On Upward Drawings of Trees on a Given Grid

Therese Biedl[1(✉)] and Debajyoti Mondal[2]

[1] Cheriton School of Computer Science, University of Waterloo, Waterloo, ON, Canada
biedl@uwaterloo.ca

[2] Department of Computer Science, University of Saskatchewan, Saskatoon, SK, Canada
dmondal@cs.usask.ca

Abstract. Computing a minimum-area planar straight-line drawing of a graph is known to be NP-hard for planar graphs, even when restricted to outerplanar graphs. However, the complexity question is open for trees. Only a few hardness results are known for straight-line drawings of trees under various restrictions such as edge length or slope constraints. On the other hand, there exist polynomial-time algorithms for computing minimum-width (resp., minimum-height) upward drawings of trees, where the height (resp., width) is unbounded.

In this paper we take a major step in understanding the complexity of the area minimization problem for strictly-upward drawings of trees, which is one of the most common styles for drawing rooted trees. We prove that given a rooted tree T and a $W \times H$ grid, it is NP-hard to decide whether T admits a strictly-upward (unordered) drawing in the given grid. The hardness result holds both in polyline and straight-line drawing settings.

1 Introduction

Drawing planar graphs on a small integer grid is an active research area [17], which is motivated by various practical needs such as for VLSI circuit layout and small-screen visualization. Trees are one of the most studied graph classes in this context. While computing a minimum-area planar straight-line drawing of an arbitrary planar graph is known to be NP-complete [11], even for planar graphs with bounded pathwidth [11] and outerplanar graphs [4], the problem seems very intriguing for trees.

In this paper we examine *rooted and unordered trees*, i.e., one of the vertices is designated as the root and the left to right order of the children can be chosen arbitrarily. A natural way to display such a tree is to compute a *(strictly) upward* drawing, where each vertex is mapped to an integer grid point such that the parents have (strictly) larger y-coordinates than their children, each edge is drawn with y-monotone polylines with bends at integer grid points, and no two edges cross except possibly at their common endpoint. The *width, height* and *area*

Work of the authors supported in part by NSERC.

of the drawing are respectively the width, height, and area of the smallest axis-parallel rectangle that encloses the drawing. In a *straight-line (strictly) upward drawing*, the edges are restricted to be straight line segments.

We refer the reader to [17, Chapt. 5] or [8] for a survey on small-area drawings of trees. Here we review only the results that focus on exact minimization. In the fixed-embedding setting, there exist polynomial-time algorithms to compute minimum-area drawings for certain classes of planar graphs [4,15], namely, those that simultaneously have bounded treewidth and bounded face-degrees. On the other hand, the problem becomes NP-hard as soon as one of the above constraints is dropped [4]. The intractability of minimum-area tree drawings has been well established under some edge length and slope restrictions, e.g., when edges must be drawn with unit length or the slopes of the edges in the drawing must belong to one of the $(k/2)$ slopes determined by a k-grid. Note that an orthogonal grid is a 4-grid. Determining whether a tree can be drawn on a given k-grid, where $k \in \{4, 6, 8\}$ with edges of unit length is an NP-complete problem [2,3, 10]. Similar hardness results hold also for ordered trees under various aesthetic requirements [7].

Not much is known about minimum-area upward drawings of trees. Trevisan [18] showed that every complete tree (resp., Fibonacci tree) with n vertices admit a strictly-upward straight-line drawing in $n + O(\log n\sqrt{n})$ (resp., $1.17n + +O(\log n\sqrt{n})$) area, and conjectured that exact minimization may be possible in polynomial time. Trevisan mentioned that the problem of minimum-area strictly-upward drawing of a complete tree is a 'sparse problem'. Therefore, proving this NP-hard would imply $P = NP$ by Mahaney's theorem [12].

