Area Requirement of Visibility Representations of Trees

G. Kant, G. Liotta, R. Tamassia
and I. G. Tollis

Department of Computer Science
Brown University
Providence, Rhode Island 02912

CS-96-23
November 1996

Area Requirement of Visibility Representations of Trees*

Goos Kant[†] Giuseppe Liotta[‡] Roberto Tamassia[§] Ioannis G. Tollis[¶]

Abstract

We study the area requirement of bar-visibility and rectangle-visibility representations of trees in the plane. We prove asymptotically tight lower and upper bounds on the area of such representations, and give linear-time algorithms that construct representations with asymptotically optimal area.

1 Introduction

Visibility is a fundamental relation in computational geometry (see, e.g., [1, 8, 9]). In particular, the problem of constructing *visibility representations* of graphs, where the vertices are drawn as geometric objects of a certain class and the edges are associated with pairs of "visible" objects, has been extensively investigated (see, e.g., [3, 5, 12]).

Let S be a set of nonintersecting objects (e.g., segments or polygons) in the plane. Two objects s' and s'' of S are said to be *vertically visible* if there exists a vertical segment that intersects s' and s'' but no other object of S. The notion of *horizontally visible* pair of objects of S is similarly defined.

In a *bar-visibility representation* of a graph G, the vertices are drawn as horizontal segments, called *bars*, and the edges are represented by pairs of vertically visible bars, i.e., there is an edge (v', v'') in G if and only if the bars b' and b'' associated with vertices v' and v'' are vertically visible (see Figure 1.b). A graph admits a bar-visibility representation only if it is planar. However, the converse is not always true. The class of graphs that admit a bar-visibility representation where the bars are open or semi-open segments (i.e., they include none or one of their endpoints) is well-characterized. Namely, such graphs are the planar graphs that admit an embedding where all the cutvertices lie on the same face [11, 13], and can be recognized in linear time. If instead the bars are required to be closed segments (i.e., to include their endpoints), then it is NP-complete to determine whether a graph admits a bar-visibility representation [2]. If G is a directed graph, an *upward bar-visibility representation* of G is a bar-visibility representation such that for every edge (v', v''), the bar of v'' is vertically above the bar of v'. Clearly, if G admits such a representation, then G is acyclic.

In a *rectangle-visibility representation* of a graph G, the vertices are drawn as rectangles with horizontal and vertical sides, and the edges are represented by pairs of rectangles that are either horizontally or vertically visible, i.e., there is an edge (v', v'') in G if and only if the rectangles r' and r'' associated with vertices v' and v'' are vertically or horizontally visible (see Figure 1.c). Only partial characterizations of rectangle-visibility graphs are known. E.g., they include planar graphs [14] and complete bipartite graphs $K_{p,q}$ with $p \leq q$ and $p \leq 4$ [4]).

*Work supported in part by the US National Science Foundation under grant CCR–9423847, by the US Army Research Office under grant DAAH04–96–1–0013, by the NATO Scientific Affairs Division under collaborative research grant 911016, and by the NATO–CNR Advanced Fellowships Programme. Research originated and performed in part at the Bellairs Research Institute of McGill University.

[†]Department of Computer Science, Utrecht University P.O. Box 80.089, 3508 TB Utrecht, The Netherlands. goos@cs.ruu.nl.

[‡]Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza" via Salaria 113, I–00198 Roma, Italy liotta@dis.uniroma1.it. Work of this author performed in part while he was with the Center for Geometric Computing of the Department of Computer Science at Brown University.

[§]Center for Geometric Computing, Department of Computer Science, Brown University, Providence, RI 02912–1910, USA. rt@cs.brown.edu.

[¶]Department of Computer Science, University of Texas at Dallas, Richardson, TX 75083–0688, USA. tollis@utdallas.edu.

