, as vectors with pairwise different slopes. Then both algorithms do a depth-first (pre-order) traversal of the input tree. Whenever they hit a new edge, they assign to it the steepest unused vector. They place the root of the input tree at the origin and draw each edge (u, v) by adding the vector assigned to (u, v) to the position of u. They call such tree drawings *slope-disjoint*. We won't formally define this notion here, but it is not hard to see that it implies monotonicity.

Angelini, with a different set of co-authors [3], investigated the fixed-embedding case. They showed that, on the $O(n)\times O(n^2)$ grid, every connected plane graph admits a monotone drawing with two bends per edge and any outerplane graph admits a straight-line monotone drawing.

Our contribution. We present two main results. First, we show that any n-vertex tree admits a strictly convex and, hence, monotone drawing on the $O(n^{1.5}) \times O(n^{1.5})$ grid (see Section 3). As the drawings of Angelini et al. [2], our drawings are slope-disjoint, but we use a different set of primitive vectors (based on Farey sequences), which slightly decreases the grid size. (This also works for the BFS-based algorithm of Angelini et al.) Instead of pre-oder, we use a kind of in-oder traversal (first child – root – other children) of the input tree, which helps us to achieve convexity. Our ideas can be applied to modify the optimal angular resolution algorithm of Carlson and Eppstein [6] such that a drawing on an $O(n^{1.5}) \times O(n^{1.5})$ grid is constructed at the expense of missing the optimal angular resolution by a constant factor. Second, we show that any tree admits a *strongly* monotone drawing (see Section 4). So far, no positive results have been known for strongly monotone drawings.

In the case of proper binary trees, our drawings are additionally strictly convex. For biconnected outerplanar graphs, it is easy to construct strongly monotone drawings. On the other hand, we present a simply-connected planar graph that does not have a strongly monotone drawing in any embedding.

We leave it as an open question whether trees admit strongly monotone drawings on a grid of polynomial size (see Section 5).

2 Building Blocks: Primitive Vectors

The following algorithms require a set of integral vectors with distinct directed slopes and bounded length. In particular, we ask for a set of *primitive vectors* $P_d = \{(x, y) \mid \gcd(x, y) \in \{1, d\}, 0 \le x \le y \le d\}$. Our goal is to find the right value of d such that P_d contains at least k primitive vectors, where k is a number that we determine later. We can then use the reflections on the lines $x = y$, $y = 0$ and $x = 0$ to get a sufficiently large set of integer vectors with distinct directed slopes. The edges of the monotone drawings in Section 3 are translates of these vectors; each edge uses a different vector.

Assume that we have fixed d and want to generate the set P_d. If we consider each entry (x, y) of P_d to be a rational number x/y and order these numbers by value, we get the *Farey sequence* $\mathcal{F}_d$ (see, for example, Hardy and Wright's book [8]). The Farey sequence is well understood. In particular, it is known that $|\mathcal{F}_d| = 3d^2/\pi^2 + O(d \log d)$ [8, Thm. 331]. Furthermore, the entries of $\mathcal{F}_d$ can be computed in time $O(|\mathcal{F}_d|)$. We remark that the set $\bigcup_d \mathcal{F}_d$ coincides with the entries of the Stern–Brocot tree. However, collecting the latter level by level is not the most effective method to build a set of primitive vectors for our purpose.

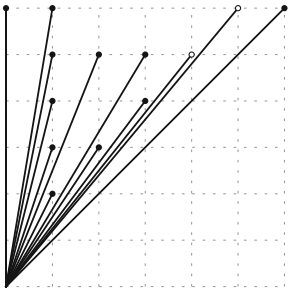


Fig. 1. The 13 primitive vectors obtained from $\mathcal{F}_6$. The smallest angle of $\approx 1.14^\circ$ is realized between the vectors $(4, 5)$ and $(5, 6)$ marked with white dots; the best possible angular resolution in this case is $45^\circ/12 = 3.75^\circ$.

To get access to a set of k primitive vectors, we use the first k entries of the Farey sequence $\mathcal{F}_d$, for $d := 4\lceil\sqrt{k}\rceil$, replacing each rational by its corresponding 2d vector. By selecting k vectors form this set we get a set of exactly k primitive vectors, which we denote by V_k; see Fig. 1.

If we wish to have more control over the aspect ratio in our final drawing, we can pick a set of primitive vectors contained inside a triangle spanned by the grid points $(0,0)$, $(m_x,0)$, (m_x,m_y). By stretching the triangle and keeping its area fixed, we may end up with fewer primitive vectors. This will result in an (only slightly) smaller constant compared to the case $m_x = m_y$. As proven by Bárány and Rote [5, Thm. 2], any such triangular domain contains at least $m_x m_y/4$ primitive vectors. This implies that we can adapt the algorithm easily to control the aspect ratio by selecting the box for the primitive vectors accordingly. For the sake of simplicity, we detail our algorithms only for the most interesting case ($m_x = m_y$).

