Drawing Trees with Perfect Angular Resolution and Polynomial Area

Christian A. Duncan[1], David Eppstein[2], Michael T. Goodrich[2],
Stephen G. Kobourov[3], and Martin Nöllenburg[2]

[1]Department of Computer Science, Louisiana Tech. Univ., Ruston, Louisiana, USA
[2]Department of Computer Science, University of California, Irvine, California, USA
[3]Department of Computer Science, University of Arizona, Tucson, Arizona, USA

Abstract. We study methods for drawing trees with perfect angular resolution, i.e., with angles at each vertex, v, equal to $2\pi/d(v)$. We show:

1. Any unordered tree has a crossing-free straight-line drawing with perfect angular resolution and polynomial area.
2. There are ordered trees that require exponential area for any crossing-free straight-line drawing having perfect angular resolution.
3. Any ordered tree has a crossing-free ***Lombardi-style*** drawing (where each edge is represented by a circular arc) with perfect angular resolution and polynomial area.

Thus, our results explore what is achievable with straight-line drawings and what more is achievable with Lombardi-style drawings, with respect to drawings of trees with perfect angular resolution.

1 Introduction

Most methods for visualizing trees aim to produce drawings that meet as many of the following aesthetic constraints as possible:

1. straight-line edges,
2. crossing-free edges,
3. polynomial area, and
4. perfect angular resolution around each vertex.

These constraints are all well-motivated, in that we desire edges that are easy to follow, do not confuse viewers with edge crossings, are drawable using limited real estate, and avoid congested incidences at vertices. Nevertheless, previous tree drawing algorithms have made various compromises with respect to this set of constraints; we are not aware of any previous tree-drawing algorithm that can achieve all these goals simultaneously. Our goal in this paper is to show what is actually possible with respect to this set of constraints and to expand it further with a richer notion of edges that are easy to follow. In particular, we desire tree-drawing algorithms that satisfy all of these constraints simultaneously. If this is provably not possible, we desire an augmentation that avoids compromise and instead meets the spirit of all of these goals in a new way, which, in the case of this paper, is inspired by the work of artist Mark Lombardi [17].

Problem Statement. The art of Mark Lombardi involves drawings of social networks, typically using circular arcs and good angular resolution. Figure 1 shows such a work of Lombardi that is crossing-free and almost a tree. Note that it makes use of both circular arcs and straight-line edges. Inspired by this work, let us define a set of problems that explore what is achievable for drawings of trees with respect to the constraints listed above but that, like Lombardi's drawings, also allow curved as well as straight edges.

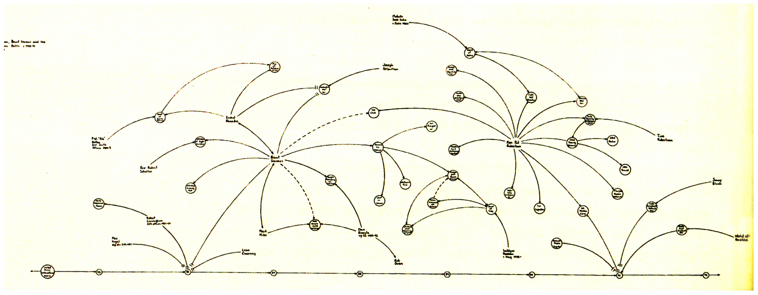
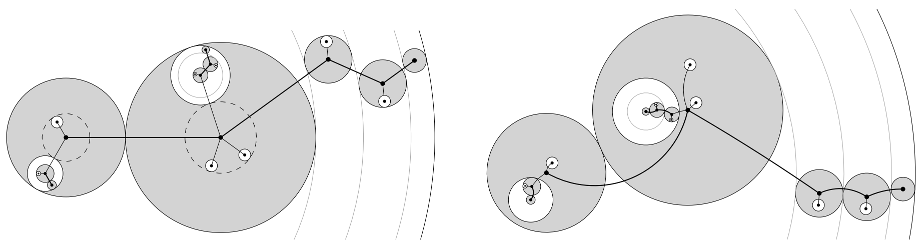


Fig. 1: Mark Lombardi, *Pat Robertson, Beurt Servaas, and the UPI Takeover Battle, ca. 1985-91*, 2000 [17].

Given a graph $G = (V, E)$, let $d(u)$ denote the ***degree*** of a vertex u, i.e., the number of edges incident to u in G. For any drawing of G, the ***angular resolution*** at a vertex u is the minimum angle between two edges incident to u. A vertex has ***perfect angular resolution*** if its minimum angle is $2\pi/d(u)$, and a drawing has perfect angular resolution if *every* vertex does. Drawings with perfect angular resolution cannot be placed on an integer grid unless the degrees of the vertices are constrained, so we do not require vertices to have integer coordinates. We define the ***area*** of a drawing to be the ratio of the area of a smallest enclosing circle around the drawing to the square of the distance between its two closest vertices.

Suppose that our input graph, G, is a rooted tree T. We say that T is ***ordered*** if an ordering of the edges incident upon each vertex in T is specified. Otherwise, T is ***unordered***. If all the edges of a drawing of T are straight-line segments, then the drawing of T is a ***straight-line*** drawing. We define a ***Lombardi drawing*** of a graph G as a drawing of G with perfect angular resolution such that each edge is drawn as a circular arc. When measuring the angle formed by two circular arcs incident to a vertex v, we use the angle formed by the tangents of the two arcs at v. Circular arcs are strictly more general than straight-line segments, since straight-line segments can be viewed as circular arcs with infinite radius. Figure 2 shows an example of a straight-line drawing and a Lombardi drawing for the same tree. Thus, we can define our problems as follows:

1. Is it always possible to produce a straight-line drawing of an unordered tree with perfect angular resolution and polynomial area?
2. Is it always possible to produce a straight-line drawing of an ordered tree with perfect angular resolution and polynomial area?
3. Is it always possible to produce a Lombardi drawing of an ordered tree with perfect angular resolution and polynomial area?



(a) Straight-line drawing for an unordered tree (b) Lombardi drawing for an ordered tree

Fig. 2: Two drawings of a tree T with perfect angular resolution and polynomial area as produced by our algorithms. Bold edges are heavy edges, gray disks are heavy nodes, and white disks are light children. The root of T is in the center of the leftmost disk.

Related Work. Tree drawings have interested researchers for many decades: e.g., hierarchical drawings of binary trees date to the 1970's [23]. Many improvements have been proposed since this early work, using space efficiently and generalizing to non-binary trees [2, 5, 12–14, 20–22]. These drawings do not achieve all the constraints mentioned above, however, especially the constraint on angular resolution.

Alternatively, several methods strive to optimize angular resolution of trees. Radial drawings of trees place nodes at the same distance from the root on a circle around the root node [10]. Circular tree drawings are made of recursive radial-type layouts [19]. Bubble drawings [15] draw trees recursively with each subtree contained within a circle disjoint from its siblings but within the circle of its parent. Balloon drawings [18] take a similar approach and heuristically attempt to optimize space utilization and the ratio between the longest and shortest edges in the tree. Convex drawings [4] partition the plane into unbounded convex polygons with their boundaries formed by tree edges. Although these methods provide several benefits, none of these methods guarantees that they satisfy all of the aforementioned constraints.

The notion of drawing graphs with edges that are circular arcs or other nonlinear curves is certainly not new to graph drawing. For instance, Cheng *et al.* [6] used circle arcs to draw planar graphs in an $O(n) \times O(n)$ grid while maintaining bounded (but not perfect) angular resolution. Similarly, Dickerson *et al.* [7] use circle-arc polylines to produce planar confluent drawings of non-planar graphs, Duncan *et al.* [8] draw graphs with fat edges that include circular arcs, and Cappos *et al.* [3] study simultaneous embeddings of planar graphs using circular arcs. Finkel and Tamassia [11] use a force-directed method for producing curvilinear drawings, and Brandes and Wagner [1] use energy minimization methods to place Bézier splines that represent express connections in a train network. In a separate paper [9] we study Lombardi drawings for classes of graphs other than trees.