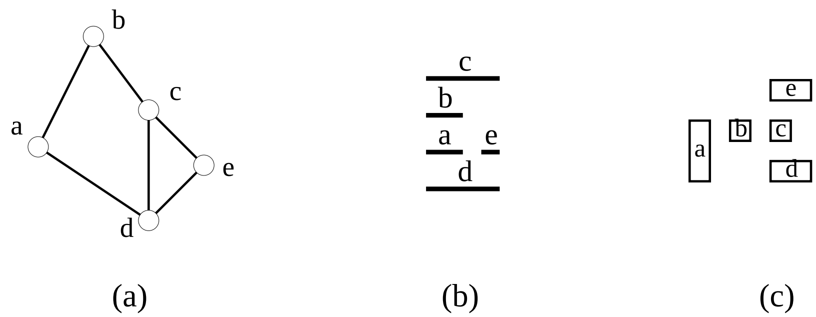

Figure 1: (a) Graph G. (b) Bar-visibility representation of G. (c) Rectangle-visibility representation of G.

The area requirement of a geometric representation of graphs is an important paramenter in assessing the practical usefulness of the representation in information visualization applications. In this paper, we investigate the area requirement of bar- and rectangle-visibility representations of trees. It is easy to see that every tree admits either representation, even if constraints are placed on the inclusion of endpoints (sides) in the bars (rectangles). Hence, in this paper we assume without loss of generality that the bars and rectangles are closed, i.e., the bars contain both their endpoints and the rectangles contain all their four sides and corners.

We define the area of a bar- or rectangle-visibility representation as the area of the smallest rectangle with horizontal and vertical sides covering the representation. We also assume that: (i) the endpoints of the bars and the corners of the rectangles have integer coordinates; (ii) the bars have nonzero length; and (iii) the rectangles have nonzero width and height. Such *resolution rules* are customary in evaluating the area requirement of drawings of graphs (see, e.g. [6, 7, 10]).

The rest of this paper is organized as follows. Preliminary definitions are given in Section 2. In Sections 3 and 4, we study the area requirement of bar- and rectangle-visibility representations of trees, respectively. We prove asymptotically tight lower and upper bounds on the area of such representations, and give linear-time algorithms that construct representations with asymptotically optimal area. Our results are summarized in Table 1. No previous results were known on the area requirement of visibility representations of trees. Open problems include determining the area requirement of rectangle-visibility representations where the "visibility corridors" are required not to intersect, and the volume requirement of three-dimensional visibility representations of trees.

Representation	Area
upward bar-visibility	$O(l \cdot h)$
bar-visibility	$O(l \cdot h^* + n - 1)$
rectangle-visibility	$O(l \cdot n)$

Table 1: Summary of our results on the area requirement of visibility representations of trees. The number of vertices and the number of leaves of the tree are denoted by n and l, respectively. If the tree is rooted, its height is denoted by h. If the tree is not rooted, its critical height (see Section 2) is denoted by h^*.

2 Preliminaries

We begin by introducing some terminology and notation used in this paper. A *free tree* is a connected acyclic graph. A *leaf* of a free tree is a vertex that has degree one in the tree. A *rooted tree* is a free tree T along with a distinguished vertex called the *root* of T. The *length* of a path P is the number of edges of P. The *height* of a rooted tree T, denoted by $h(T)$, is the length of a longest path from the root of T to a leaf.

Let T be a free tree, and let v be a vertex of T. A *subtree* of v in T is a subtree of the rooted tree obtained by rooting T at v. Let $T_1, T_2, \cdots, T_d$ be the subtrees of v sorted by nonincreasing height (i.e., $h(T_1) \geq h(T_2) \geq \cdots \geq h(T_d)$). We call $h(T_k)$ the *k-th height* of v. A subtree of v with height equal to the k-th height of v is called a *k-tallest subtree* of v. If T has degree at least 3, we call *critical height* of T the maximum third-height of any vertex of T, and *critical vertex* of T a vertex with maximum third-height. In the example of Figure 2, each degree-3 vertex is labeled with the heights of its subtrees, and the unique critical vertex is shown.