Lemma 1. *Let $P \subseteq P_d$ be a set of $k = |P_d|/c$ primitive vectors with no coordinate greater than d for some constant $c \geq 1$. Then any two primitive vectors of P are separated by an angle of $\Omega(1/k)$.*

Proof. Since $|P_d| = 3d^2/\pi^2 + O(d \log d)$ we have that $2d^2 \approx 2\pi^2 ck/3$. Any line with slope m encloses an angle α with the x-axis, such that $\tan(\alpha) = m$. Let m_1 and m_2 be the slopes of two lines and let α_1 and α_2 be the corresponding angles with respect to the x-axis. By the trigonometric addition formulas we have that the separating angle of these two lines equals

$$\tan\phi := \tan(\alpha_1 - \alpha_2) = \frac{\tan\alpha_1 - \tan\alpha_2}{1 + \tan\alpha_1 \tan\alpha_2} = \frac{m_1 - m_2}{1 + m_1 m_2}.$$

For any two neighboring entries p/q and r/s in the Farey sequence, it holds that $qr - ps = 1$ [8, Thm. 3.1.2], and therefore p/q and r/s differ by exactly $(qr - ps)/(qs) = 1/(qs)$. As a consequence, $\tan\phi = 1/(pr + qs)$. The angle ϕ is minimized if $pr + qs$ is maximized. Clearly, we have that $pr + qs < 2d^2 \approx 2\pi^2 ck/3$. By the Taylor expansion, $\arctan(x) = x - x^2\xi/(1+\xi^2)^2$ for some value $0 \leq \xi \leq x$. Substituting x with $3/(2\pi^2 ck)$ yields, for $k \geq 2$, that

$$\phi \geq \frac{3}{2\pi^2 ck} - \frac{9\xi}{4\pi^4 c^2 k^2 (1+\xi^2)^2} > \frac{3}{2\pi^2 ck} - \frac{9}{4\pi^4 c^2 k^2} = \Omega(1/k).$$

□

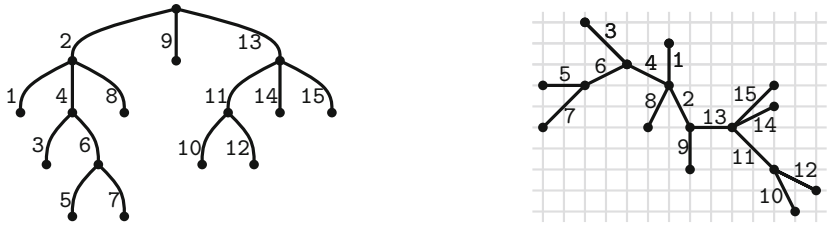
Since the best possible resolution for a set of k primitive vectors is $2\pi/k$, Lemma 1 shows that the resolution of our set differs from the optimum by at most a constant.

3 Monotone Grid Drawings with Good Angles

We start by ensuring that convex tree drawings are crossing-free. This has already been stated by Carlson and Eppstein [6].

Lemma 2. *Any convex straight-line drawing of a tree is crossing-free.*

We now present a simple method for drawing a tree on a grid in a strictly convex, and therefore monotone and, by Lemma 2, crossing-free way. We name our strategy the *inorder-algorithm*. The algorithm first computes a reasonable large set of primitiv vectors, then selects a subset of these vectors and finally assigns the slopes to the edges.



(a) tree with edge numbers $s(\cdot)$ (b) grid drawing of the tree

Fig. 2. A strictly convex drawing of a tree

The drawing is then generated by translating the selected primitive vectors. In the following, an *extended* subtree will refer to a subtree including the edge leading into the subtree (if the subtree is not the whole tree).

Every edge e will be assign with a number $s(e)$. This number will refer to the rank of the edge's slope (in circular order) in the final assignment. The rank assignment is done in a recursive fashion. At any time, let $\hat{s}$ be 1 plus the maximum rank $s(e)$ assigned so far. Initially, $\hat{s} = 1$. Let $e = uv$ be an edge (directed away from the root), and let $v_1, v_2, \ldots, v_\ell$ be the children of v ordered from left to right. We recursively set the ranks of all edges in the extended subtree rooted at v_1. Then we set $s(e) = \hat{s}$ (which increases $\hat{s}$ by one). Finally, we set, for $i = 2, \ldots, \ell$, the ranks of the edges in the extended subtree rooted at v_i. For an example of a tree with its edge ranks, see Fig. 2a.

Second, we assign actual slopes to the edges. Let e be an edge with $s(e) = j$. Then we assign to e some vector $s_j \in \mathbb{Z}^2$ and draw e as a translate of s_j. We pick the vectors $s_1, s_2, \ldots, s_{n-1}$ by selecting a sufficiently large set of primitive vectors and their reflections in counterclockwise order, see Section 2. Our drawing algorithm has the following requirements, which can be fulfilled as the following lemma shows:

(R1) Edges that are incident to the root and consecutive in circular order are assigned to vectors that together span an angle less than π.
(R2) In every extended subtree hanging off the root, the edges are assigned to a set of vectors that spans an angle less than π.