Our Contributions. In this paper we present the first algorithm for producing straight-line, crossing-free drawings of unordered trees that ensures perfect angular resolution and polynomial area. In addition we show, in Section 3, that if the tree is ordered (i.e., given with a fixed combinatorial embedding) then it is not always possible to maintain perfect angular resolution and polynomial drawing area when using straight lines for edges. Nevertheless, in Section 4, we show that crossing-free polynomial-area Lombardi drawings of ordered trees are possible. That is, we show that the answers to the questions posed above are "yes," "no," and "yes," respectively.

2 Straight-line drawings for unordered trees

Let T be an unordered tree with n nodes. We wish to construct a straight-line drawing of T with perfect angular resolution and polynomial area.

The main idea of our algorithm is, similarly to the common bubble and balloon tree constructions [15, 18], to draw the children of each node of the given tree in a disk centered at that node; however, our algorithm differs in several key respects in order to achieve the desired area bounds and perfect angular resolution.

2.1 Heavy Path Decomposition

The initial step before drawing the tree T is to create a heavy path decomposition [16] of T. To make the analysis simpler, we assume T is rooted at some arbitrary node r. We let T_u represent the subtree of T rooted at u, and $|T_u|$ the number of nodes in T_u. A node c is the ***heavy child*** of u if $|T_c| \geq |T_v|$ for all children v of u. In the case of a tie, we arbitrarily designate one node as the heavy child. We refer to the non-heavy children as ***light*** and let $L(u)$ denote the set of all light children of u. The ***light subtrees*** of u are the subtrees of all light children of u. We define $l(u) = 1 + \sum_{v \in L(u)} |T_v|$ to be the ***light size*** of u. An edge is called a ***heavy edge*** if it connects a heavy child to its parent; otherwise it is a ***light edge***. The set of all heavy edges creates the ***heavy-path decomposition*** of T, a disjoint set of (heavy) paths where every node in T belongs to exactly one path, see Figure 3. The heavy path decomposition has the following interesting property. If we treat each heavy path as a node, and each light edge as connecting two heavy-path nodes, we obtain a tree $H(T)$. This tree has height $h(T) \leq \log_2 n$ since the size of each light child is less than half the size of its parent. We refer to the ***level*** of a heavy path as the depth of the corresponding node in the decomposition tree, where the root has depth 0. We extend this notion to nodes, i.e., the level of a node v is the level of the heavy path to which v belongs.

2.2 Drawing Algorithm

Our algorithm draws T incrementally in the order of a depth-first traversal of the corresponding heavy-path decomposition tree $H(T)$, i.e., given drawings of the lower-level heavy paths (the light children and their descendents) connected to a heavy path P in $H(T)$ we construct a drawing of P and its subtrees. Let $P = (v_1, \ldots, v_k)$ be a heavy path. Then we draw each node v_i of P in the center of a disk D_i and place smaller disks containing the drawings of the light children of v_i and their descendents around v_i in two concentric annuli of D_i. We guarantee perfect angular resolution at v_i by connecting the centers of the child disks with appropriately spaced straight-line edges to v_i. Next, we create the drawing of P and its descendents within a disk D by placing D_1 in the

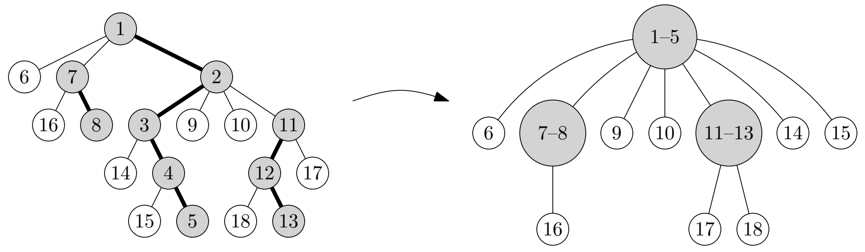


Fig. 3: The tree T on the left highlights its heavy edges. The corresponding heavy-path decomposition tree $H(T)$ on the right has each heavy path represented by a single node.

center of D and $D_2, \ldots, D_k$ on concentric circles around D_1. We show that the radius of D is linear in the number $n(P)$ of nodes descending from P and exponential in the level of P. In this way, at each step downwards in the heavy path decomposition, the total radius of the disks at that level shrinks by a constant factor, allowing room for disks at lower levels to be placed within the higher-level disks. Figure 2a shows a drawing of an unordered tree according to our method.

Before we can describe the details of our construction we require the following simple geometric property. We define an (R,δ)***-wedge***, $\delta \leq \pi$ as a sector of angle δ of a radius-R disk, see Figure 4.

Lemma 2.1. *The largest disk that fits inside an* (R,δ)*-wedge has radius* $r = R\frac{\sin(\delta/2)}{1+\sin(\delta/2)}$.

Proof. The largest disk inside the (R,δ)-wedge touches the circular arc and both radii of the wedge. Thus we immediately obtain a right triangle formed by the apex of the wedge, the center of the disk we want to fit, and one of its tangency points with the two radii of the wedge, see Figure 4. This triangle has one side of length r and the hypothenuse of length $R-r$. From $\sin(\delta/2) = \frac{r}{R-r}$ we obtain $r = R\frac{\sin(\delta/2)}{1+\sin(\delta/2)}$. □

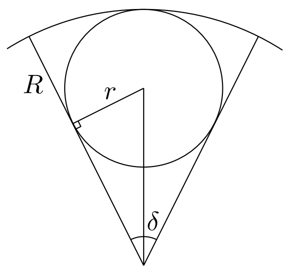


Fig. 4: An (R,δ)-wedge and the largest disk that can be placed inside it.

In the next lemma we show how to draw a single node v of a heavy path P given drawings of all its light subtrees.

Lemma 2.2. *Let v be a node of T at level j of $H(T)$ with degree $d(v) \geq 2$ and two incident heavy edges. For each light child $u \in L(v)$ assume there is a disk D_u of radius $r_u = 2 \cdot 8^{h(T)-j-1}|T_u|$ that contains a fixed drawing of T_u with perfect angular resolution and such that u is in the center of D_u. Then we can construct a drawing of v and its light subtrees inside a disk D such that the following properties hold:*

1. *the edge between v and any light child $u \in L(v)$ is a straight-line segment that does not intersect any disk other than D_u;*
2. *the heavy edges do not intersect any disk D_u;*
3. *any two disks D_u and $D_{u'}$ for $u \neq u'$ are disjoint;*
4. *the angular resolution of v is $2\pi/d(v)$;*
5. *the angle between the two heavy edges is at least $2\pi/3$ and at most $4\pi/3$;*
6. *the disk D has radius $r_v = 8^{h(T)-j} l(v)$.*

Proof. We assume that the heavy edge to the parent of v is directed horizontally to the left. We draw a disk D with radius r_v centered at v and create $d(v)$ ***spokes***, i.e., rays extending from v including the fixed heavy edge and being equally spaced by an angle of $2\pi/d(v)$. Obviously, every neighbor of v must be placed on a distinct spoke in order to satisfy properties 3 and 4. The main difficulty is that there can be child disks that are too large to place without overlap on adjacent spokes inside D.