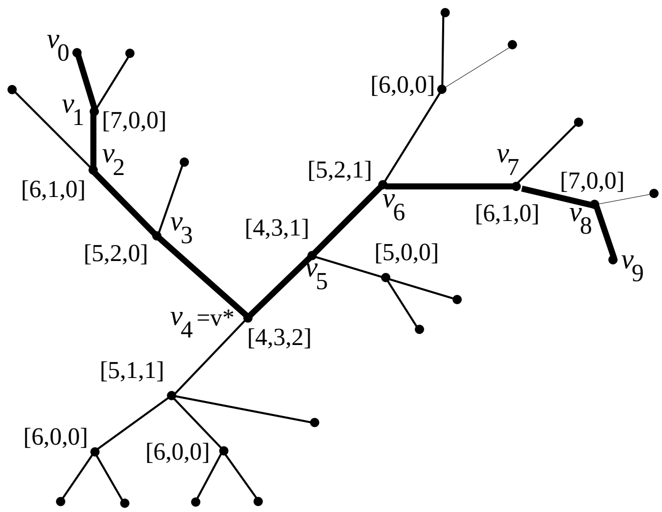

Figure 2: Example of free tree with critical height $h^* = 3$. A longest path is drawn with thick lines. Each internal vertex is labeled with the heights of its subtrees, and the unique critical vertex v^* is shown.

In this paper, points, segments and rectangles are drawn on an integer grid (i.e., with integer coordinates). In a visibility representation, we denote by $S(v)$ the bar or rectangle associated with vertex v. The geometry of $S(v)$ is specified by its rightmost and leftmost x-coordinates, and its lower and upper y-coordinates:

$$S(v) = ([x_1, x_2], [y_1, y_2]).$$

If $x_1 = x_2$ ($y_1 = y_2$), for brevity we write $S(v) = (x_1, [y_1, y_2])$ ($S(v) = ([x_1, x_2], y_1)$). We also denote x_1 by $x_L(v)$ and x_2 by $x_R(v)$.

3 Bar-Visibility Representations

In this section we study the area required by bar-visibility representations of trees. We consider first rooted trees, and then free trees.

3.1 Rooted Trees

Lemma 1 *Let T be a rooted tree with l leaves and height h. The area required by an upward bar-visibility representation of T is at least $(2l - 1) \cdot h$.*

Proof: Let Γ be an upward bar-visibility representation of T. To avoid unwanted visibilities, no two leaves of T can be represented by bars sharing the same column of the grid. Thus, the width of Γ is at least $2l - 1$. Also, the bar of a nonleaf vertex Γ must be placed above the bars of its children. Thus, the height of Γ is at least h. □

The following lemma shows an algorithm that constructs an upward bar-visibility representation of a rooted tree.

Lemma 2 *Let T be a rooted tree with n vertices, l leaves, and height h. There exists an algorithm that constructs in $O(n)$ time an upward bar-visibility representation of T with width $2l - 1$ and height h.*

Proof: The algorithm works as follows: First, the children of each vertex of T are arbitrarily ordered and the leaves are labeled as $v_1, \cdots, v_l$, from left to right. Next, each leaf v_i is drawn as bar $S(v_i) = ([2i, 2i+1], 0)$. Finally, for each internal vertex v, let T_v be the subtree rooted at v, $h(v)$ be the height of $T(v)$, and v_i and v_j be the leftmost and rightmost leaves in T_v; vertex v is drawn as bar $S(v) = ([2i, 2j+1], h(v)]$.

The bounds on the width, height, and time complexity are trivial. We show that set Γ of bars constructed by the algorithm is an upward bar-visibility representation of tree T. No two bars of Γ intersect. For each internal vertex v of T the leftmost (rightmost) x-coordinate of $S(v)$ in Γ is the leftmost (rightmost) x-coordinate of the bar associated with the leftmost (rightmost) child of v. The y-coordinate of $S(v)$ is greater than the maximum y-coordinate of its children. Thus $S(v)$ and the bars associated with the children of v are visible. Since the subtrees rooted at the children of v are entirely drawn in the vertical strips below the bars associated with their roots, no other descendants of v are visible from $S(v)$. □

In the rest of the paper we will refer to the algorithm in the proof of Lemma 2 as Algorithm Upward-Bar-Visibility. In Figure 3, we give an example of the upward bar-visibility representation constructed by Algorithm Upward-Bar-Visibility.