Interestingly, there exist polynomial-time algorithms for computing minimum-width upward drawings of trees, where the height is unbounded [5], and minimum-height drawings where the width is unbounded [1] (even for non-upward drawings [14]). Marriott and Stuckey [13] showed that minimizing the width of a strictly-upward straight-line drawing is NP-hard under the additional constraint that the x-coordinate of a parent is the average of the x-coordinates of its children. This additional requirement implies that a vertex with a single child must be placed directly above the child. Minimizing the width is known to be NP-hard even for ordered tree drawings under several other constraints [16].

We show that computing a strictly-upward drawing of a tree that resides in a given $W \times H$-grid is NP-hard. Formally, we consider the following problem:

Problem: Strictly-Upward Tree Drawing (SUTD)
Instance: An unordered rooted tree T, and two natural numbers W and H.
Question: Does T admit a strictly-upward (polyline or straight-line) drawing with width at most W and height at most H?

2 NP-Hardness

We prove the NP-hardness of SUTD by a reduction from the following problem:

Problem: Numerical 3-Dimensional Matching (N3DM)
Instance: Positive integers r_i, g_i, b_i, where $1 \leq i \leq k$, and an integer B such that $\sum_{i=1}^{k}(r_i + b_i + g_i) = k \cdot B$.
Question: Do there exist permutations π and π' of $\{1, \ldots, k\}$ such that $r_{\pi(i)} + b_i + g_{\pi'(i)} = B$ for all $1 \leq i \leq k$?

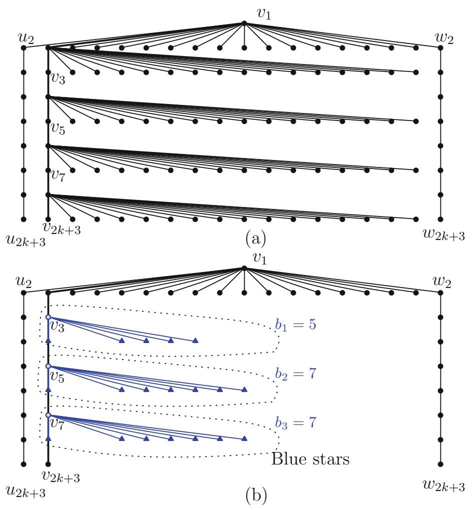
N3DM is strongly NP-complete [9], and remains NP-complete even if we require all b_i's to be odd, and the b_i's are large and the g_i's are huge relative to the r_i's. (More precisely, $3k^c \leq r_i \leq 4k^c$, $k^{2c} \leq b_i \leq k^{2c} + k^c$, and $k^{4c} \leq g_i \leq k^{4c} + k^c$, where $c > 1$ is a constant.) See the full version [6] for further details.

Idea of the Reduction: One crucial ingredient is to construct a tree whose height matches the height-bound H of the drawing; this determines the layer for all vertices on paths to the deepest leaves because in a strictly-upward drawing every edge must lead to a layer farther down. Another crucial ingredient is to add so many vertices to the tree that (other than on the topmost layer) all grid-points must have a vertex on them. The next ingredient is to add "walls" that force the given $W \times H$-grid to be divided into k regions that have B grid points available in each. (These walls are simply high-degree vertices, but since every grid-point must be used, nearly all neighbours of such a vertex must be in the layer below.) Finally we add gadgets that encode r_i, b_i, g_i in such a way that b_i vertices must be in the ith region defined by the walls, while r_i and g_i can freely choose into which of the regions they fall. Therefore, the division of them into the regions gives rise to a solution to N3DM.

Construction of T: Given an instance $\mathcal{I}$ of N3DM, we construct a tree T with height $2k + 3$ as follows. We start with the *spinal path* $v_1, \ldots, v_{2k+3}$ and choose v_1 to be the root of T. We add two *supporting paths* $u_2, \ldots, u_{2k+3}$ and $w_2, \ldots, w_{2k+3}$ where u_2 and w_2 are children of v_1. We set $H := 2k + 3$; hence in any strictly-upward drawing, u_i, v_i, w_i must be on layer ℓ_i for $2 \leq i \leq 2k+3$. (We count the layers from top to bottom, i.e., layer ℓ_1 is the layer with y-coordinate H that contains the root v_1, and ℓ_i is the layer whose y-coordinate is $H-i+1$.)