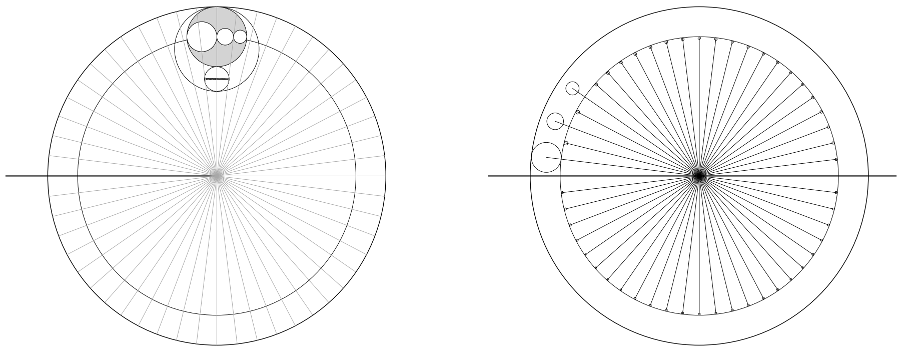
Let $D_{\max}$ be the largest disk D_u of any $u \in L(v)$ and let $r_{\max}$ be its radius. We split D into an outer annulus A and an inner disk B by a concentric circle of radius $R = r_v - 2r_{\max}$, see Figure 5. We define a child $u \in L(v)$ to be a ***small*** child, if its radius $r_u \leq R\frac{\sin(\pi/d(v))}{1+\sin(\pi/d(v))}$, and to be a ***large*** child otherwise. We further say D_u is a small (large) disk if u is a small (large) child. We denote the number of small children as n_s and the number of large children as n_l. By Lemma 2.1 we know that any small disk D_u can be placed inside an $(R, 2\pi/d(v))$-wedge. This means that we can place all n_s small disks on any subset of n_s spokes inside B without violating property 3. So once we have placed all large disks correctly then we can always distribute the small children on the unused spokes.

We place all large disks in the outer annulus A. Observe that

$$4 \sum_{u \in L(v)} r_u = 4 \sum_{u \in L(v)} 2 \cdot 8^{h(T)-j-1}|T_u| = 8^{h(T)-j} \sum_{u \in L(v)} |T_u| < 8^{h(T)-j} l(v) = r_v,$$

i.e., we can place all light children on the diameter of a disk of radius at most $r_v/4$. If we order all light children along that diameter by their size we can split them into one disk containing the large disks and one containing the small disks, see Figure 5a.

Assume that the large disks are arranged on the horizontal diameter of their disk and that this disk is placed vertically above v and tangent to D as shown in Figure 5a. Since that disk has radius at most $r_v/4$ we can use Lemma 2.1 to show that it always fits inside an $(r_v, \pi/4)$-wedge. If we now translate the large disks vertically upward onto a circle centered at v with radius $r_v - r_{\max}$ then they are still disjoint and they all lie in the intersection of A and the $(r_v, \pi/4)$-wedge. We now rotate them counterclockwise around v until the leftmost disk $D_{\max}$ touches the horizontal heavy edge. Thus all large disks are placed disjointly inside a $\pi/4$-sector of A. However, they are not centered on the spokes yet.


(a) All light children fit into a disk of radius $r_v/4$ and are split into small and large disks.

(b) Large disks are placed in the outer annulus and small disks in the inner disk.

Fig. 5: Drawing a node v and its light children $L(v)$.

Beginning from the leftmost large disk, we rotate each large disk D_u and all its right neighbors clockwise around v until D_u snaps to the next available spoke. Clearly, in each of the n_l steps we rotate by at most $2\pi/d(v)$ in order to reach the next spoke.

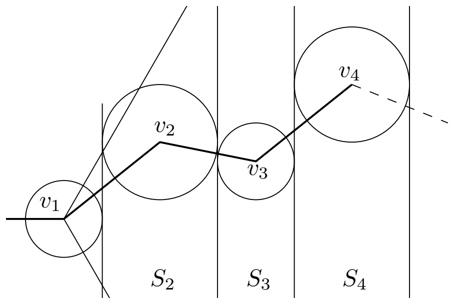
We now bound the number n_l of large children. By definition a child is large if $r_u = 2 \cdot 8^{h(T)-j-1}|T_u| > (r_v - 2r_{\max})\frac{\sin(\pi/d(v))}{1+\sin(\pi/d(v))}$. We also have $r_v \geq 8^{h(T)-j}\sum_{u\in L(v)}|T_u|$. Let w be the light child of v with maximum disk radius $r_w = r_{\max}$. Then $r_w = 2\cdot 8^{h(T)-j-1}|T_w|$ and hence $r_v - 2r_{\max} \geq 4\cdot 8^{h(T)-j-1}(2\sum_{u\in L(v)}|T_u| - |T_w|)$. So for a light child u to be large its subtree T_u has to contain $|T_u| > 2\cdot(2\sum_{u\in L(v)}|T_u| - |T_w|)\frac{\sin(\pi/d(v))}{1+\sin(\pi/d(v))}$ nodes. This yields

$$n_l < 1 + \frac{\sum_{u\in L(v)}|T_u| - |T_w|}{2\cdot(2\sum_{u\in L(v)}|T_u| - |T_w|)\frac{\sin(\pi/d(v))}{1+\sin(\pi/d(v))}} < 1 + \frac{1+\sin(\pi/d(v))}{4\sin(\pi/d(v))}.$$

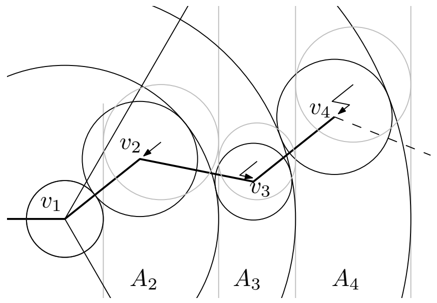
From this we obtain that for $d(v) \geq 5$ we have $n_l < 3d(v)/8$. So for $d(v) \geq 5$ we can always place all large disks correctly on spokes inside at most half of the outer annulus A since we initially place all large disks in a $\pi/4$-wedge and then enlarge that wedge by at most $3d(v)/8 \cdot 2\pi/d(v) = 3\pi/4$ radians. For $d(v) = 2$ there are no light children, for $d(v) = 3$ we immediately place the single light child on its spoke without intersecting the two heavy edges, and for $d(v) = 4$ we place the two light children on opposite vertical spokes separated by the two heavy edges, which does not produce any intersections either.

Since we require at most half of A to place all large children, we can assign the remaining heavy edge to the spoke exactly opposite of the first heavy edge if $d(v)$ is even. If $d(v)$ is odd, we choose one of the two spokes whose angle with the fixed heavy edge is closest to π. Finally, we arbitrarily assign the n_s small children to the remaining free spokes inside the inner disk B.

By construction the drawing for v and its light subtrees obtained in this way satisfies properties 1–6. □



(a) Placing disks in vertical strips.



(b) Final transformation of the drawing.

Fig. 6: Constructing the heavy path drawing by appending drawings of its heavy nodes.

Lemma 2.2 shows how to draw a single heavy node v and its light subtrees. It also applies to the root of T if we ignore the incoming heavy edge, and to the root node v_1 of a heavy path $P = (v_1, \dots, v_k)$ at level $l \geq 1$ if we consider the light edge uv_1 to its parent u as a heavy edge for v_1. We note that the last node v_k of P is always a leaf that is trivial to draw. For drawing an entire heavy path $P = (v_1, \dots, v_k)$ we need to link the drawings of the heavy nodes into a path.