Lemma 3. *We can select $n-1$ vectors with distinct directed slopes from a $[-d, d] \times [-d, d]$ grid with $d = 4\lceil\sqrt{n}\rceil$ such that the requirements (R1) and (R2) are fulfilled.*

Proof. We first preprocess our tree by adding temporary edges at some leaves. These edges will receive slopes, but are immediately discarded after the assignment.

First, our objective is to ensure that the tree can be split up into three parts that all have n edges. In particular, we adjust the sizes of the extended subtrees hanging off the root by adding temporary edges such that we can partition them into three sets of consecutive extended subtrees which all contain n edges. Note that we have to add $2n+1$ edges to achieve this.

Second, we define three cones C_1, C_2, and C_3 (see Fig. 3). Each cone has its apex at the origin and spans an angle of $\pi/4$. The angular ranges are $C_1 = [0, \pi/4]$, $C_2 = [3\pi/4, \pi]$, and $C_3 = [3\pi/2, 7\pi/4]$; angles are measured from the x-axis pointing in positive direction. Note that C_2 is separated from the two other cones by an angle of $\pi/2$. As mentioned in Section 2 the set V_n contains n primitive vectors in the $[0, d] \times [0, d]$ grid. When reflected on the $x = y$ line these vectors lie in C_1. Reflecting the vectors in C_1 further we generate n vectors in C_2 and n vectors in C_3. In every cone we "need" at most $n - 3$ edges, hence we can remove the vectors on the boundary of each cone. After removing the temporary edges, the number of vectors will drop from $3n$ to $n - 1$.

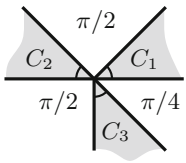


Fig. 3. The cones that contain the slopes used in the algorithm

Now, we observe the following. Every two consecutive edges incident to the root lie in the *interiors* of our cones. This yields requirement (R1) given the sizes and angular distances of the cones. Furthermore, any extended subtree is assigned slopes from a single cone. This yields (R2). □

For the example tree of Fig. 2a, it suffices to pick the 16 vectors that one gets from reflecting the primitive vectors from the $[0, 2] \times [0, 2]$ grid. These vectors already fulfill requirements (R1) and (R2). Hence, we did not have to apply the more involved slope selection as described in Lemma 3. The resulting drawing is shown in Fig. 2b.

Every face in the drawing contains two leaves. The leaves are ordered by their appearance in some DFS-sequence $\mathcal{D}$ respecting some rooted combinatorial embedding of T. For a face f, we call the leaf that comes first in $\mathcal{D}$ the *left leaf* and the other leaf of f the *right leaf* of f. The only exception is the face whose leaves are the first and last child of $\mathcal{D}$. Here we call the first vertex in $\mathcal{D}$ the right leaf and the last vertex in $\mathcal{D}$ the left leaf.

Lemma 4. *Let u be the left leaf, and let v be the right leaf of a face of T. Further, let w be the lowest common ancestor of u and v. The above assignment of slope ranks s to the tree edges implies the following.*
(a) If edge e_1 is on the w–u path and edge e_2 is on the w–v path, then $s(e_1) < s(e_2)$.
(b) The ordered sequence of edges on the path $w \to u$ is increasing in $s(\cdot)$.
(c) The ordered sequence of edges on the path $w \to v$ is decreasing in $s(\cdot)$.

The proof is omitted because of space constraints. We now prove the correctness of our algorithm.

Theorem 1. *Given an embedded tree with n vertices (none of degree 2), the inorder-algorithm produces a strictly convex and crossing-free drawing with angular resolution $\Omega(1/n)$ on a grid of size $O(n^{1.5}) \times O(n^{1.5})$. The algorithm runs in $O(n)$ time.*

Proof. We first show that in the drawing no face is incident to an angle larger than π. Let f be a face, let e and e' be two consecutive edges on the boundary of f, and let α be the angle formed by e and e' in the interior of f. If e and e' are incident to the root, requirement (R1) implies $\alpha < \pi$. If both edges contain the lowest common ancestor of the leafs belonging to f, then by requirement (R2) also $\alpha < \pi$. In the remaining case, e and e' both lie on a path to the left leaf of f, or both lie on a path to the right

leaf of f. At vertex v we have at least two outgoing edges. Let e_1 be the first outgoing edge and e_2 be the last outgoing edge at v – one of the edges is e'. By the selection of the slope ranks we have $s(e_1) < s(e) < s(e_2)$. As a consequence, the supporting line of e separates e_1 and e_2, and hence both faces containing e have an angle less than π at v and therefore $\alpha < \pi$.

