Improved Bounds for Drawing Trees on Fixed Points with L-Shaped Edges

Therese Biedl[1](✉), Timothy M. Chan[2], Martin Derka[1], Kshitij Jain[1], and Anna Lubiw[1]

[1] University of Waterloo, Waterloo, ON N2L 3G1, Canada
{biedl,mderka,k22jain,alubiw}@uwaterloo.ca
[2] University of Illinois at Urbana-Champaign, Urbana, IL, USA
tmc@illinois.edu

Abstract. Let T be an n-node tree of maximum degree 4, and let P be a set of n points in the plane with no two points on the same horizontal or vertical line. It is an open question whether T always has a planar drawing on P such that each edge is drawn as an orthogonal path with one bend (an "L-shaped" edge). By giving new methods for drawing trees, we improve the bounds on the size of the point set P for which such drawings are possible to: $O(n^{1.55})$ for maximum degree 4 trees; $O(n^{1.22})$ for maximum degree 3 (binary) trees; and $O(n^{1.142})$ for perfect binary trees.

Drawing ordered trees with L-shaped edges is harder—we give an example that cannot be done and a bound of $O(n \log n)$ points for L-shaped drawings of ordered caterpillars, which contrasts with the known linear bound for unordered caterpillars.

1 Introduction

The problem of drawing a planar graph so that its vertices are restricted to a specified set of points in the plane has been well-studied, both from the perspective of algorithms and from the perspective of bounding the size of the point set and/or the number of bends needed to draw the edges. Throughout this paper we consider the version of the problem where the points are unlabelled, i.e., we may choose to place any vertex at any point.

One basic result is that every planar n-vertex graph has a planar drawing on any set of n points, even with the limitation of at most 2 bends per edge [11]. If every edge must be drawn as a straight-line segment then any n points in general position still suffice for drawing trees [4] and outerplanar graphs [3] but this result does not extend to any non-outerplanar graph [9], and the decision version of the problem becomes NP-complete [5]. Since n points do not always suffice, the next natural question is: How large must a *universal* point set be, and how many points are needed for *any* point set to be universal? An upper bound of $O(n^2)$ follows from the 1990 algorithms that draw planar graphs on an $O(n) \times O(n)$ grid [8,13], but the best known lower bound, dating from 1989, is $c \cdot n$ for some $c > 1$ [6].

Although orthogonal graph drawing has been studied for a long time, analogous questions of universal point sets for orthogonal drawings have only been explored recently, beginning with Katz et al. [10] in 2010. Since at most 4 edges can be incident to a vertex in an orthogonal drawing, attention is restricted to graphs of maximum degree 4. Throughout the paper we will restrict attention to point sets in "general orthogonal position" meaning that no two points share the same x- or y-coordinate. We study the simplest type of orthogonal drawings where every edge must be drawn as an orthogonal path of two segments. Such a path is called an "L-shaped edge" and these drawings are called "planar L-shaped drawings". Observe that any planar L-shaped drawing lives in the grid formed by the n horizontal and n vertical lines through the points.

Di Giacomo et al. [7] introduced the problem of planar L-shaped drawings and showed that any tree of maximum degree 4 has a planar L-shaped drawing on any set of $n^2 - 2n + 2$ points (in general orthogonal position, as will be assumed henceforth). Aichholzer et al. [1] improved the bound to $O(n^c)$ with $c = \log_2 3 \approx 1.585$. It is an open question whether n points always suffice. Surprisingly, nothing better is known for trees of maximum degree 3.

The largest subclass of trees for which n points are known to suffice is the class of caterpillars of maximum degree 3 [7]. A *caterpillar* is a tree such that deleting the leaves gives a path, called the *spine*. For caterpillars of maximum degree 4 with n nodes, any point set of size $3n - 2$ permits a planar L-shaped drawing [7], and the factor was improved to 5/3 by Scheucher [12].

1.1 Our Results

We give improved bounds as shown in Table 1. A tree of max degree 3 (or 4) is *perfect* if it is a rooted binary tree (or ternary tree, respectively) in which all leaves are at the same height and all non-leaf nodes have 2 (or 3, respectively) children. Our bounds are achieved by constructing the drawings recursively and analyzing the resulting recurrence relations, which is the same approach used previously by Aichholzer et al. [1]. Our improvements come from more elaborate drawing methods. Results on maximum degree 3 trees are in Sect. 3 and results on maximum degree 4 trees are in Sect. 4.

Table 1. Previous and new bounds on the number of points sufficient for planar L-shaped drawings of any tree of n nodes. The previous bounds all come from Aichholzer et al. [1].

	Previous	New
deg 3 perfect	$n^{1.585}$	$n^{1.142}$
deg 3 general	$n^{1.585}$	$n^{1.22}$
deg 4 perfect	$n^{1.465}$ [a]	
deg 4 general	$n^{1.585}$	$n^{1.55}$

[a]The bound of $n^{1.465}$ is not explicit in [1] but will be explained in Sect. 4.