Next we add the "walls" as shown in Fig. 1(a). Namely, add $B + 1$ leaves to T that are children of the root; the star graph induced by v_1 and its children is called the *wall of* v_1 with *wall root* v_1. The *wall children* of v_1 are these $B + 1$ added leaves, as well as child v_2. Similarly we add a *wall of* v_{2j} for $j \in \{1, \ldots, k+1\}$, by adding $B+1$ leaves to T that are children of v_{2j} so that v_{2j} has $B+2$ wall children (including v_{2j+1}).

Finally, we encode r_i, g_i, b_i of the N3DM instance. For $1 \leq i \leq k$, we add $b_i - 1$ leaves to T and make them children of v_{2i+1}, see Fig. 1(b). We will call v_{2i+1} a *blue parent* and its b_i children (including v_{2i+2}) the *blue children* of v_{2i+1}. For each r_i (resp., g_i), we create a star with r_i (resp., g_i) leaves and connect the center



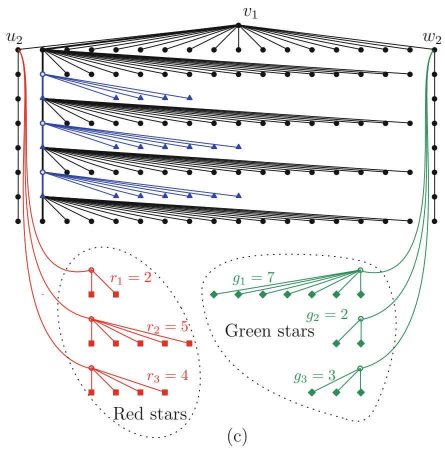


Fig. 1. (a) Illustration for the spinal path, supporting paths and walls. (b) Construction of the blue vertices. (c) Construction of the red and green vertices. For space reasons, the numbers in this example do not satisfy the constraints that we imposed in N3DM. (Color figure online)

to u_2 (resp., w_2), see Fig. 1(c). We refer to the stars corresponding to r_i and g_i as the *red* and *green stars*, respectively. This finishes the construction of tree T. Set $W := B+4$ and observe that T has $2kB+2B+8k+9 = (B+4)(2k+2)+1 = W \times (H-1)+1$ vertices, which means that in any strictly-upward drawing in a $W \times H$-grid, all layers except the top one must be completely full because the top layer can contain only one vertex (the root v_1).

From N3DM to Tree Drawing: If $\mathcal{I}$ is a yes-instance, then we create a straight-line strictly-upward drawing of T on a $W \times H$ grid as illustrated in Fig. 2. The root v_1 is anywhere in the top layer. We place the supporting paths in layers $\ell_2, \dots, \ell_{2k+3}$ in the leftmost (resp., rightmost) column, as forced by the strictly-upward constraints. Vertex v_1 has $B+4$ children (u_2, w_2 and the $B+2$ wall children); we place all these in the second layer. We do not yet pick which of these $B+2$ wall children becomes v_2; this will be determined later.

The solution to $\mathcal{I}$ gives two permutations π, π'. Place the red, blue and green parents corresponding to $r_{\pi(i)}$, b_i and $g_{\pi'(i)}$ on layer ℓ_{2i+1}. More precisely, from left to right, we have first u_{2i}, then the red parent, then B wall children of v_{2i}, then the green parent, and then w_{2i}. Later, we will choose one of these B children to be v_{2i+1}, hence the blue parent corresponding to b_i. Observe that layer ℓ_{2i+1} is completely filled with vertices of T.

Place the red/blue/green children corresponding to $r_{\pi(i)}$, b_i and $g_{\pi'(i)}$ on layer ℓ_{2i+2}. More precisely, from left to right, we have first u_{2i}, then $r_{\pi(i)}$ red children, then one wall child of v_{2i}, then b_i blue children of v_{2i+1}, then another wall child of v_{2i}, then $g_{\pi'(i)}$ green children and finally w_{2i}. Since $r_{\pi(i)} + b_i + g_{\pi'(i)} = B$, layer ℓ_{2i+2} is also completely filled.