Next, we show that the edges and rays of a face do not intersect. Then, by Lemma 2, no edges will cross. Assume that there are two edges/rays ℓ and r in a common face that have an intersection in some point x. Let t be the lowest common ancestor of ℓ and r and assume that ℓ lies on the path to the left leaf and r on the path to the right leaf. We define a closed polygonal chain P of as follows. The chain starts with the path from t to ℓ, continues via x to r, and finally returns to t. We direct the edges according to this walk (for measuring the directed slopes) and call them $e_1, e_2, \ldots, e_k$. We may assume that P is simple, otherwise we find another intersection point. By Lemma 4, the slopes are monotone when we traverse P. For $i = 1, \ldots, k-1$, let α_i be the difference between the directed slopes of the edges e_i and e_{i+1}. Then the sum $\sum_{i<k} \alpha_i$ equals the angle between the slopes of e_1 and e_k. Due to requirement (R2), this angle is less than π. Let $\beta_i = \pi - \alpha_i$ be the angle between e_i and e_{i+1} in P, and let $\beta_0 > 0$ be the "interior" angle at t. We have that

$$\sum_{0 \le i < k} \beta_i = \beta_0 + \sum_{1 \le i < k} (\pi - \alpha_i) > 0 + (k-1)\pi - \pi = (k-2)\pi.$$

This, however, contradicts the fact that the angle sum of the polygon with boundary P is $(k-2)\pi$. Thus, our assumption that two edges/rays cross was wrong.

Since the drawing is assembled from $n-1$ vectors whose absolute coordinates are at most $O(\sqrt{n})$, the complete drawing uses a grid of dimension $O(n^{1.5}) \times O(n^{1.5})$. Since all vectors are reflections of (a subset of) vectors defined by a Farey sequence with at most n entries, Lemma 1 yields that the angular resolution is bounded by $\Omega(1/n)$. □

We conclude this section with comparing our result with the drawing algorithm of Carlson and Eppstein [6]. Their algorithm produces a drawing with optimal angular resolution. It draws trees convex, but, in contrast to our algorithm, not necessarily strictly convex. Allowing parallel leaf edges can have a great impact on the angular resolution. However, our ideas can be applied to modify the algorithm of Carlson and Eppstein. For the leaf edges, their algorithm uses a set of k slopes and picks the slopes such that they are separated by an angle of $2\pi/k$. The slopes of interior edges have either one of the slopes of the leaf edges, or are chosen such that they bisect the wedge spanned by their outermost child edges. However, it suffices to assure that the slope of an interior edge differs from the extreme slopes in the following subtree by at least $2\pi/(2k)$.

We can now modify the algorithm as follows. We pick $2k/8$ primitive vectors and reflect them such that they fill the whole angular space with $2k$ distinct integral vectors. We use every other vector of this set for the leaf edges. For an interior edge we take any vector from our preselected set whose slope lies in between the extreme slopes of the edges in its subtree. We can always find such a vector, since we have sufficiently spaced out our set of primitive vectors. By this we obtain a drawing on the $O(n^{1.5}) \times O(n^{1.5})$ grid. Clearly, the drawing doesn't have optimal angular resolution. However, since we

use $2k$ integral vectors having, by Lemma 1, an angular resolution of $\Omega(1/k)$, we differ from the best possible angular resolution $2\pi/k$ only by a constant factor.

4 Strongly Monotone Drawings

We first show how to draw any proper binary tree (that is, any internal vertex has exactly two children). We name our strategy the *disk-algorithm*. Then, we generalize our result to arbitrary trees. Further, we show that connected planar graphs do not necessarily have a strongly monotone drawing. Finally, we show how to draw biconnected outerplanar graphs in a strongly monotone fashion.

Let T be a proper binary tree, let D be any disk with center c, and let C be the boundary of D. Recall that a strictly convex drawing cannot have a vertex of degree 2. Thus, we consider the root of T a dummy vertex and ensure that the angle at the root is π. We draw T inside D. We start by mapping the root of T to c. Then, we draw a horizontal line h through c and place the children of the root on $h \cap \text{int}(D)$ such that they lie on opposite sites relative to c. We cut off two circular segments by dissecting D with two vertical lines running through points representing the children of the root. We inductively draw the right subtree of T into the right circular segment and the left subtree into the left circular segment.

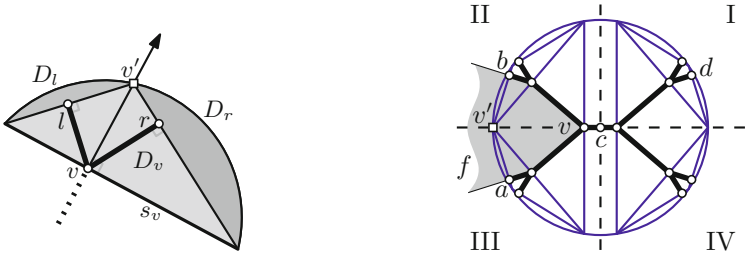
In any step of the inductive process, we are given a vertex v of T, its position in D (which we also denote by v) and a circular segment D_v; see Fig. 4a. The preconditions for our construction are that

(i) v lies in the relative interior of the chord s_v that delimits D_v, and
(ii) D_v is empty, that is, the interiors of D_v and D_u are disjoint for any vertex u that does not lie on a root–leaf path through v.