We also consider the case of *ordered* trees where the cyclic order of edges incident to each vertex is specified. We give an example of an n-node ordered tree (in fact, a caterpillar) and a set of n points such that the tree has no L-shaped planar drawing on the point set. We also give a positive result about drawing some ordered caterpillars on $O(n \log n)$ points. The caterpillars that can be drawn on such $O(n \log n)$ points include our example that cannot be drawn on a given set of n points. These results are in Sect. 2.

1.2 Further Background

Katz et al. [10] introduced the problem of drawing a planar graph on a specified set of points in the plane so that each edge is an *orthogeodesic path*, i.e. a path of horizontal and vertical segments whose length is equal to the L_1 distance between the endpoints of the path. They showed that the problem of deciding whether an n-vertex planar graph has a planar orthogeodesic drawing on a given set of n points is NP-complete. Subsequently, Di Giacomo et al. [7] showed that any n-node tree of maximum degree 4 has an orthogeodesic drawing on any set of n points where the drawing is restricted to the $2n \times 2n$ grid that consists of the "basic" horizontal and vertical lines through the points, plus one extra line between each two consecutive parallel basic lines. If the drawing is restricted to the basic grid, their bounds were $4n - 3$ points for degree-4 trees, and $3n/2$ points for degree-3 trees. These bounds were improved by Scheucher [12] and then by Bárány et al. [2].

2 Ordered Trees—The Case of Caterpillars

All previous work has assumed that trees are unordered, i.e., that we may freely choose the cyclic order of edges incident to a vertex. In this section we show that ordered trees on n vertices do not always have planar L-shaped drawings on n points. Our counterexample is a *top-view caterpillar*, i.e., a caterpillar such that the two leaves adjacent to each vertex lie on opposite sides of the spine. The main result in this section is that every top-view caterpillar has a planar L-shaped drawing on $cn \log n$ points for some $c > 0$.

First the counterexample. We prove the following in the full version:

Lemma 1. *The top-view caterpillar C_{14} on $n = 14$ nodes shown in Fig. 1(a) cannot be drawn with L-shaped edges on the point set P_{14} of size 14 shown in Fig. 1(c).*

It is conceivable that this counter-example is an isolated one—we have been unable to extend it to a family of such examples.

Next we explore the question of how many points are needed for a planar L-shaped drawing of an n-vertex top-view caterpillar. Consider the appearance of the caterpillar's spine (a path) in such a drawing. Each node of the spine, except for the two endpoints, must have its two incident spine edges aligned—both horizontal or both vertical. Define a *straight-through* drawing of a path to

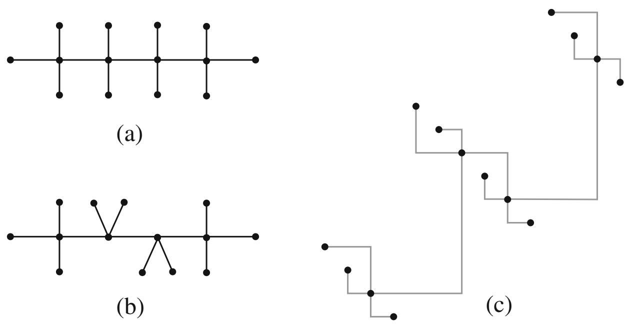

Fig. 1. The ordered top-view caterpillar C_{14} shown in (a) does not have a planar L-shaped drawing on the point set P_{14} shown in (c). The ordering shown in (b) does.

be a planar L-shaped drawing such that the two incident edges at each vertex are aligned. The best bound we have for the number of points that suffice for a straight-through drawing of a path is obtained when we draw the path in a monotone fashion, i.e. with non-decreasing x-coordinates.

Theorem 1. *Any path of n vertices has an x-monotone straight-through drawing on any set of at least $c \cdot n \log n$ points for some constant c.*

Proof. We prove that if the number of points satisfies the recurrence $M(n) = 2M(\frac{n}{2}) + 2n$ then any path of n vertices has an x-monotone straight-through drawing on the points. Observe that this recurrence relation solves to $M(n) \in \Theta(n \log n)$ which will complete the proof. Within a constant factor we can assume without loss of generality that n is a power of 2.

Order the points by x-coordinate. Recall our assumption that no two points share the same x- or y-coordinate. By induction, the first half of the path has an x-monotone straight-through drawing on the first $M(\frac{n}{2})$ points. We add the assumption that the path starts with a horizontal segment.