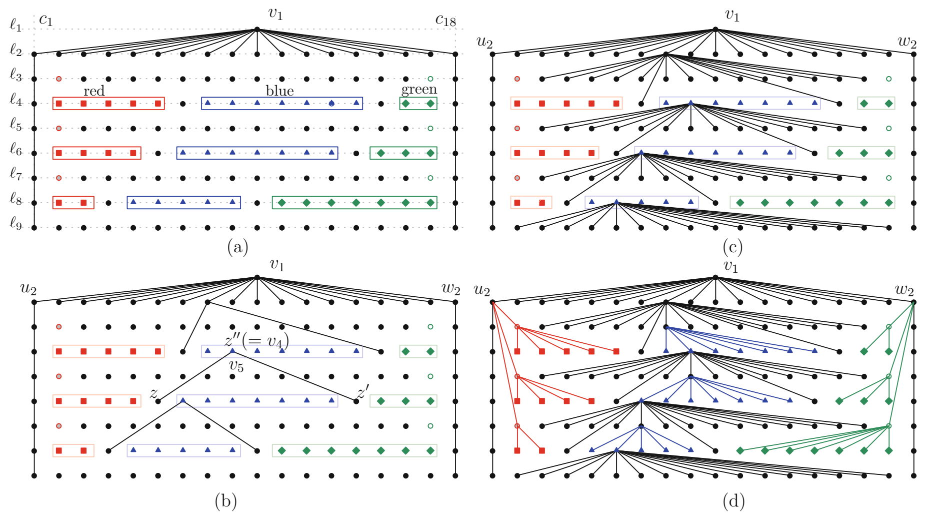


Fig. 2. Construction of a drawing of T on a $W \times H$ grid. (a) Placement of the vertices of T, drawing of the edges of the supporting paths, and the wall of v_1. (b)–(c) Drawing of the blue stars and the remaining walls. (d) Drawing of the red and green stars. (Color figure online)

We must argue that all edges can be connected straight-line. This holds for the red stars since red children lie in the layer below their parent, and red parents lie in the column next to u_2. See also Fig. 2(d). Similarly we can connect green stars. For the blue stars, because the b_i's are much larger than the r_i's, one can argue that the blue intervals of children of v_{2i-1} and v_{2i+1} overlap. We pick v_{2i} to be within this overlap in such a way that it can connect to the two wall children that are on layer ℓ_{2i+2} without using up grid points. We use as v_{2i+1} the point directly below v_{2i}; due to the choice of v_{2i} then v_{2i+1} can connect to the blue interval on layer ℓ_{2i+2} without crossing. Details are in the full version [6].

From Tree Drawing to N3DM: We now show that any strictly-upward polyline drawing of T in a $W \times H$-grid gives rise to permutations π and π' such that $r_{\pi(i)} + b_i + g_{\pi'(i)} = B$ holds for $1 \leq i \leq k$. We select $r_{\pi(i)}$ and $g_{\pi'(i)}$ in a bottom-up fashion, i.e., we first construct the triple $(r_{\pi(k)}, b_k, g_{\pi'(k)})$, then the triple $(r_{\pi(k-1)}, b_{k-1}, g_{\pi'(k-1)})$, and so on.

Since H equals the height of the tree, we know that v_i, u_i, w_i (for $i > 1$) are on layer ℓ_i, and the wall children of v_{2k+2} are on layer ℓ_{2k+3} (the bottommost layer). Hence layer ℓ_{2k+3} contains these $B + 2$ wall children, as well as u_{2k+3} and w_{2k+3}. By $W = B + 4$ this layer is full and contains no other vertices. Also v_{2k} lies is on layer $2k$, and so all its wall children must be on layers ℓ_{2k+1} and ℓ_{2k+2}. We need an observation that crucially requires that all grid points are used. Details are in the full version [6].