In order to place the two children l and r of v (if any), we shoot a ray $\boldsymbol{v}$ from v perpendicular to s_v into D_v. Let v' be the point where $\boldsymbol{v}$ hits C. Consider the chords that connect the endpoints of s_v to v'. The chords and s_v form a triangle with height vv'. The height is contained in the interior of the triangle and splits it into two right subtriangles. The chords are the hypotenuses of the subtriangles. We contruct l and r by connecting v to these chords perpendicularly. Note that, since the subtriangles are right triangles, the heights lie inside the subtriangles. Hence, l and r lie in the relative interiors of the chords. Further, note that the circular segments D_l and D_r delimited by the two chords are disjoint and both are contained in D_v. Hence, D_l and D_r are empty, and the preconditions for applying the above inductive process to r and l with D_l and D_r are fulfilled. See Fig. 4b for the output of our algorithm for a tree of height 3.

Lemma 5. *For a proper binary tree rooted in a dummy vertex, the disk-algorithm yields a strictly convex drawing.*

Proof. Let T be a proper binary tree and let f be a face of the drawing generated by the algorithm described above. Clearly, f is unbounded. Let a and b be the leaves of T that are incident to the two unbounded edges of f, and let v be the lowest common ancestor of a and b; see Fig. 4b. Consider the two paths from v to a and b. We assume that the path from v through its left child ends in a and the path through its right child ends in b.



(a) sketch of the inductive step (b) output of our algorithm for a tree of height 3

Fig. 4. Strongly monotone drawings of proper binary trees

Due to our inductive construction that uses disjoint disk sections for different subtrees, it is clear that the two paths do not intersect. Moreover, each vertex on the two paths is convex, that is, the angle that such a vertex forms inside f is less than π. This is due to the fact that we always turn right when we go from v to a, and we always turn left when we go to b. Vertex v is also convex since the two edges from v to its children lie in the same half-plane (bounded by s_v).

It remains to show that the two rays $\boldsymbol{a}$ and $\boldsymbol{b}$ (defined analogously to $\boldsymbol{v}$ above) don't intersect. To this end, recall that $v' = \boldsymbol{v} \cap C$. By our construction, $\boldsymbol{a}$ and $\boldsymbol{b}$ are orthogonal to two chords of C that are both incident to v'. Clearly, the two chords form an angle of less than π in v'. Hence, the two rays diverge, and the face f is strictly convex. □

For the proof that the algorithm described above yields a strongly monotone drawing, we need the following tools. Let $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ be two vectors. We say that $\boldsymbol{v}_3$ *lies between* $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ if $\boldsymbol{v}_3$ is a positive linear combination of $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. For two vectors $\boldsymbol{v}$ and $\boldsymbol{w}$, we define $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = |\boldsymbol{v}||\boldsymbol{w}| \cos(\boldsymbol{v}, \boldsymbol{w})$ as the scalar product of $\boldsymbol{v}$ and $\boldsymbol{w}$.

Lemma 6. *If a path p is monotone with respect to two vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, then it is monotone with respect to any vector $\boldsymbol{v}_3$ between $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$.*

Proof. Let $\boldsymbol{v}_3 = \lambda_1 \boldsymbol{v}_1 + \lambda_2 \boldsymbol{v}_2$ with $\lambda_1, \lambda_2 > 0$. Assume that the path p is given by the sequence of vectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_k$. Since p is monotone with respect to vectors $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$, we have that $\langle \boldsymbol{v}_1, \boldsymbol{w}_i \rangle > 0$ and $\langle \boldsymbol{v}_2, \boldsymbol{w}_i \rangle > 0$ for all $i \leq k$. This yields, for all $i \leq k$,

$$\langle \boldsymbol{v}_3, \boldsymbol{w}_i \rangle = \langle \lambda_1 \boldsymbol{v}_1 + \lambda_2 \boldsymbol{v}_2, \boldsymbol{w}_i \rangle = \lambda_1 \langle \boldsymbol{v}_1, \boldsymbol{w}_i \rangle + \lambda_2 \langle \boldsymbol{v}_2, \boldsymbol{w}_i \rangle > 0,$$

since $\lambda_1, \lambda_2 > 0$. It follows that p is monotone with respect to $\boldsymbol{v}_3$. □

Lemma 7. *For a proper binary tree rooted in a dummy vertex, the disk-algorithm yields a strongly monotone drawing.*

Proof. We split the drawing into four sectors: I, II, III and IV; see Fig. 4b. Let a and b be two vertices in the graph. We will show that the path that connects a and b in the drawing output by our algorithm is strongly monotone. Let c be the root of the tree. W.l.o.g., assume that a lies in sector III. Then, we distinguish three cases.