Lemma 2.3. *Given a heavy path $P = (v_1, \dots, v_k)$ and a drawing for each v_i and its light subtrees inside a disk D_i of radius r_i, we can draw P and all its descendents inside a disk D such that the following properties hold:*

1. *the heavy edge v_iv_{i+1} is a straight-line segment that does not intersect any disk other than D_i and D_{i+1};*
2. *the light edge connecting v_1 and its parent does not intersect the drawing of P;*
3. *any two disks D_i and D_j for $i \neq j$ are disjoint;*
4. *the drawing has perfect angular resolution;*
5. *the radius r of D is $r = 2\sum_{i=1}^{k} r_i$.*

Proof. Let v_1 be the root of P and let u be the parent of v_1 (unless P is the heavy path at level 0). We place the disk D_1 at the center of D and assume that the edge uv_1 extends horizontally to the left. We create $k-1$ vertical strips $S_2, \dots, S_k$ to the right of D_1, each S_i of width $2r_i$, see Figure 6a. Each disk D_i will be placed inside its strip S_i. So from v_1 we extend the ray induced by the stub reserved for the heavy edge v_1v_2 until it intersects the vertical line bisecting S_2. We place v_2 at this intersection point. By property 5 of Lemma 2.2 we know that the angle between the two heavy edges incident to a heavy node is between $2\pi/3$ and $4\pi/3$. Thus v_2 is inside a right-open $2\pi/3$-wedge W that is symmetric to the x-axis. Now for $i = 2, \dots, k$ we extend from v_i the stub of the heavy edge v_iv_{i+1} into a ray and place v_{i+1} at the intersection of that ray and the bisector of S_{i+1}. Since at each v_i we can either place D_i or its mirror image we know that one of the two possible rays stays within W.

Since each disk D_i is placed in its own strip S_i no two disks intersect (property 3) and since heavy edges are straight-line segments within two adjacent strips they do not intersect any non-incident disks (property 1). The light edge uv_1 is completely to the left of all strips and thus does not intersect the drawing of P (property 2). Since we were using the existing drawings (or their mirror images) of all heavy nodes, their perfect angular resolution is preserved (property 4).

The current drawing has a width that is equal to the sum of the diameters of the disks $D_1, \dots, D_k$. However, it does not yet necessarily fit into a disk D centered at v_1 whose radius equals that sum of the diameters. To achieve this we create $k-1$ annuli $A_2, \dots, A_k$ centered around v_1, each A_i of width $2r_i$. Then from $i = 2, \dots, k$ we either shorten or extend the edge $v_{i-1}v_i$ until D_i is contained in its annulus A_i, see Figure 6b. At each step i we treat the remaining path $(v_i, \dots, v_k)$ and its disks $D_i, \dots, D_k$ as a rigid structure that is translated as a whole, see the translation vectors indicated in Figure 6b. In the end, each disk D_i is contained in its own annulus A_i and thus all disks are still pairwise disjoint. Since we only stretch or shrink edges of an x-monotone path but do not change any edge directions, the whole transformation preserves the previous properties of the drawing. Clearly, all disks now lie inside a disk D of radius $r = r_1 + 2\sum_{i=2}^{k} r_i \leq 2\sum_{i=1}^{k} r_i$ (property 5). □

Combining Lemmas 2.2 and 2.3 we now obtain the following theorem:

Theorem 2.4. *Given an unordered tree T with n nodes we can find a crossing-free straight-line drawing of T with perfect angular resolution that fits inside a disk D of radius $2 \cdot 8^{h(T)}n$, where $h(T)$ is the height of the heavy-path decomposition of T. Since $h(T) \leq \log_2 n$ the radius of D is no more than $2n^4$.*

Proof. From Lemma 2.2 we know that for each node v of a heavy path P at level j the radius of the disk D containing v and all its light subtrees is $r_v = 8^{h(T)-j}l(v)$. So if $P = (v_1, \dots, v_k)$ Lemma 2.3 yields that P and all its descendents can be drawn in a disk of radius $r = 2\sum_{i=1}^{k} r_{v_i} = 2 \cdot 8^{h(T)-j}\sum_{i=1}^{k} l(v_i) = 2 \cdot 8^{h(T)-j}n(P)$, where $n(P)$ is the number of nodes of P and its descendents. This holds, in particular, for the heavy path $\hat{P}$ at the root of $H(T)$, which proves the theorem. □

3 Straight-line drawings for ordered trees

In many cases, the ordering of the children around each vertex of a tree is given; that is, the tree is ordered (or has a fixed combinatorial embedding). In the previous section we rely on the freedom to order subtrees as needed to achieve a polynomial area bound. Hence that algorithm cannot be applied to ordered trees with fixed embeddings. As we now show, there are ordered trees that have no straight-line crossing-free drawings with polynomial area and perfect angular resolution.

Specifically we present a class of ordered trees for which any straight-line crossing-free drawing of the tree with perfect angular resolution requires exponential area. Figure 7a shows a caterpillar tree, which we call the ***Fibonacci caterpillar*** because of its simple behavior when required to have perfect angular resolution. This tree has as its spine a k-vertex path, each vertex of which has 3 additional leaf nodes embedded on the same side of the spine. When drawn with straight-line edges, no crossings, and with

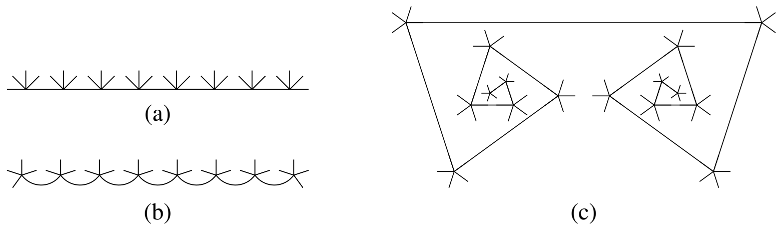


Fig. 7: (a) A Fibonacci caterpillar; (b) Lombardi drawing; (c) Straight-line drawing with perfect angular resolution and exponential area.

perfect angular resolution, the caterpillar is forced to spiral (a single or a double spiral). The best drawing area, exponential in the number of vertices in the caterpillar, is achieved when the caterpillar forms a symmetric double spiral; see Figure 7c.

The Fibonacci caterpillar shows that we cannot maintain all constraints (straight-line edges, crossing-free, perfect angular resolution, polynomial area) for ordered trees. However, as we show next, using circular arcs instead of straight-line edges allows us to respect the remaining three constraints. See, for example, Figure 7b.

4 Lombardi drawings for ordered trees

In this section, let T be an ordered tree with n nodes. As we have seen in Section 3, we cannot find polynomial area drawings for all ordered trees using straight-line edges. An augmentation of the straight-line edge requirement is the use of circular arcs as edges. Circular arcs are curves that are not only still easy to follow visually but they also let us achieve all remaining three constraints, i.e., we can find crossing-free circular arc drawings with perfect angular resolution and polynomial area. We call a drawing with circular arcs and perfect angular resolution a Lombardi drawing, so in other words we aim for crossing-free Lombardi drawings with polynomial area.

The flavor of the algorithm for Lombardi tree drawings is similar to our straight-line tree drawing algorithm of Section 2: We first compute a heavy-path decomposition $H(T)$ for T. Then we recursively draw all heavy paths within disks of polynomial area. Unlike before, we need to construct the drawing in a top-down fashion since the placement of the light children of a node v now depends on the curvature of the two heavy edges incident to v.

Our construction in this section uses the invariant that a heavy path P at level j is drawn inside a disk D of radius $2 \cdot 4^{h(T)-j} n(P)$, where $n(P) = |T_v|$ for the root v of P.

4.1 Drawing heavy paths

Let $P = (v_1, \ldots, v_k)$ be a heavy path at level j of the heavy-path decomposition that is rooted at the last node v_k. We denote each edge $v_i v_{i+1}$ by e_i. Recall that the angle in an intersection point of two circular arcs is measured as the angle between the tangents to the arcs at that point. We define the angle $\alpha(v_i)$ for $2 \le i \le k-1$ to be the angle between

e_{i-1} and e_i in node v_i (measured counter-clockwise). The angle $\alpha(v_k)$ is defined as the angle in v_k between e_{k-1} and the light edge $e = v_k u$ connecting the root v_k of P to its parent u. Due to the perfect angular resolution requirement for each node v_i, the angle $\alpha(v_i)$ is obtained directly from the number of edges between e_{i-1} and e_i and the degree $d(v_i)$.