Let p be the second last point used. Since n is a power of 2, the path goes through p on a horizontal segment. Let T be the set of points to the right of and above p. Let B be the set of points to the right of and below p. Refer to Fig. 2(a). In T, consider the partial order $(x_1, y_1) \prec_T (x_2, y_2)$ if $x_1 < x_2$ and $y_1 < y_2$. Let T' be the set of minimal elements in this partial order. Similarly, in B, let B' be the set of elements that are minimal in the ordering $(x_1, y_1) \prec_B (x_2, y_2)$ if $x_1 < x_2$ and $y_1 > y_2$. If T' has n or more points, then we can draw the whole path on T' with an x-monotone straight-through drawing starting with a horizontal segment. The same holds if B' has n or more points. Thus we may assume that $|T'|, |B'| < n$. We now remove T' and B'; let $R = (T - T') \cup (B - B')$. Then $|R| \geq M(\frac{n}{2})$.

By induction the second half of the path has an x-monotone straight-through drawing on the set R starting with a horizontal segment. Let r be the first point used for this drawing. Assume without loss of generality that r lies in T. (The other case is symmetric.) Consider the rectangle with opposite corners at p and

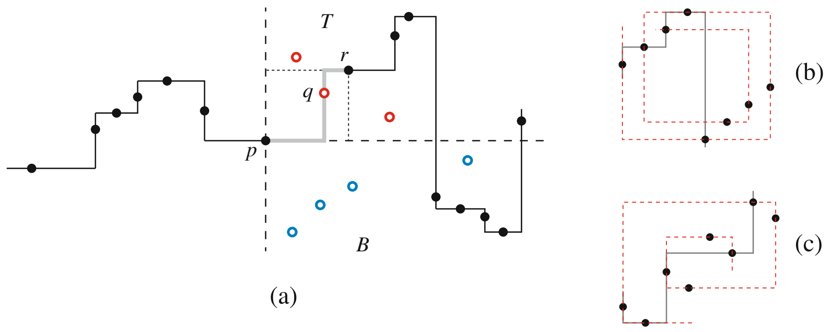

Fig. 2. (a) The construction for the proof of Theorem 1. The points of T' and B' are drawn as hollow red points above p and hollow blue points below p, respectively. (b-c) Examples of point sets of size $2n$ for which the maximum length of an x-monotone straight-through path is $n+1$. Such paths are shown in grey. In both cases there are non-monotone planar straight-through paths of length $2n$ (dashed). (Color figure online)

r. Since r is not in T', there is a point $q \in T$ inside the rectangle. We can join the two half paths using a vertical segment through q and the last vertex of the first half path is embedded at q. □

We can extend the above result to draw the entire caterpillar (not just its spine) with the same bound on the number of points:

Theorem 2. *Any top-view caterpillar of n vertices has a planar L-shaped drawing on any set of $c \cdot n \log n$ points for some constant c.*

Proof (outline). Follow the above construction, but in addition to T' and B', also take the second and third layers. If any layer has n or more points, we embed the whole caterpillar on it [7]. Otherwise, we remove at most a linear number of points, and embed the second half of the caterpillar by induction on the remaining points. Then, in the rectangle between p and r there must be an increasing sequence of 3 points. Use the middle one for the left-over spine-vertex q and the other two for the leaves of q. □

We conjecture that $2n$ points suffice for an x-monotone straight-through drawing of any n-path. See Fig. 2(b-c) for a lower bound of $2n$. Do n points suffice if the x-monotone condition is relaxed to planarity? Finally, we mention that the problem of finding monotone straight-through paths is related to a problem about alternating runs in a sequence, as explained in the full paper.

3 Trees of Maximum Degree 3

In this section, we prove bounds on the number of points needed for L-shaped drawings of trees with maximum degree 3. We treat the trees as rooted and thus, we refer to them as binary trees. We name the parts of the tree as shown in Fig. 3(a). The root r_0 has two subtrees T_1 and T_2 of size n_1 and n_2, respectively, with $n_1 \leq n_2$. T_2's root, r_1, has subtrees of sizes $n_{2,1}$ and $n_{2,2}$ with $n_{2,1} \leq n_{2,2}$.

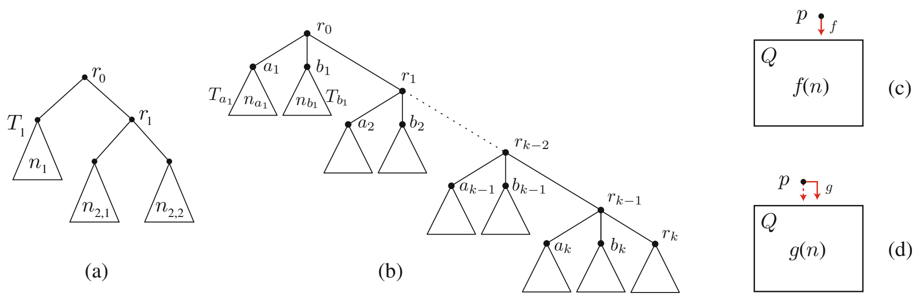

Fig. 3. The naming conventions for (a) binary and (b) ternary trees. The set-up for (c) f-configurations and (d) g-configurations.