Case 1: a and b lie on a common root–leaf path; see a and v in Fig. 4b. Obviously, b lies in sector III. W.l.o.g., assume that b lies on a path between a and c. By construction, all edges in sector III, seen as vectors directed towards c, lie between $\boldsymbol{x} = (0, 1)$ and $\boldsymbol{y} = (1, 0)$. Thus, all edges on the path from a to b, and in particular $\boldsymbol{ab}$, lie between $\boldsymbol{x}$ and $\boldsymbol{y}$. Since $\boldsymbol{x}$ is perpendicular to $\boldsymbol{y}$, the path from a to b is monotone with respect to $\boldsymbol{x}$ and $\boldsymbol{y}$. Following Lemma 6, the path between a and b is monotone with respect to $\overrightarrow{ab}$, and thus strongly monotone.

Case 2: b lies in sector I; see a and d in Fig. 4b. In Case 1, we have shown that the all edges on the path from a to c lie between $\boldsymbol{x} = (0, 1)$ and $\boldsymbol{y} = (1, 0)$. Analogously, the same holds for the path from c to b. Thus, the path between a and b is monotone with respect to $\boldsymbol{x}$ and $\boldsymbol{y}$ and, following Lemma 6, strongly monotone.

Case 3: a and b do not lie on a common root–leaf path, and b does not lie in sector I; see a and b in Fig. 4b. Let d be the lowest common ancestor of a and b. Let $a_0, a_1, \ldots, a_{k-1}, a_k$ be the path from d to a where $a_0 = d$ and $a_k = a$. Now, let $b_0, b_1, \ldots, b_{m-1}, b_m$ be the path from d to b where $d = b_0$ and $b = b_m$. Finally, let $p = a_k, a_{k-1}, \ldots, a_0, b_1, \ldots, b_{m-1}, b_m$ be the path from a to b.

Below, we describe how to rotate and mirror the drawing such that the any vector $\overrightarrow{a_i, a_{i-1}}$, $1 \leq i \leq k$ lies between $\boldsymbol{x} = (0, 1)$ and $\boldsymbol{y} = (1, 0)$, and any vector $\overrightarrow{b_{j-1}, b_j}$, $1 \leq j \leq m$ lies between $\boldsymbol{x}$ and $-\boldsymbol{y}$. This statement is equivalent to $x(a_i) < x(d) < x(b_j), 1 \leq i \leq k, 1 \leq j \leq m$ and $y(a_k) < \ldots < y(a_1) < y(d) > y(b_1) > \ldots > y(b_m)$; see Fig. 5. If b lies in sector IV, then $d = c$ and this statement is true by construction. If b lies in sector II, then d is a child of c. We rotate the drawing by $\pi/2$ in counterclockwise direction and then mirror it horizontally. If b lies in sector III, let $p(d)$ be the parent of d. We rotate the drawing such that the edge $(p(d), d)$ is drawn vertically. Recall that, by construction, the ray from d in direction $\overrightarrow{p(d)d} = -\boldsymbol{y}$ separates the subtrees of the two children of d; see Fig. 4a. Further, the angle between any edge (directed away from d) in the subtree of d and $\overrightarrow{p(d)d} = -\boldsymbol{y}$ is at most $\pi/2$, i.e., they are directed downwards.

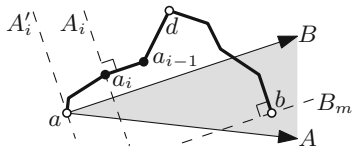


Fig. 5. Illustration of case 3 in the proof of Lemma 7

Let $A_i, 1 \leq i \leq k$ be the straight line through a_i and perpendicular to $\overrightarrow{a_{i-1}a_i}$. Let A_i' be the parallel line to A_i that passes through a. Due to the x-monotonicity of p the point a lies below A_i. During the construction of the tree, the line A_i defined a circular sector in which the subtree rooted at a_i including a was exclusively drawn. It follows that a and b lie on opposite sites of A_i. Thus, b lies above A_i and also above A_i'. Let $B_j, 1 \leq j \leq m$ be the straight line through b_j and perpendicular to $\overrightarrow{b_{j-1}b_j}$. Let B_j' be the parallel line to B_j that passes through a. By construction, b lies below B_j and a lies above B_j. Thus, b lies below B_j'.