Lemma 4.1. *Given a heavy path $P = (v_1, \dots, v_k)$ and a disk D_i of radius r_i for the drawing of each v_i and its light subtrees, we can draw P with each v_i in the center of its disk D_i inside a large disk D such that the following properties hold:*

1. *each heavy edge e_i is a circular arc that does not intersect any disk other than D_i and D_{i+1};*
2. *there is a stub edge incident to v_k that is reserved for the light edge connecting v_k and its parent;*
3. *any two disks D_i and D_j for $i \neq j$ are disjoint;*
4. *the angle between any two consecutive heavy edges e_{i-1} and e_i is $\alpha(v_i)$;*
5. *the radius r of D is $r = 2\sum_{i=1}^{k} r_i$.*

Proof. We draw P incrementally starting from the leaf v_1 by placing D_1 in the center M of the disk D of radius $r = 2\sum_{i=1}^{k} r_i$. We may assume that D_1 is rotated such that the edge e_1 is tangent to a horizontal line at v_1 and that it leaves v_1 to the right. All disks $D_2, \dots, D_k$ will be placed with their centers $v_2, \dots, v_k$ on concentric circles $C_2, \dots, C_k$ around M. The radius of C_i is $r_1 + 2\sum_{j=2}^{i-1} r_j + r_i$ so that D_{i-1} and D_i are placed in disjoint annuli and hence by construction no two disks intersect (property 3). Each disk D_i will be rotated around its center such that the tangent to C_i at v_i is the bisector of the angle $\alpha(v_i)$.

We now describe one step in the iterative drawing procedure that draws edge e_i and disk D_{i+1} given a drawing of $D_1, \dots, D_i$. Disk D_i is placed such that C_i bisects the angle $\alpha(v_i)$ and hence we know the tangent of e_i at v_i. This defines a family $\mathscr{F}_i$ of circular arcs emitted from v_i that intersect the circle C_{i+1}, see Figure 8. We consider all arcs from v_i until their first intersection point with C_{i+1}. Observe that the intersection angles of $\mathscr{F}_i$ and C_{i+1} bijectively cover the full interval $[0, \pi]$, i.e., for any angle $\alpha \in [0, \pi]$ there is a unique arc in $\mathscr{F}_i$ that has intersection angle α with C_{i+1}. Hence we choose for e_i the unique circular arc that realizes the angle $\alpha(v_{i+1})/2$ and place the center v_{i+1} of D_{i+1} at the endpoint of e_i. We continue this process until the last disk D_k is placed. This drawing of P realizes the angle $\alpha(v_i)$ between any two heavy edges e_{i-1} and e_i (property 4). Note that for the edge from v_k to its parent we can only reserve a stub whose tangent at v_k has a fixed slope (property 2). Figure 10 shows an example.

Note that each edge e_i is contained in the annulus between C_i and C_{i+1} and thus does not intersect any other edge of the heavy path or any disk other than D_i and D_{i+1} (property 1). Furthermore, the disk D with radius $r = 2\sum_{i=1}^{k} r_i$ indeed contains all the disks $D_1, \dots, D_k$ (property 5). □

Lemma 4.1 shows how to draw a heavy path P with prescribed angles between the heavy edges and an edge stub to connect it to its parent. Since each heavy path P (except the path at the root of $H(T)$) is the light child of a node on the previous level of $H(T)$ that light edge is actually drawn when placing the light children of a node, which we describe next.

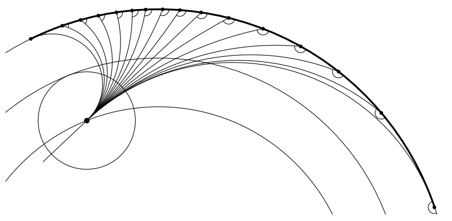


Fig. 8: Any angle $\alpha \in [0, \pi]$ can be realized.

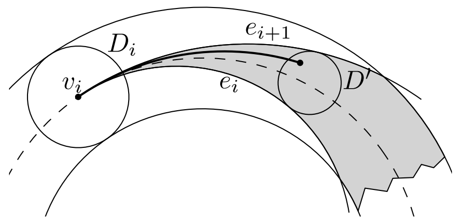


Fig. 9: Placing a single disk D' in the extended small zone of D_i (shaded gray).

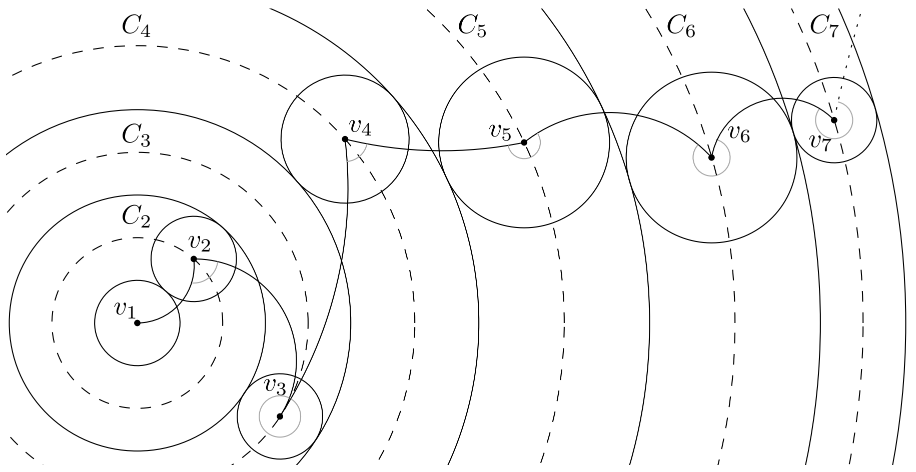


Fig. 10: Drawing a heavy path P on concentric circles with circular-arc edges. The angles $\alpha(v_i)$ are marked in gray; the edge stub to connect v_7 to its parent is dotted.

4.2 Drawing light children

Once the heavy path P is drawn as described above, it remains to place the light children of each node v_i of P. For each node v_i the two heavy edges incident to it partition the disk D_i into two regions. We call the region that contains the larger conjugate angle the ***large zone*** of v_i and the region that contains the smaller conjugate angle the ***small zone***. If both angles equal π, then we can consider both regions small zones.

For a node v_i at level j of $H(T)$ we define the radius r_i of D_i as $r_i = 4^{h(T)-j}(1 + \sum_{u \in L(v_i)} |T_u|) = 4^{h(T)-j} l(v_i)$. All light children of v_i are at level $j+1$ of $H(T)$ and thus by our invariant every light child u of v_i is drawn in a disk of radius $r_u = 2 \cdot 4^{h(T)-j-1}|T_u|$. Thus we know that $r_u \leq r_i/2$; in fact, we even have $\sum_{u \in L(v_i)} r_u \leq r_i/2$.

Light children in the small zone. Depending on the angle $\alpha(v_i)$, the small zone of a disk D_i might actually be too narrow to directly place the light children in it. Fortunately, we can always place another disk D' of radius at most $r_i/2$ in an extension of the small zone along the annulus of D_i in the drawing of P such that D' touches e_{i-1} and e_i and does

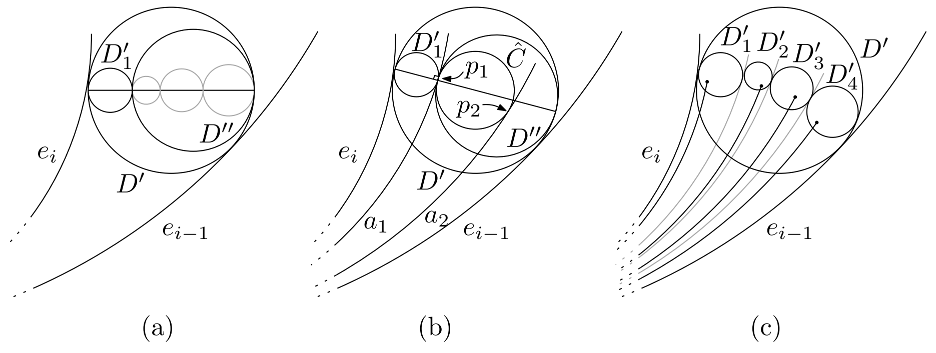


Fig. 11: Placing disks D'_1 and D'' inside the disk D'.

not intersect any other previously placed disk, see Figure 9. If there is a single child u in the small zone then $D' = D_u$ and we are done. The next lemma shows how to place more than one child.