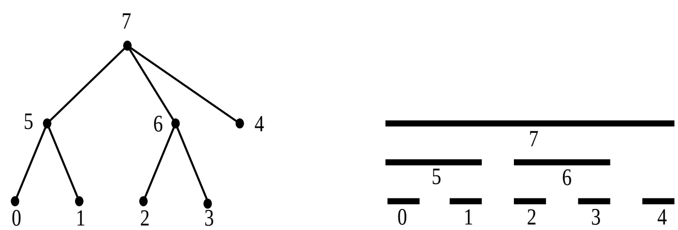

Figure 3: Example of upward bar-visibility representation constructed by Algorithm Upward-Bar-Visibility.

Combining the results of Lemmas 1 and 2, we obtain the following theorem:

Theorem 1 *Let T be a rooted tree with n vertices, l leaves and height h. The area required by an upward bar-visibility representation of T is $\Omega(l \cdot h)$. Also, an upward bar-visibility representation of T with $O(l \cdot h)$ area can be computed in $O(n)$ time.*

3.2 Free Trees

Lemma 3 *Let T be a free tree with n vertices, l leaves, and critical height h^*. The area required by a bar-visibility representation of T is $\Omega(l \cdot h^* + n - 1)$.*

Proof: Let Γ be a bar-visibility representation of T. Each edge of T contributes at least one unit to the area of Γ since it uses such unit either for the visibility corridor, or to separate two bars. Thus the area of Γ is at least $n - 1$. Since the bar associated with a leaf of T can see only one other bar, a grid column can contain at most two bars associated with leaves (one above and one below the bars representing the vertices of the path between them), and hence the width of Γ is at least $l - 1$. Also, for each vertex v, all but two subtrees of v must be drawn upward in the vertical strip between the endpoints of $S(v)$. It follows that the height of Γ is at least h^*. □

The following lemma shows how critical vertices and longest paths are related. Its proof is illustrated in Figure 4.

Lemma 4 *Let P be a longest path of free tree T, and v^* a critical vertex of T. Then v^* is on P.*

Proof: Assume, for a contradiction, that v^* is not in P. Then, P must be entirely contained in a subtree of v^*. We claim that any third-tallest subtree T^* of v^* does not contain path P.

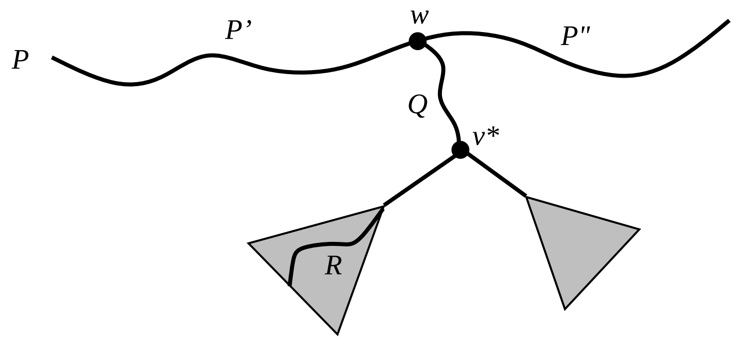

Figure 4: Illustration of the proof of Lemma 4.