Let A be the line A_i' with maximum slope and let B be the line B_j' with minimum slope. First, we will show that the path is monotone with respect to the unit vector $\boldsymbol{A}$ on A directed to the right. By choice of A, the angle between any edge $(a_i, a_{i-1}), 1 \leq i \leq k$ and $\boldsymbol{A}$ is at most $\pi/2$. Recall that any vector $\overrightarrow{a_i, a_{i-1}}$, $1 \leq i \leq k$ lies between $\boldsymbol{x}$ and $\boldsymbol{y}$. Since $\boldsymbol{A}$ is perpendicular to one of these edges and directed to the right, it lies

between $\boldsymbol{x}$ and $-\boldsymbol{y}$. Since any vector $\overrightarrow{b_{j-1}, b_j}$, $1 \leq j \leq m$ also lies between $\boldsymbol{x}$ and $-\boldsymbol{y}$, the angle between $\boldsymbol{A}$ and these edges is at most $\pi/2$. Because the angle between $\boldsymbol{A}$ and any edge on the path from a to b is at most $\pi/2$, the path is monotone with respect to $\boldsymbol{A}$.

Analogously, it can be shown that the path is monotone with respect to $\boldsymbol{B}$. Recall that b lies above A and below B. So the vector $\overrightarrow{ab}$ lies between $\boldsymbol{A}$ and $\boldsymbol{B}$. Following Lemma 6, the path is monotone with respect to $\overrightarrow{ab}$ and thus strongly monotone. □

Lemmas 5 and 7 immediately imply the following.

Theorem 2. *Any proper binary tree rooted in a dummy vertex has a strongly monotone and strictly convex drawing.*

Next, we (partially) extend this result to arbitrary trees.

Theorem 3. *Any tree has a strongly monotone drawing.*

Proof. Let T be a tree. If T has a vertex of degree 2, we root T in this vertex. Otherwise, we subdivide any edge by creating a vertex of degree 2, which we pick as root. Then, we add a leaf to every vertex of degree 2, except the root. Now, let v be a vertex with out-degree $k > 2$. Let $(v, w_1), \ldots, (v, w_k)$ be the outgoing edges of v ordered from right to left. We substitute v by a path $\langle v = v_1, \ldots, v_{k+1} \rangle$, where v_{i+1} is the left child of v_i, for $i = 1, \ldots, k$. Then, we substitute the edges (v, w_i) by $(v_i, w_i), i = 2, \ldots, k$; see Fig. 6.

Let T' be the resulting proper binary tree. Clearly, all vertices of T', except its root, have degree 1 or 3, so T' is a proper binary tree. We use Theorem 2 to get a strongly monotone drawing $\Gamma_{T'}$ of T'. Then, we remove the dummy vertices inserted above and draw the edges that have been subdivided by a path as a straight-line. This yields a drawing Γ_T of T that is crossing-free since the only new edges form a set of stars that are drawn in disjoint areas of the drawing.

Now, we show that Γ_T is strongly monotone. Let (v, w) be an edge in T. Let $p = \langle v = v_1, \ldots, v_m = w \rangle$ be the path in T' between v and w. Suppose p is monotone with respect to some direction $\boldsymbol{d}$. Thus, $\angle\{\overrightarrow{v_i v_{i+1}}, \boldsymbol{d}\} < \pi/2$ for $1 \leq i \leq m-1$. Clearly, $\overrightarrow{vw} = \sum_{i=1}^{m-1} \overrightarrow{v_i v_{i+1}}$ is a positive linear combination of $\overrightarrow{v_i v_{i+1}}, i = 1, \ldots, m$ and hence $\angle\{\overrightarrow{vw}, \boldsymbol{d}\} < \pi/2$. It follows that the path between two vertices a and b is monotone to a direction $\boldsymbol{d}$ in Γ_T if the path between a and b is monotone to $\boldsymbol{d}$ in $\Gamma_{T'}$. With $\boldsymbol{d} = \overrightarrow{ab}$, it follows that Γ_T is strongly monotone. □

We add to this another positive result concerning biconnected outerplanar graphs.

Theorem 4. *Any biconnected outerplanar graph has a strongly monotone and strictly convex drawing.*

Proof. Let G be a biconnected outerplanar graph with outer cycle $\langle v_1, \ldots, v_n, v_1 \rangle$. We place the vertices $v_2, \ldots, v_{n-1}$ in counterclockwise order on a quarter circle C that has $v_1 = (0, 0)$ and $v_n = (1, 1)$ as its endpoints; see Fig. 7. Since the outer cycle is drawn strictly convex, the drawing is planar and strictly convex. Clearly, the path $\langle v_1, \ldots, v_n \rangle$ is x- and y-monotone. Also, every vector $\overrightarrow{v_i v_j}, j > i$ lies between $\boldsymbol{x} = (0, 1)$ and $\boldsymbol{y} = (1, 0)$. Thus, by Lemma 6, the drawing is strongly monotone. □

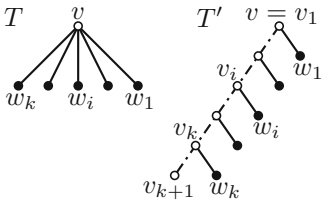


Fig. 6. Subdivision of a vertex v with k outgoing edges

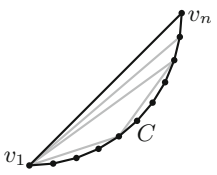


Fig. 7. A strongly monotone drawing of a biconnected outerplanar graph

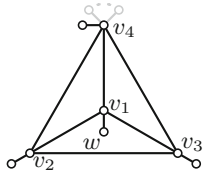


Fig. 8. A planar graph without any strongly monotone drawing

We close with a negative result. Note that the graphs in the family that we construct are neither outerplanar nor biconnected.