Lemma 4.2. *If a single disk D' of radius r' can be placed in the possibly extended small zone of the disk D_i, then we can correctly place any sequence of l disks $D'_1, \dots, D'_l$ with radii $r'_1, \dots, r'_l$ and $\sum_{i=1}^{l} r'_i = r'$ in the (extended) small zone of D_i.*

Proof. The idea of the algorithm for placing the l disks is to first place the disk D' in the small zone as before. The disks $D'_1, \dots, D'_l$ will then be placed within D' so that no additional space is required.

In the first step of the recursive placement algorithm we either place D'_1 or D'_l (whichever has smaller radius) and a disk D'' containing the remaining sequence of disks $D'_2, \dots, D'_l$ or $D'_1, \dots, D'_{l-1}$, respectively. Without loss of generality, let $r'_1 \leq r'_l$ and thus in particular $r'_1 \leq r'/2$. In order to fit inside D' the disks D'_1 and D'' must be placed with their centers on a diameter of D', see Figure 11a. The degree of freedom that we have is the rotation of that diameter around the center of D'. Then the locus of the tangent point of D'_1 and D'' is a circle $\hat{C}$ of radius $r' - 2r'_1$ around the center of D', see Figure 11b. There are exactly two circular arcs a_1 and a_2 tangent to $\hat{C}$ that are also tangent to v_i with the slope required for the edge to D'_1. Let the two points of tangency on $\hat{C}$ be p_1 and p_2. Now we rotate D'_1 and D'' such that their point of tangency coincides with either p_1 or p_2 depending on which of them yields the correct embedding order of D'_1 and D'' around v_i. Clearly, a_1 or a_2 are also tangent to D'_1 and D'' now. Assume we choose p_1 and the corresponding arc a_1 as in Figure 11b. Then we can connect any point in D'_1 to v_i with the unique circular arc of the required slope in v_i. We will describe the exact placement of that arc later. Any such edge clearly stays inside the horn-shaped region that encloses D'_1 and is formed by a boundary arc of the small zone and a_1. Since a_1 separates D'_1 from D'', neither the new edge nor D'_1 can interfere with any of the disks $D'_2, \dots, D'_l$ and their respective edges as long as they stay inside D'' or connect to points in D''.

For placing $D'_2, \dots, D'_l$ we recursively apply the same procedure again, now using D'' as the disk D' and a_1 as one of the boundary arcs. Then after l steps, we have disjointly placed all disks $D'_1, \dots, D'_l$ inside the disk D' such that their order respects the given tree order and no two edges intersect. Figure 11c gives an example.

Note that we require that the edges e_{i-1} and e_i are tangent to D', which is possible only for an opening angle α of the small zone of at most π. For any angle $\alpha \leq \pi$ the arcs a_1 and a_2 always stay within the extended small zone and form at most a semi-circle. This does not hold for $\alpha > \pi$. □

Light children in the large zone. Placing the light children of a vertex v_i in the large zone of D_i must be done slightly different from the algorithm for the small zone since Lemma 4.2 holds only for opening angles of at most π. On the other hand, the large zone does not become too narrow and there is no need to extend it beyond D_i. Our approach splits the large zone into two parts that again have an opening angle of at most π so that we can apply Lemma 4.2 and draw all children accordingly.

Let l be the number of light children in the large zone of D_i. We first place a disk D' of radius at most $r_i/2$ such that it touches v_i and such that its center lies on the line bisecting the opening angle of the large zone. The disk D' is large enough to contain the disjoint disks $D'_1, \dots, D'_l$ for the light children of v_i along its diameter. We need to distinguish whether l is even or odd. For even l we create a container disk D''_1 for disks $D'_1, \dots, D'_{l/2}$ and a container disk D''_2 for $D'_{l/2+1}, \dots, D'_l$. Now D''_1 and D''_2 can be tightly packed on the diameter of D'. Using a similar argument as in Lemma 4.2 we separate the two disks by a circular arc through v_i that is tangent to the bisector of $\alpha(v_i)$ in v_i. Since D' is centered on the bisector this is possible even though the actual opening angle of the large zone is larger than π. If l is odd, we create a container disk D''_1 for disks $D'_1, \dots, D'_{\lfloor l/2 \rfloor}$ and a container disk D''_2 for $D'_{\lceil l/2 \rceil+1}, \dots, D'_l$. The median disk $D'_{\lceil l/2 \rceil}$ is not included in any container. Then we apply Lemma 4.2 to D' and the three disks $D''_1, D'_{\lceil l/2 \rceil}, D''_2$ along the diameter of D', see Figure 12a. The separating circular arcs in v_i are again tangent to the bisector of $\alpha(v_i)$, which is, since l is odd, also the correct slope for the circular arc connecting v_i to the median disk $D'_{\lceil l/2 \rceil}$.

In both cases we split the large zone and the sequence of light children to be placed into two parts that each have an opening angle at v_i of at most π between a separating circular arc and the edge e_{i-1} or e_i, respectively. Next, we move D''_1 and D''_2 along the separating circular arcs keeping their tangencies until they also touch the edge e_{i-1} or e_i, respectively. Then we can apply Lemma 4.2 to both container disks and thus place all light children in the large zone, see Figure 12b.

Drawing light edges The final missing step is how to actually connect a heavy node v_i to its light children given a position of v_i and positions of all disks containing its light subtrees. Let u be a light child of v_i and let D_u be the disk containing the drawing of T_u. When placing the disk D_u in the small or large zone of v_i we made sure that a circular arc from v_i with the tangent required for perfect angular resolution at v_i can reach any point inside D_u without intersecting any other edge or disk.

On the other side, we know by Lemma 4.1 that u is placed in the outermost annulus of D_u and that it has a stub for the edge $e = uv_i$. This stub is the required tangent for e in order to obtain perfect angular resolution in u. Let C_u be the circle that is the locus of u if we rotate D_u and the drawing of T_u around the center of D_u.

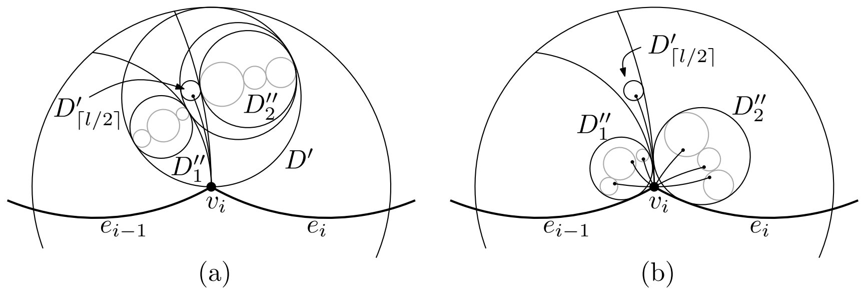


Fig. 12: Placing light children in the large zone by first splitting it into two parts (a) and then applying the algorithm for small zones to each part (b).

There is again a family $\mathscr{F}$ of circular arcs with the correct tangent in u that lead towards D_u and intersect the circle C_u. As observed in Lemma 4.1 the intersection angles formed between $\mathscr{F}$ and C_u bijectively cover the full interval $[0,\pi]$, i.e., for any angle $\alpha \in [0,\pi]$ there is a unique circular arc in $\mathscr{F}$ that has an intersection angle of α with C_u. In order to correctly attach u to v_i we first choose the arc a in $\mathscr{F}$ that realizes an intersection angle of $\alpha(u)/2$ with C_u, where $\alpha(u)$ is the angle between e and the heavy edge from u to its heavy child that is required for perfect angular resolution in u. Let p be the intersection point of that arc with C_u. Then we rotate D_u and the drawing of T_u around the center of D_u until u is placed at p, see node v_7 in Figure 10. Since the stub of u for e also has an angle of $\alpha(u)/2$ with C_u, the arc a indeed realizes the edge e with the angles in both u and v_i required for perfect angular resolution. Furthermore, a does not enter the disk bounded by C_u and hence it does not intersect any part of the drawing of T_u other than u.