Lemma 1. *Presume we know that all wall children of v_{2i} are on layers ℓ_{2i+1} and ℓ_{2i+2}. Then at most two wall children of v_{2i} are on layer ℓ_{2i+2}, and the wall children of v_{2i} on layer ℓ_{2i+1} occupy a consecutive set of points.*

Thus there are at least B wall children of v_{2k} that form an interval on ℓ_{2k+1}. Also u_{2k+1} and w_{2k+1} are on layer ℓ_{2k+1}, leaving at most two points in this layer free to be used for red or green stars, or blue or wall children from higher up.

We argue that indeed these two points must be used for a red parent and a green parent. To see this, consider the interval λ_m that consists of the middle $W-16=B-12$ points on layer ℓ_{2k+2}. T has $O(k^{2c+1})$ vertices that are not in green stars while $B > 3k^c + k^{2c} + k^{4c}$ so there must exist a green vertex in λ_m. It must be a green leaf l_g since layer ℓ_{2k+3} is full.

We claim that if a vertex x uses a point in λ_m, then its parent p_x is either v_{2k} or p_x is on layer ℓ_{2k+1}, i.e., one layer above x. To see this, assume otherwise and consider the place where edge (x, p_x) traverses layer ℓ_{2k+1}. If $p_x \neq v_{2k}$ then this point must be outside the interval of the $B = W - 4$ wall children of v_{2k} in this layer, else we would intersect an edge. So this point is within the leftmost or rightmost four units of ℓ_{2k+1}. Since λ_m covers all but the leftmost and rightmost 8 points of layer ℓ_{2k+2}, this forces p_x to be outside the allotted width of the drawing, a contradiction. See the details in the full version [6].

Consequently, the green parent p_g of l_g is on layer ℓ_{2k+1} and its green children are all on ℓ_{2k+2}. Due to our assumptions on N3DM, these green children, plus the blue children of v_{2k+1}, are not enough to fill λ_m, but leave too little space to place another green star. Therefore some point in λ_m must be occupied by a red child l_r, and its parent p_r hence is in layer ℓ_{2k+1}. Let $r_{\pi(k)}$ and $g_{\pi'(k)}$ be the numbers corresponding to the red and green stars at p_r and p_g.

We had $u_{2k+1}, w_{2k+1}, p_r, p_g$, and at least B wall children in ℓ_{2k+1}, which by $W = B + 4$ means that there are exactly B wall children in ℓ_{2k+1} and no other vertices. So two wall children are in layer ℓ_{2k+2}. These two, plus u_{2k+2} and w_{2k+2}, leave B points for the red, blue and green children, so $r_{\pi(k)} + b_k + g_{\pi'(k)} \leq B$. On the other hand, λ_m cannot contain any vertex x other than two wall-children of v_{2k} and these red, blue and green children, because layer ℓ_{2k+1} has no space left for the parent of x. So $r_{\pi(k)} + b_k + g_{\pi'(k)} \geq B - 14$. Since all input numbers are big enough, this implies $r_{\pi(k)} + b_k + g_{\pi'(k)} = B$, as desired.

It also follows that ℓ_{2k+1} is completely filled by these vertices, which means that no wall children of v_{2k-2} can be in layer ℓ_{2k+1} or below. We can now apply the same argument iteratively to compute the upper level triples $(r_{\pi(k-1)}, b_{k-1}, g_{\pi'(k-1)}), \ldots, (r_{\pi(1)}, b_1, g_{\pi'(1)})$. This completes the NP-hardness reduction.