The main idea is to draw a tree T on a set of points in a rectangle Q by partitioning the rectangle into subrectangles in which we recursively draw subtrees. This gives rise to recurrence relations for the number of points needed to draw trees of size n, which we then analyze. We distinguish two subproblems or "configurations." In each, we must draw a tree T rooted at r_0 in a rectangle Q that currently has no part of the drawing inside it. Furthermore, the parent p of r_0 has already been drawn, and one or two rays outgoing from p have been reserved for drawing the first segment of edge (p, r_0) (without hitting any previous part of the drawing).

In the *f-configuration* the reserved ray from p goes vertically downward to Q. See Fig. 3(c). Let $f(n)$ be the smallest number of points such that any binary tree with n vertices can be drawn in any rectangle with $f(n)-1$ points in the f-configuration[1]. We will give various ways of drawing trees in the f-configuration, each of which gives an upper bound on $f(n)$. Among these choices, the algorithm uses the one that requires the fewest points.

In the *g-configuration* we reserve a horizontal ray from p, that allows the L-shaped edge (p, r_0) to turn downward into Q at any point without hitting any previous part of the drawing. In addition, we reserve the vertical ray downward from p in case this ray enters Q. See Fig. 3(d) for the case where the horizontal ray goes to the right. Let $g(n)$ be the smallest number of points such that any binary tree with n vertices can be drawn in any rectangle with $g(n) - 1$ points in the g-configuration. Observe that $f(n) \geq g(n)$ since the g-configuration gives us strictly more freedom.

We start with two easy constructions to give the flavour of our methods.

***f*-draw-1.** This method, illustrated in Fig. 4(a), applies to an f-configuration. We first describe the construction and then say how many points are required.

[1] Beware: we will use the same notation $f(n)$ in Sect. 4 to refer to ternary trees.

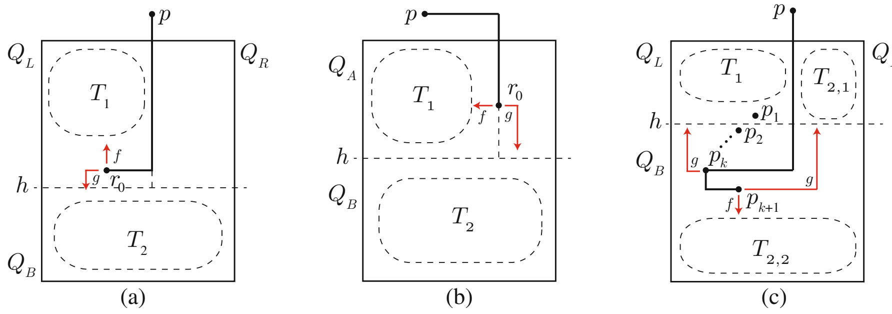

Fig. 4. Three methods: (a) f-draw-1; (b) g-draw; and (c) f-draw-2.

Continue the vertical ray from p downward to a horizontal half-grid line h determined as follows. Partition Q by h and the ray down to h into 3 parts: Q_B, the rectangle below h; Q_L, the upper left rectangle; and Q_R, the upper right rectangle. Choose h to be the highest half-grid line such that Q_L or Q_R has $f(n_1)$ points. Without loss of generality, assume that Q_L has $f(n_1)$ points, and Q_R has at most $f(n_1)$ points. Place r_0 at the bottommost point of Q_L. Draw the edge (p, r_0) down and left. Start a ray vertically up from r_0, and recursively draw T_1 in f-configuration (rotated 180°) in the subrectangle of Q_L above r_0, which has $f(n_1) - 1$ points. This leaves the leftward and downward rays free at r_0, so we can draw T_2 recursively in g-configuration in Q_B so long as there are $g(n_2) - 1$ points. The total number of points required is $2f(n_1) + g(n_2) - 1$. Recall that $f(n)$ is 1 more than the number of required points, so this proves:

$$f(n) \leq 2f(n_1) + g(n_2). \qquad (f\text{-}1)$$

Observe that we could have swapped f and g which proves:

$$f(n) \leq 2g(n_1) + f(n_2). \qquad (f\text{-}1')$$

The above method can be viewed as a special case of Aichholzer et al.'s method for ternary trees [1] (see Sect. 4). We incorporate two new ideas to improve their result: first, they used only f-configurations, but we notice that one of the above two recursive subproblems is a g-configuration in the binary tree case, and can be solved by a better recursive algorithm; second, their method wasted all the points in Q_R, but we will give more involved constructions that allow us to use some of those points.

g-**draw.** This method applies to a g-configuration where the ray from the parent node p goes to the right. Partition Q at the highest horizontal half-grid line such that the top rectangle Q_A has $f(n_1)$ points. We separate into two cases depending whether the rightmost point, q, of Q_A is to the right or left of p.