We can summarize our results for drawing the light children of a node as follows:

Lemma 4.3. *Let v be a node of T at level j of $H(T)$ with two incident heavy edges. For every light child $u \in L(v)$ assume there is a disk D_u of radius $r_u = 2 \cdot 4^{h(T)-j-1}|T_u|$ that contains a fixed drawing of T_u with perfect angular resolution and such that u is exposed in the outer annulus of D_u. Then we can construct a drawing of v and its light subtrees inside a disk D, potentially with an extended small zone, such that the following properties hold:*

1. *the edge between v and any light child $u \in L(v)$ is a circular arc that does not intersect any disk other than D_u;*
2. *the heavy edges do not intersect any disk D_u;*
3. *any two disks D_u and $D_{u'}$ for $u \neq u'$ are disjoint;*
4. *the angular resolution of v is $2\pi/d(v)$;*
5. *the disk D has radius $r_v = 4^{h(T)-j}l(v)$.*

By combining Lemmas 4.1 and 4.3 we obtain the following theorem:

Theorem 4.4. *Given an ordered tree T with n nodes we can find a crossing-free Lombardi drawing of T that preserves the embedding of T and fits inside a disk D of radius $2 \cdot 4^{h(T)}n$, where $h(T)$ is the height of the heavy-path decomposition of T. Since $h(T) \leq \log_2 n$ the radius of D is no more than $2n^3$.*

Figure 2b shows a drawing of an ordered tree according to our method. We note that instead of asking for perfect angular resolution, the same algorithm can be used to construct a circular-arc drawing of an ordered tree with any assignment of angles between consecutive edges around each node that add up to 2π. The drawing remains crossing-free and fits inside a disk of radius $O(n^3)$.

5 Conclusion and Closing Remarks

We have shown that straight-line drawings of trees can be performed with perfect angular resolution and polynomial area, by carefully ordering the children of each vertex and by using a style similar to balloon drawings in which the children of any vertex are placed on two concentric circles rather than on a single circle. However, using our Fibonacci caterpillar example we showed that this combination of straight lines, perfect angular resolution, and polynomial area could no longer be achieved if the children of each vertex may not be reordered. For trees with a fixed embedding, Lombardi drawings in which edges are drawn as circular arcs allow us to retain the other desirable qualities of polynomial area and perfect angular resolution. In the appendix we report on a basic implementation and some practical improvements of the straight-line drawing algorithm.

Our work opens up new problems in the study of Lombardi drawings of trees, but much remains to be done in this direction. In particular, our polynomial area bounds seem unlikely to be tight, and our method is impractically complex. It would be of interest to find simpler Lombardi drawing algorithms that achieve perfect angular resolution for more limited classes of trees, such as binary trees, with better area bounds.

Acknowledgments. This research was supported in part by the National Science Foundation under grant 0830403, by the Office of Naval Research under MURI grant N00014-08-1-1015, and by the German Research Foundation under grant NO 899/1-1.

References

[1] U. Brandes and D. Wagner. Using graph layout to visualize train interconnection data. *J. Graph Algorithms Appl.* 4(3):135–155, 2000, `http://jgaa.info/accepted/00/BrandesWagner00.4.3.pdf`.

[2] C. Buchheim, M. Jünger, and S. Leipert. Improving Walker's algorithm to run in linear time. *Proc. 10th Int. Symp. Graph Drawing (GD'02)*, pp. 344–353. Springer-Verlag, LNCS 2528, 2002, doi:10.1007/3-540-36151-0_32.

[3] J. Cappos, A. Estrella-Balderrama, J. J. Fowler, and S. G. Kobourov. Simultaneous graph embedding with bends and circular arcs. *Computational Geometry* 42(2):173–182, 2009, doi:10.1016/j.comgeo.2008.05.003.

[4] J. Carlson and D. Eppstein. Trees with convex faces and optimal angles. *Proc. 14th Int. Symp. Graph Drawing (GD'06)*, pp. 77–88. Springer-Verlag, LNCS 4372, 2007, doi:10.1007/978-3-540-70904-6_9, arXiv:cs.CG/0607113.

[5] T. Chan, M. T. Goodrich, S. R. Kosaraju, and R. Tamassia. Optimizing area and aspect ratio in straight-line orthogonal tree drawings. *Computational Geometry* 23(2):153–162, 2002, doi:10.1016/S0925-7721(01)00066-9.

[6] C. C. Cheng, C. A. Duncan, M. T. Goodrich, and S. G. Kobourov. Drawing planar graphs with circular arcs. *Discrete Comput. Geom.* 25(3):405–418, 2001, doi:10.1007/s004540010080.

[7] M. Dickerson, D. Eppstein, M. T. Goodrich, and J. Meng. Confluent drawings: visualizing non-planar diagrams in a planar way. *Proc. 11th Int. Symp. Graph Drawing (GD'03)*, pp. 1–12. Springer-Verlag, LNCS 2912, 2003, arXiv:cs.CG/0212046.

[8] C. A. Duncan, A. Efrat, S. G. Kobourov, and C. Wenk. Drawing with fat edges. *Int. J. Found. Comput. Sci.* 17(5):1143–1164, 2006, doi:10.1142/S0129054106004315.

[9] C. A. Duncan, D. Eppstein, M. T. Goodrich, S. G. Kobourov, and M. Nöllenburg. Lombardi drawings of graphs. *Proc. 18th Int. Symp. on Graph Drawing (GD 2010)*. Springer-Verlag, to appear.

[10] P. Eades. Drawing free trees. *Bull. Inst. Combinatorics and Its Applications* 5:10–36, 1992.

[11] B. Finkel and R. Tamassia. Curvilinear graph drawing using the force-directed method. *Proc. 12th Int. Symp. Graph Drawing (GD'04)*, pp. 448–453. Springer-Verlag, LNCS 3383, 2004.

[12] A. Garg, M. T. Goodrich, and R. Tamassia. Planar upward tree drawings with optimal area. *Int. J. Comput. Geom. Appl.* 6(3):333–356, 1996, `http://www.cs.brown.edu/cgc/papers/ggt-aoutd-96.ps.gz`.

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

[14] A. Garg and A. Rusu. Straight-line drawings of binary trees with linear area and arbitrary aspect ratio. *J. Graph Algorithms Appl.* 8(2):135–160, 2004, `http://jgaa.info/accepted/2004/GargRusu2004.8.2.pdf`.

[15] S. Grivet, D. Auber, J. P. Domenger, and G. Melançon. Bubble tree drawing algorithm. *Proc. Int. Conf. Computer Vision and Graphics*, pp. 633–641. Springer-Verlag, 2004, `http://www.labri.fr/publications/is/2004/GADM04`.

[16] D. Harel and R. E. Tarjan. Fast algorithms for finding nearest common ancestors. *SIAM J. Comput.* 13(2):338–355, 1984, doi:10.1137/0213024.

[17] R. Hobbs and M. Lombardi. *Mark Lombardi: Global Networks*. Independent Curators International, New York, 2003.

[18] C.-C. Lin and H.-C. Yen. On balloon drawings of rooted trees. *J. Graph Algorithms Appl.* 11(2):431–452, 2007, `http://jgaa.info/accepted/2007/LinYen2007.11.2.pdf`.