Proof of the Claim: Assume, for a contradiction, that T^* contains P. Let w be the vertex of P closest to v^*, and let Q be the path of T joining v^* to w. Let $\ell(Z)$ denote the length of a path Z). Vertex w splits P into subpaths P' and P'' , where $\ell(P') \geq \ell(P'')$ and $\ell(P') \geq \ell(P)/2$. Let R be a longest path from v^* to a leaf. Since the first-tallest subtree of v^* does not contain P, we have that $\ell(R) \geq \ell(Q)+\ell(P')$. Hence, the path obtained by the concatenation of P', Q, and R has length at least $2(\ell(Q)+\ell(P')) > \ell(P)$, which contradicts the hypothesis that P is a longest path. This concludes the proof of the claim.

As in the above proof of the claim, we let w be the vertex of P closest to v^*, and Q be the path of T joining v^* to w. Since P is a longest path, the two subtrees of w containing a subpath of P are each a first- or a second-tallest subtree. Hence, a third-tallest subtree of w has height at least equal to $\ell(Q)+h^* > h^*$, which contradicts the fact that v^* is a critical vertex. □

Let P be a longest path of free tree T, and let v be a vertex of P. We denote with T_v the tree rooted at v obtained by deleting from T the subtrees of v rooted at the neighbors of v in P.

Lemma 5 *Let P be a longest path of free tree T with critical height h^*, and let v be a vertex of P. Then $h(T_v) \leq h^*+1$.*

Proof: Since P is a longest path, the two subtrees of v containing a subpath of P are a first- and second-tallest ones. □

Now, we show how to construct a bar-visibility representation of a free tree T that has a longest path P without degree-2 vertices. Next, we will extend the algorithm to work for any tree.

Lemma 6 *Let T be a free tree with n vertices, l leaves, and critical height h^*, such that T that has a longest path P without degree-2 vertices. There exists an algorithm that constructs in $O(n)$ time a bar-visibility representation of T with width at most $3 \cdot l$ and height at most h^*+2.*

Proof: Let $P = (v_0, v_1, \cdots, v_{m+1})$. The algorithm consists of four steps. (1) For each $i = 1, \cdots, m$, apply Algorithm Upward-Bar-Visibility to T_{v_i}, which yields an upward bar-visibility representation of T_{v_i}. If i is odd, extend $S(v_i)$ two units to the left and two units to the right. If i is even and $S(v_i)$ has length ≤ 2, extend $S(v_i)$ one unit to the left and one unit to the right. Let Γ_i be the resulting bar-visibility representation of T_{v_i}. (2) Draw v_0 as bar $S(v_0) = ([0,2],0)$. (3) For $i = 1, \cdots, m$, translate Γ_i such that $x_L(S(v_i)) = x_R(S(v_{i-1})) - 1$ and $y(S(v_i)) = i \bmod 2$. (4) Draw v_{m+1} as bar $S(v_{m+1}) = (x_R(S(v_m)) - 1, [x_R(S(v_m)) + 2], (m+1) \bmod 2)$.

Let Γ be the set of bars constructed by the algorithm, The correctness of the algorithm is established by observing that two bars of Γ are visible if and only if its associated vertices are adjacent in T. By Lemma 2, the height of Γ_i is equal to $h(T_{v_i})$, and by Lemma 5, $h(T_{v_i}) \leq h^*+1$. Hence, the height of Γ is at most h^*+2. By Lemma 2, the width of Γ_i is $2 \cdot l_i + 3$ if i is odd, and at most $2 \cdot l_i + 1$ if i is even, where l_i is the number of leaves of T_{v_i} (recall that bar $S(v_i)$ may be extended in Step 1) . Hence, since Γ_i and Γ_{i+1} have vertical overlap in a vertical strip of unit width, each leaf contributes to at most three units of width. The time-complexity bound follows from Lemma 2. □

In the rest of the paper we will refer to the algorithm in the proof of Lemma 6 as Algorithm Special-Bar-Visibility. The following lemma shows how to construct a bar-visibility representation of a path with linear-area any prescribed height.