We can extract π and π' from any polyline drawing, while a solution to $\mathcal{I}$ gives rise to a straight-line drawing. Therefore, the reduction holds both in polyline and straight-line drawing settings. Since SUTD is clearly in NP we hence have:

Theorem 1. *Given a tree T, and two natural numbers W and H, it is NP-complete to decide whether T admits a strictly-upward (polyline or straight-line) drawing with width at most W and height at most H.*

3 Directions for Future Research

Several interesting questions remain. In our reduction, we crucially used that the underlying tree has high degree, that the height of the drawing equals the height of the tree, that the order among children is not fixed, and that the drawing is *strictly* upward. Does SUTD remain NP-hard for bounded degree trees? What if the given width is optimal for the tree and the height needs to be minimized? How about order-preserving drawings and/or upward drawings?

The above questions are open also for many other popular styles for drawing trees. Specifically, is the problem of computing (not necessarily upward) drawings of trees on a given grid NP-hard? Are there polynomial-time algorithms that can approximate the area within a constant factor?

References

1. Alam, M.J., Samee, M.A.H., Rabbi, M., Rahman, M.S.: Minimum-layer upward drawings of trees. J. Graph Algorithms Appl. **14**(2), 245–267 (2010)
2. Bachmaier, C., Matzeder, M.: Drawing unordered trees on k-grids. J. Graph Algorithms Appl. **17**(2), 103–128 (2013)
3. Bhatt, S.N., Cosmadakis, S.S.: The complexity of minimizing wire lengths in VLSI layouts. Inf. Process. Lett. **25**(4), 263–267 (1987)
4. Biedl, T.: On area-optimal planar graph drawings. In: Esparza, J., Fraigniaud, P., Husfeldt, T., Koutsoupias, E. (eds.) ICALP 2014, Part I. LNCS, vol. 8572, pp. 198–210. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-43948-7_17
5. Biedl, T.: Optimum-width upward drawings of trees I: rooted pathwidth. CoRR abs/1502.02753 (2015)
6. Biedl, T., Mondal, D.: On upward drawings of trees on a given grid. CoRR abs/1708.09515 (2017). http://arxiv.org/abs/1708.09515
7. Brunner, W., Matzeder, M.: Drawing ordered $(k-1)$-ary trees on k-grids. In: Brandes, U., Cornelsen, S. (eds.) GD 2010. LNCS, vol. 6502, pp. 105–116. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-18469-7_10
8. Di Battista, G., Frati, F.: A survey on small-area planar graph drawing (2014), coRR report 1410.1006
9. Garey, M., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W H Freeman & Co, New York (1979)
10. Gregori, A.: Unit-length embedding of binary trees on a square grid. Inf. Process. Lett. **31**(4), 167–173 (1989)
11. Krug, M., Wagner, D.: Minimizing the area for planar straight-line grid drawings. In: Hong, S.-H., Nishizeki, T., Quan, W. (eds.) GD 2007. LNCS, vol. 4875, pp. 207–212. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-77537-9_21
12. Mahaney, S.R.: Sparse complete sets of NP: Solution of a conjecture of Berman and Hartmanis. J. Comput. Syst. Sci. **25**(2), 130–143 (1982)
13. Marriott, K., Stuckey, P.J.: NP-completeness of minimal width unordered tree layout. J. Graph Algorithms Appl. **8**(2), 295–312 (2004)
14. Mondal, D., Alam, M.J., Rahman, M.S.: Minimum-layer drawings of trees. In: Katoh, N., Kumar, A. (eds.) WALCOM 2011. LNCS, vol. 6552, pp. 221–232. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19094-0_23

15. Mondal, D., Nishat, R.I., Rahman, M.S., Alam, M.J.: Minimum-area drawings of plane 3-trees. J. Graph Algorithms Appl. **15**(2), 177–204 (2011)
16. Supowit, K.J., Reingold, E.M.: The complexity of drawing trees nicely. Acta Informatica **18**, 377–392 (1982)
17. Tamassia, R. (ed.): Handbook of Graph Drawing and Visualization (Discrete Mathematics and Its Applications). Chapman and Hall/CRC, Boca Raton (2014)
18. Trevisan, L.: A note on minimum-area upward drawing of complete and Fibonacci trees. Inf. Process. Lett. **57**(5), 231–236 (1996)