Theorem 5. *There is an infinite family of connected planar graphs that do not have a strongly monotone drawing in any combinatorial embedding.*

Proof. Let C be the graph that arises by attaching to each vertex of K_4 a "leaf"; see Fig. 8. Let $v_1, \ldots, v_4$ be the vertices of K_4. For K_4 to be crossing-free, one of its vertices, say v_1, lies in the interior. Let w be the leaf incident to v_1. Because of planarity, w has to be placed inside a triangular face incident to v_1. W.l.o.g., assume that w is placed in the face (v_1, v_2, v_3). If the drawing is strongly monotone, then $\angle(\overrightarrow{wv_2}, \overrightarrow{wv_1}) < \pi/2$ and $\angle(\overrightarrow{wv_1}, \overrightarrow{wv_3}) < \pi/2$ and thus $\angle(\overrightarrow{wv_3}, \overrightarrow{wv_2}) > \pi$. However, this means that w does not lie inside the triangle (v_1, v_2, v_3), which is a contradiction to the assumption. Thus, C does not have a strongly monotone drawing in any combinatorial embedding. We create an infinite family from C by adding more leaves to the vertices of K_4. □

5 Conclusion and Open Problems

We have shown that any tree has a monotone drawings on a grid with area $O(n^3)$ and a strongly monotone drawing, but can we combine the two features, that is, does any tree have a strongly monotone drawing on a grid of polynomial size?

Angelini et al. [2, Fig. 18(b)] have constructed a drawing of a triangulation that is not strongly monotone. But is there a triconnected (or biconnected) planar graph that does not have any strongly monotone drawing? If yes, can this be tested efficiently?

If we could show that our drawings for general trees are not just strongly monotone but also convex (as in the proper binary case), then all Halin graphs would automatically have convex and strictly monotone drawings, too.

References

1. Alamdari, S., Chan, T.M., Grant, E., Lubiw, A., Pathak, V.: Self-approaching graphs. In: Didimo, W., Patrignani, M. (eds.) GD 2012. LNCS, vol. 7704, pp. 260–271. Springer, Heidelberg (2013)
2. Angelini, P., Colasante, E., Battista, G.D., Frati, F., Patrignani, M.: Monotone drawings of graphs. J. Graph Algorithms Appl. 16(1), 5–35 (2012)

3. Angelini, P., Didimo, W., Kobourov, S., Mchedlidze, T., Roselli, V., Symvonis, A., Wismath, S.: Monotone drawings of graphs with fixed embedding. To appear in Algorithmica, http://dx.doi.org/10.1007/s00453-013-9790-3
4. Arkin, E.M., Connelly, R., Mitchell, J.S.: On monotone paths among obstacles with applications to planning assemblies. In: Proc. 5th Ann. ACM Symp. Comput. Geom. (SoCG 1989), pp. 334–343 (1989)
5. Bárány, I., Rote, G.: Strictly convex drawings of planar graphs. Doc. Math. 11, 369–391 (2006)
6. Carlson, J., Eppstein, D.: Trees with convex faces and optimal angles. In: Kaufmann, M., Wagner, D. (eds.) GD 2006. LNCS, vol. 4372, pp. 77–88. Springer, Heidelberg (2007)
7. Dhandapani, R.: Greedy drawings of triangulations. Discrete Comput. Geom. 43(2), 375–392 (2010)
8. Hardy, G., Wright, E.M.: An Introduction to the Theory of Numbers, 5th edn. Oxford University Press (1979)
9. Hossain, M. I., Rahman, M. S.: Monotone Grid Drawings of Planar Graphs. In: Chen, J., Hopcroft, J.E., Wang, J. (eds.) FAW 2014. LNCS, vol. 8497, pp. 105–116. Springer, Heidelberg (2014)
10. Leighton, T., Moitra, A.: Some results on greedy embeddings in metric spaces. Discrete Comput. Geom. 44(3), 686–705 (2010)
11. Nöllenburg, M., Prutkin, R.: Euclidean greedy drawings of trees. In: Bodlaender, H.L., Italiano, G.F. (eds.) ESA 2013. LNCS, vol. 8125, pp. 767–778. Springer, Heidelberg (2013)