If q is to the right of p, place r_0 at q, and draw the edge (p, r_0) right and down. See Fig. 4(b). Start a ray leftward from r_0 and recursively draw T_1 in f-configuration in the subrectangle of Q_A to the left of q. Note that there are $f(n_1) - 1$ points here, which is sufficient. The rightward and downward rays at r_0 are free, so we can draw T_2 recursively in g-configuration in Q_B if there are $g(n_2) - 1$ points. The total number of points required is $f(n_1) + g(n_2) - 1$.

If all points of Q_A lie to the left of p, then place r_0 at the bottommost point of Q_A and observe that we now have the situation of f-draw-1 with Q_R empty, and $f(n_1) + g(n_2) - 1$ points suffice.

This proves:

$$g(n) \leq f(n_1) + g(n_2). \qquad (g)$$

We now describe a different f-drawing method that is more efficient than f-draw-1 above, and will be the key for our bound for binary trees.

f-draw-2. This method applies to an f-configuration. We begin as in f-draw-1, though with the f-drawing and the g-drawing switched. Partition Q by a horizontal half-grid line h and the ray from p down to h into 3 parts: Q_B, the rectangle below h; Q_L, the upper left rectangle; and Q_R, the upper right rectangle. Choose h to be the highest half-grid line such that Q_L or Q_R has $g(n_1)$ points. Without loss of generality, assume the former. We separate into two cases depending on the size of Q_R.

If $|Q_R| < g(n_{2,1})$ then we follow the f-draw-1 method. Let p_1 be the bottommost point of Q_L. Place r_0 at p_1, draw the edge (p, r_0) down and left, recursively draw T_1 in g-configuration in Q_L using leftward/upward rays from r_0, and recursively draw T_2 in f-configuration in Q_B using a downward ray from r_0. This requires $g(n_1) + g(n_{2,1}) + f(n_2) - 1$ points, where $g(n_{2,1})$ accounts for the wasted points in Q_R.

If $|Q_R| \geq g(n_{2,1})$ then we make use of the points in Q_R by drawing subtree $T_{2,1}$ there. Let p_1 be the bottommost point of Q_L, and let $p_2, p_3, \ldots$ be the points of Q_B below p_1 in decreasing y-order. Let $k \geq 2$ be the smallest index such that either $k = n$ or point p_{k+1} lies to the right of p_k. See Fig. 4(c).

We have two subcases. If $k = n$, then $p_1, \ldots, p_k$ form a monotone chain of length n, i.e., a diagonal point set in the terminology of Di Giacomo et al. [7]. They showed that any tree of n points can be embedded on a diagonal point set, so we simply draw T on these n points. (Note that if this construction is used in the induction step, upward visibility is needed for connecting T to the rest of the tree, and this can be achieved.)

Otherwise $k < n$. Place r_0 at point p_k and r_1 at p_{k+1}. Draw the edge (p, r_0) down and left, and the edge (r_0, r_1) down and right. Recursively draw T_1 in g-configuration in Q_L using leftward/upward rays from r_0. Draw $T_{2,2}$ in f-configuration in the rectangle below r_1 using a downward ray from r_1. Draw $T_{2,1}$ in g-configuration in Q_R using the rightward ray from r_1. Observe that if r_1 lies to the right of p (i.e., below Q_R rather than below Q_L) then the upward ray from r_1 is clear (as required for a g-drawing). The number of points required is at most $2g(n_1) + n + f(n_{2,2}) - 1$. This accounts for at most $g(n_1)$ points in Q_R, and at most n points below h and above r_1.

This proves:

$$f(n) \leq \max\{g(n_1) + g(n_{2,1}) + f(n_2),\ 2g(n_1) + f(n_{2,2}) + n\}. \qquad (f\text{-}2)$$

Theorem 3. *Any perfect binary tree with n nodes has an L-shaped drawing on any point set of size $c \cdot n^{1.142}$ for some constant c.*

Proof. For perfect binary trees we have $n_1 = n_2 = \frac{1}{2}n$ and $n_{2,1} = n_{2,2} = \frac{1}{4}n$. We solve the simultaneous recurrence relations for f and g in the full paper. □

Theorem 4. *Any binary tree has an L-shaped drawing on any point set of size $c \cdot n^{1.22}$ for some constant c.*

Proof. For $n_1 \leq 0.349n$, we use recursion (f-1). For $n_{2,1} \leq 0.082n$, we combine recursion (f-1') and (f-1) to obtain $f(n) \leq 2g(n_1) + 2f(n_{2,1}) + g(n_{2,2})$. For $n_1 > 0.349n$ and $n_{2,1} > 0.082n$, we use recursion (f-2). We solve the simultaneous recurrence relations for f and g in the full paper. □

4 Trees of Maximum Degree 4

In this section, we prove bounds on the number of points needed for L-shaped drawings of trees with maximum degree 4. We treat the trees as rooted and refer to them as ternary trees. Given a ternary tree of n nodes, let a_1, b_1 and r_1 be the three children of the root r_0. We use T_v to denote the subtree rooted at a node v, and n_v to denote the number of nodes in T_v. We name the children of the root such that $n_{a_1} \leq n_{b_1} \leq n_{r_1}$. For $i \geq 2$, let a_i, b_i, r_i be the three children of r_{i-1}, named such that $n_{a_i} \leq n_{b_i} \leq n_{r_i}$. See Fig. 3(b).