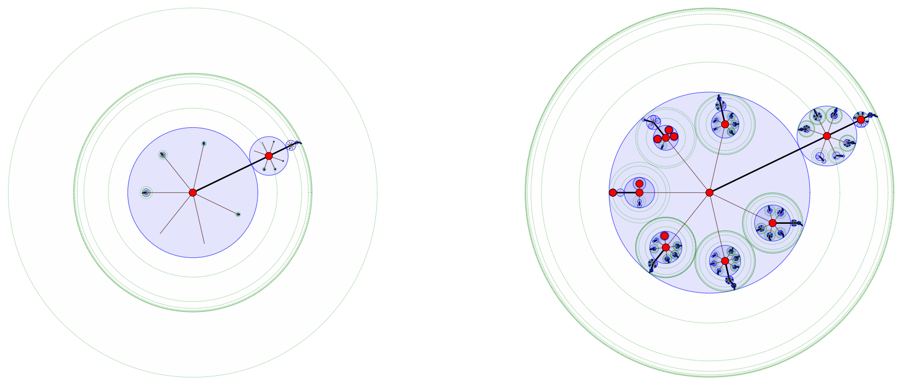
[19] G. Melançon and I. Herman. Circular Drawings of Rooted Trees. Tech. Rep. INS-R9817, CWI Amsterdam, 1998.

[20] E. M. Reingold and J. S. Tilford. Tidier drawings of trees. *IEEE Trans. Software Engineering* 7(2):223–228, 1981.

[21] C.-S. Shin, S. K. Kim, and K.-Y. Chwa. Area-efficient algorithms for straight-line tree drawings. *Computational Geometry* 15(4):175–202, 2000, doi:10.1016/S0925-7721(99)00053-X.

[22] J. Walker. A node-positioning algorithm for general trees. *Software Practice and Experience* 20(7):685–705, 1990, doi:10.1002/spe.4380200705.

[23] C. Wetherell and A. Shannon. Tidy drawings of trees. *IEEE Trans. Software Engineering* 5(5):514–520, 1979.



(a) The unmodified straight-line tree drawing algorithm. Though the area is bounded it is not quite desirable.

(b) A space-optimized drawing that still maintains the stated guarantees.

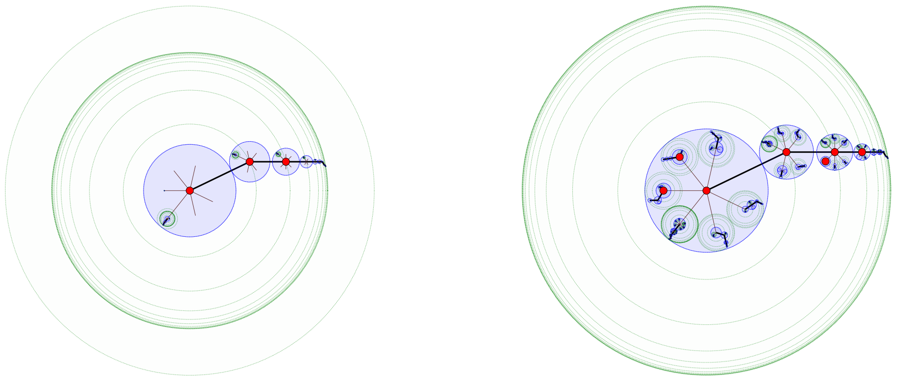
Fig. 13: A partial snapshot of a tree drawing.

A Implementation Details

Although theoretically interesting, tree drawings with perfect angular resolution are also of practical importance. To that end, we have implemented a basic version of our straight-line drawing algorithm. The algorithm, though polynomially bounded, from a practical viewpoint is still far from desirable. In particular, as Figure 13a illustrates, there is significant space left between sibling nodes as our algorithm essentially focuses on providing a guaranteed bound. As Fig 13b demonstrates, with some simple heuristical refinements, however, far better use of space can be achieved.

We highlight three key improvements that we made to the algorithm that do not affect the overall layout and so still provide the same guaranteed bound as in the regular algorithm with additional quite simple improvements in space efficiency.

- In the construction, only large nodes are placed on the outer region. The remaining small nodes are placed inside the inner annulus. There is no reason not to place further small nodes in the outer region as well. As a result, we continue with the greedy approach and repeatedly insert the next largest in the outer region, skipping the spoke associated with the heavy edge, until no more nodes fit. We fill the remaining spokes with the smaller children. We also note that in many cases, all children fit inside the outer region, as the largest light child nodes are often small enough to fit in one wedge region. Figure 14 illustrates the improvement.
- The radii for the light subtrees are increased to allow the disk to fit maximally within its wedge region. To ensure that the subtrees are still constrained initially to the primary layout algorithm, we defer the scaling of the nodes until after the layout has completed. This has the effect of using considerably more of the allocated space as demonstrated again by Figure 14.

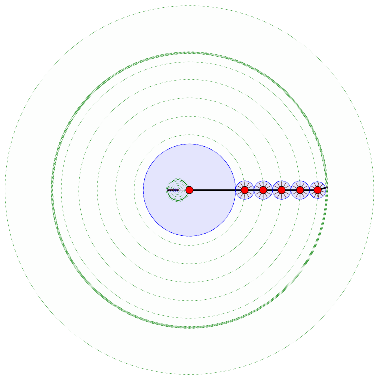


(a) A portion of an unmodified straight-line tree drawing algorithm that only placed large nodes on the outside annulus.

(b) The same tree but with space-filling optimization in place.

Fig. 14: A partial snapshot of a tree drawing demonstrating filling the disk associated with the light subtree.

- The heavy path does not completely fill the disk associated with its head node. As a result, we also increase this radius as a constant factor after having laid out the main drawing, see Figure 15.
- There were other improvements possible as well. One notable intentional omission was to use the heavy path breakdown for a subtree only if the entire subtree could not fit within the nodes' light-children radius. In many cases, the heavy path is small enough to still fit within this radius. We kept the path present to highlight the key feature in our algorithm that allows for the bounded area construction. The path itself could also be designed to use more of its underlying region, however, we do not see any easy way to do this effectively and avoid intersecting the path at a later point. Nonetheless, it is a promising area for space improvement.


(a) A portion of an unmodified straight-line tree drawing of a caterpillar-like tree.

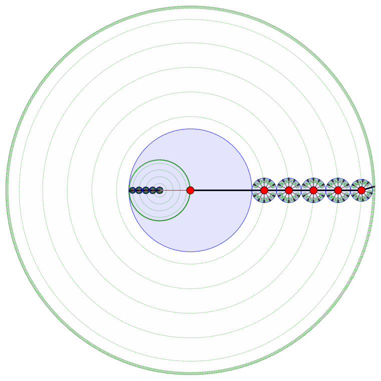
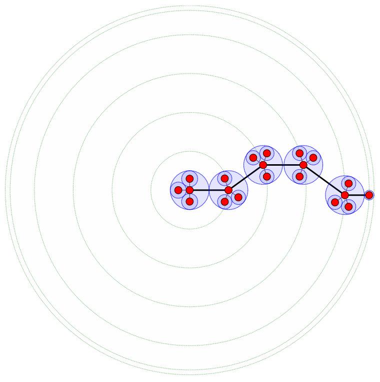

(b) The same tree but with space-filling optimization in place.

Fig. 15: A partial snapshot demonstrating expanding the heavy path to fit outer disk.


(a) The Fibonacci caterpillar drawn as an unordered tree.

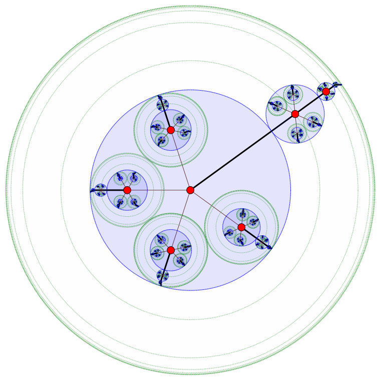

(b) A 5-ary tree with different weight distributions per child.

Fig. 16: Example illustrations.