We will draw ternary trees using only the f-configuration as defined in Sect. 3 (see Fig. 3(c)). In this section (as opposed to the previous one) we define $f(n)$ to be minimum number such that any ternary tree of n nodes can be drawn in f-configuration on any set of $f(n) - 1$ points.

As in Sect. 3, we will give various drawing methods, each of which gives a recurrence relation for $f(n)$. We begin with a re-description of Aichholzer et al.'s method [1].

f_4-draw-1. Extend the vertical ray from p downward to a horizontal half-grid line h determined as follows. Partition Q by h and the ray down to h into 3 parts: Q_B, the rectangle below h; Q_L, the upper left rectangle; and Q_R, the upper right rectangle. Choose h to be the highest half-grid line such that Q_L or Q_R has $2f(n_{a_1}) + 2f(n_{b_1})$ points. Without loss of generality, assume the former. Partition Q_L vertically into two rectangles Q_{LL} and Q_{LR} with at least $f(n_{a_1})$ points on the left and at least $f(n_{b_1})$ points on the right respectively, with Q_{LL} to the left of Q_{LR}. Place r_0 at the bottommost point in Q_{LR}. Extend a ray upward from r_0 and recursively draw T_{b_1} on the remaining $f(n_{b_1}) - 1$ points in Q_{LR}. Extend a ray to the left from r_0 and recursively draw T_{a_1} on the $f(n_{a_1})$ points in Q_{LL}. Finally, extend a ray downward from r_0 and recursively draw T_{r_1} in Q_B. See Fig. 5(a). The number of points required is $2f(n_{a_1}) + 2f(n_{b_1}) + f(n_{r_1}) - 1$, so this proves:

$$f(n) \le 2f(n_{a_1}) + 2f(n_{b_1}) + f(n_{r_1}). \qquad (f_4\text{-}1)$$

For example, in the case when T is perfect (with $n_{a_1} = n_{b_1} = n_{r_1} = \frac{n}{3}$), the inequality ($f_4$-1) becomes $f(n) \le 5f(n/3)$, which resolves to $O(n^{\log_3 5})$ and $\log_3 5 \approx 1.465$. The critical case for this recursion, though, turns out to be when $n_{a_1} = 0$ and $n_{b_1} = n_{r_1} = \frac{n}{2}$, which gives $f(n) \le 3f(n/2)$ and leads to Aichholzer et al.'s $O(n^{\log_3 2})$ result.

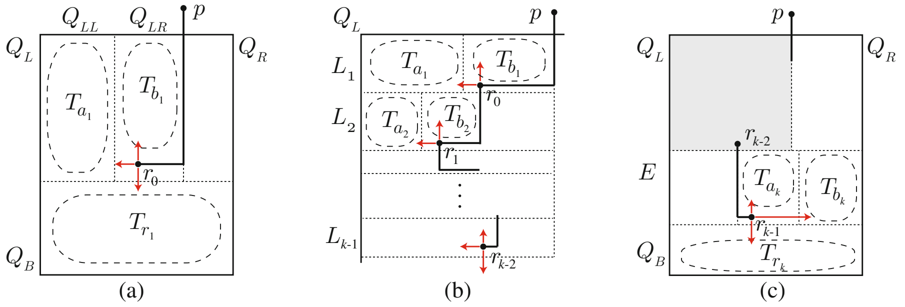

Fig. 5. (a) f_4-draw-1. (b) Drawing the "small" subtrees in Q_L. (c) f_4-draw-2A.

f_4**-draw-2.** To improve their result, our idea again is to avoid wasting the points in Q_R, and use some of those points in the recursive drawings of subtrees at the next levels. However, simply considering subtrees at the second level is not sufficient for an asymptotic improvement if T_{a_2} and T_{b_2} are too small. Thus, we consider a more complicated approach that stops at the first level $k \ge 2$ where $n_{r_k} \le 0.9n_{r_{k-1}}$. Note that for $i = 2, \ldots, k-1$, we have $n_{r_i} > 0.9n_{r_{i-1}}$ and $n_{a_i}, n_{b_i} \le 0.1n_{r_{i-1}}$, and so T_{a_i} and T_{b_i} are "small" subtrees. We apply the same idea as above to draw not just T_{a_1}, T_{b_1} but also all the small subtrees T_{a_i} and T_{b_i}, $i = 2, \ldots, k-1$ in Q_L (appropriately defined), and then consider a few cases for how to draw the remaining "big" subtrees T_{a_k}, T_{b_k}, and T_{r_k}, possibly using some points in Q_R. The number of points we will need to reserve for drawing $T_{a_1}, T_{b_1}, \ldots, T_{a_{k-1}}, T_{b_{k-1}}$ is

$$Y = f(n_{a_1}) + f(n_{b_1}) + \sum_{i=2}^{k-1}(2f(n_{a_i}) + 2f(n_{b_i})).$$

Extend the vertical ray from p downward until Q_L or Q_R has Y points. Without loss of generality, assume the former.

Drawing the Small Subtrees. We draw nodes r_i and subtrees T_{a_i} and T_{b_i}, $i = 1, \ldots, k-1$ in Q_L as follows. Split Q_L horizontally into rectangles $L_1, \ldots, L_{k-1}$. The plan is to draw r_i, T_{a_i} and T_{b_i} in L_i, in the same way that T_{a_1} and T_{b_1} were drawn in f_4-draw-1. See Fig. 5(b). Level L_1 is special because the vertical ray from p is at the right boundary of L_1. Thus, we require $f(n_{a_1}) + f(n_{b_1})$ points. For levels L_i, $i = 2, \ldots, k-1$ the vertical ray from r_{i-2} may enter L_i at any point, so we require $2f(n_{a_i}) + 2f(n_{b_i})$ points to follow the plan of f_4-draw-1, and the L-shaped edge from r_{i-2} to r_{i-1} may turn left or right. The total number of points we need in all levels is Y, which is why we defined Y as we did.

Drawing the Final Three Subtrees. It remains to draw r_{k-1} and its three subtrees T_{a_k}, T_{b_k}, and T_{r_k}. We will draw T_{r_k} on the bottommost $f(n_{r_k}) - 1$ points of Q. Call this rectangle Q_B. Let E be the "equatorial zone" that lies between Q_L, Q_R above and Q_B below. See Fig. 5(c). If we are lucky, then not too many points are wasted in Q_R. Let $Z \leq Y$ be the number of points in Q_R.

Case A: $Z < f(n_{b_k})$. In this case we draw r_{k-1}, T_{a_k} and T_{b_k} in E as in f_4-draw-1. See Fig. 5(c). For this, we need $2f(n_{a_k}) + 2f(n_{b_k})$ points in E. The total number of points required in this case is $Y + Z + 2f(n_{a_k}) + 2f(n_{b_k}) + f(n_{r_k}) - 1$, so this proves:

$$\begin{aligned} f(n) &\leq Y + Z + 2f(n_{a_k}) + 2f(n_{b_k}) + f(n_{r_k}) && (f_4\text{-2A}) \\ &= f(n_{a_1}) + f(n_{b_1}) + \sum_{i=2}^{k-1} \left(2f(n_{a_i}) + 2f(n_{b_i})\right) + 2f(n_{a_k}) + 3f(n_{b_k}) + f(n_{r_k}). \end{aligned}$$

We must now deal with the unlucky case when $Z \geq f(n_{b_k})$. We will require $3f(n_{a_k}) + f(n_{b_k})$ points in E. We sum up the total number of points below, but first we describe how to complete the drawing in E. Partition E into three regions: E_L, E_M, and E_R, where E_L is the region to the left of r_{k-2}, E_R is the region to the right of p, and E_M is the region between them. See Fig. 6. Observe that either $|E_L| \geq f(n_{a_k}) + f(n_{b_k})$, or $|E_M| \geq f(n_{a_k})$, or $|E_R| > f(n_{a_k})$. We show how to draw r_{k-1}, T_{a_k} and T_{b_k} in each of these 3 cases.

Case B1: $|E_L| \geq f(n_{a_k}) + f(n_{b_k})$. In this case we draw r_{k-1}, T_{a_k} and T_{b_k} in E_L as in f_4-draw-1. See Fig. 6(a). Since E_L is to the left of the ray down from r_{k-2}, we have sufficiently many points.

Case B2: $|E_M| \geq f(n_{a_k})$. In this case we place r_{k-1} at the lowest point of E_M, draw T_{a_k} above it in E_M, and T_{b_k} to its right in $Q_R \cup E_R$. See Fig. 6(b). Since $|Q_R| = Z \geq f(n_{b_k})$, we have enough points to do this.

Case B3: $|E_R| > f(n_{a_k})$. In this case we place r_{k-1} at the leftmost point of E_R, draw T_{a_k} to its right in E_R and T_{b_k} above it in Q_R. See Fig. 6(c). Again, there are sufficiently many points.

The total number of points required in each of these three cases is $Y + Z + 3f(n_{a_k}) + f(n_{b_k}) + f(n_{r_k}) - 1$, and $Y \leq Z$ which yields:

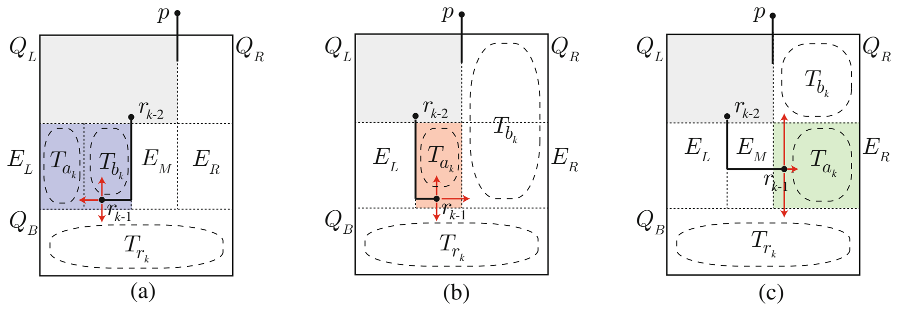

Fig. 6. The drawings for f_4-draw-2B: (a) Case B1, with E_L in blue; (b) Case B2, with E_M in red; (c) Case B3, with E_R in green. (Color figure online)

$$\begin{aligned} f(n) &\le Y + Z + 3f(n_{a_k}) + f(n_{b_k}) + f(n_{r_k}) \qquad (f_4\text{-2B}) \\ &= 2f(n_{a_1}) + 2f(n_{b_1}) + \sum_{i=2}^{k-1}(4f(n_{a_i}) + 4f(n_{b_i})) + 3f(n_{a_k}) + f(n_{b_k}) + f(n_{r_k}). \end{aligned}$$

The bound on $f(n)$ obtained from f_4-draw-2 is the maximum of (f_4-2A) and (f_4-2B).

Theorem 5. *Any ternary tree with n nodes has an L-shaped drawing on any point set of size $2n^{1.55}$.*

Proof. For $n_{b_1} \le 0.47n$, we use recursion (f_4-1). Otherwise, we use (f_4-2A) or (f_4-2B) and take the larger of the two bounds. We solve the recurrence relation for f in the full paper. □

5 Conclusions

We have made slight improvements on the exponent t in the bounds that $c \cdot n^t$ points always suffice for drawing trees of maximum degree 4, or 3, with L-shaped edges. Improving the bounds to, e.g., $O(n \log n)$ will require a breakthrough. In the other direction, there is still no counterexample to the possibility that n points suffice.

We introduced the problem of drawing ordered trees with L-shaped edges, where many questions remain open. For example: Do $c \cdot n$ points suffice for drawing ordered caterpillars? Can our isolated example be expanded to prove that n points are not sufficient in general?

Acknowledgments. We thank Jeffrey Shallit for investigating the alternating sequences discussed in Sect. 2. This work was done as part of a Problem Session in the Algorithms and Complexity group at the University of Waterloo. We thank the other participants for helpful discussions.

References

1. Aichholzer, O., Hackl, T., Scheucher, M.: Planar L-shaped point set embeddings of trees. In: European Workshop on Computational Geometry (EuroCG) (2016). http://www.eurocg2016.usi.ch/
2. Bárány, I., Buchin, K., Hoffmann, M., Liebenau, A.: An improved bound for orthogeodesic point set embeddings of trees. In: European Workshop on Computational Geometry (EuroCG) (2016). http://www.eurocg2016.usi.ch/
3. Bose, P.: On embedding an outer-planar graph in a point set. Comput. Geom. **23**(3), 303–312 (2002)
4. Bose, P., McAllister, M., Snoeyink, J.: Optimal algorithms to embed trees in a point set. J. Graph Algorithms Appl. **1**(2), 1–15 (1997)
5. Cabello, S.: Planar embeddability of the vertices of a graph using a fixed point set is NP-hard. J. Graph Algorithms Appl. **10**(2), 353–363 (2006)
6. Chrobak, M., Karloff, H.: A lower bound on the size of universal sets for planar graphs. ACM SIGACT News **20**(4), 83–86 (1989)
7. Di Giacomo, E., Frati, F., Fulek, R., Grilli, L., Krug, M.: Orthogeodesic point-set embedding of trees. Comput. Geom. **46**(8), 929–944 (2013)
8. Fraysseix, H.D., Pach, J., Pollack, R.: How to draw a planar graph on a grid. Combinatorica **10**(1), 41–51 (1990)
9. Gritzmann, P., Mohar, B., Pach, J., Pollack, R.: Embedding a planar triangulation with vertices at specified points. Am. Math. Monthly **98**, 165–166 (1991)
10. Katz, B., Krug, M., Rutter, I., Wolff, A.: Manhattan-geodesic embedding of planar graphs. In: Eppstein, D., Gansner, E.R. (eds.) GD 2009. LNCS, vol. 5849, pp. 207–218. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-11805-0_21
11. Kaufmann, M., Wiese, R.: Embedding vertices at points: few bends suffice for planar graphs. J. Graph Algorithms Appl. **6**(1), 115–129 (2002)
12. Scheucher, M.: Orthogeodesic point set embeddings of outerplanar graphs. Master's thesis, Graz University of Technology (2015)
13. Schnyder, W.: Embedding planar graphs on the grid. In: Proceedings of the First ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 138–148 (1990)