Lemma 7 *Let P be a path with n vertices, and let k be a positive integer parameter with $k \leq n$. There exists an algorithm that constructs in $O(n)$ time an $O(n)$-area bar-visibility representation of P with height k and width $2 \cdot \lceil \frac{n}{k} \rceil + 1$.*

Proof: The algorithm consists of three steps. (1) Partition P into subpaths $P_1, \cdots, P_s$, where P_s has at most k vertices, and the remaining subpaths have exactly k vertices ($s = \lceil \frac{n}{k} \rceil$). (2) For each subpath P_i, draw P_i as a stack of left-aligned bars, vertically spaced by one unit. The last bar has length three, and the remaining bars have length one. The stack extends downward or upward according to whether i is odd or even. (3) Combine the representations of the subpaths as follows. Let v_i and w_i be the first and last vertex of P_i, respectively. For $1 < i \leq s$, set $x_L(v_{i+1}) = x_R(w_i) - 1$ and $|y(v_{i+1}) - y(w_i)| = 1$. Also bar $S(v_{i+1})$ is below or above $S(w_i)$ according to whether i is odd or even.

The time complexity bound is immediate. Each bar does not intersect any other bar. Also, a bar $S(v)$ is visible only by the bars representing the predecessor and the successor of v in P. The stack constructed for each subpath of P in Step 2 has width 3 and height $k-1$. In Step 3, we form the representation of the path by combining the stacks representing the subpaths. We use one extra grid row to make visible the first and last bar of two consecutive stacks, and we add two new grid columns each time we add a new stack. Since there are $\lceil \frac{n}{k} \rceil$ stacks, we conclude that the representation has height k and width $2 \cdot \lceil \frac{n}{k} \rceil + 1$. □

In the rest of the paper we will refer to the algorithm in the proof of Lemma 7 as Algorithm Path-Bar-Visibility. An example of bar-visibility representation produced by Algorithm Path-Bar-Visibility is displayed in the dotted area of Figure 5.b.

We are now ready to describe the algorithm for general free trees.

Lemma 8 *Let T be a free tree with n vertices, l leaves and critical height h^*. There exists an algorithm that constructs in $O(n)$ time a bar-visibility representation of T with area $O(l \cdot h^* + n)$.*

Proof: The algorithm, consists of four steps. (1) Compute a longest path P of T. (2) For each maximal subpath Q of P consisting of degree-2 vertices, contract Q into a single edge e_Q. Let T' and P' be the resulting modified tree and path. (3) Apply Algorithm Special-Bar-Visibility to compute a bar-visibility representation Γ' of T', where we use path P' instead of the longest path of T'. Note that each edge of P' associated with a subpath of degree-2 vertices of P is associated with a unit square in Γ'. (4) For each maximal subpath Q of P consisting of degree-2 vertices, construct a bar-visibility representation $\Gamma(Q)$ of Q by applying Algorithm Path-Bar-Visibility with parameter h^*, and replace s_Q with $\Gamma(Q)$, as shown in Figure 5.

By Lemma 6, the bar-visibility representation Γ' of T' constructed in Step 3 has width $O(l)$ and height $O(h^*)$. By Lemma 7, each bar-visibility representation $\Gamma(Q)$ constructed in Step 4 has height h^* and area proportional to the number of vertices n_Q of Q. The replacements performed in Step 4 increase the height by at most a constant. Hence, each replacement increases the area by $O(n_Q)$. We conclude that the area of the bar-visibility representation of T is $O(n + l \cdot h^*)$. The time complexity bound follows from Lemmas 6 and 7. □

By combining Lemmas 3 and 8, we summarize the results of this section in the following theorem.

Theorem 2 *Let T be a free tree with n vertices, l leaves, and critical height h^*. The area required by a bar-visibility representation of T is $\Omega(l \cdot h^* + n - 1)$. Also, a bar-visibility representation of T with $O(l \cdot h^* + n)$ area can be computed in $O(n)$ time.*

4 Rectangle-Visibility Representations

In this section, we study the area required by rectangle-visibility representations of free trees.

Lemma 9 *Let T be a free tree with n vertices and l leaves. A rectangle-visibility representation of T requires area $\Omega(l \cdot n)$.*

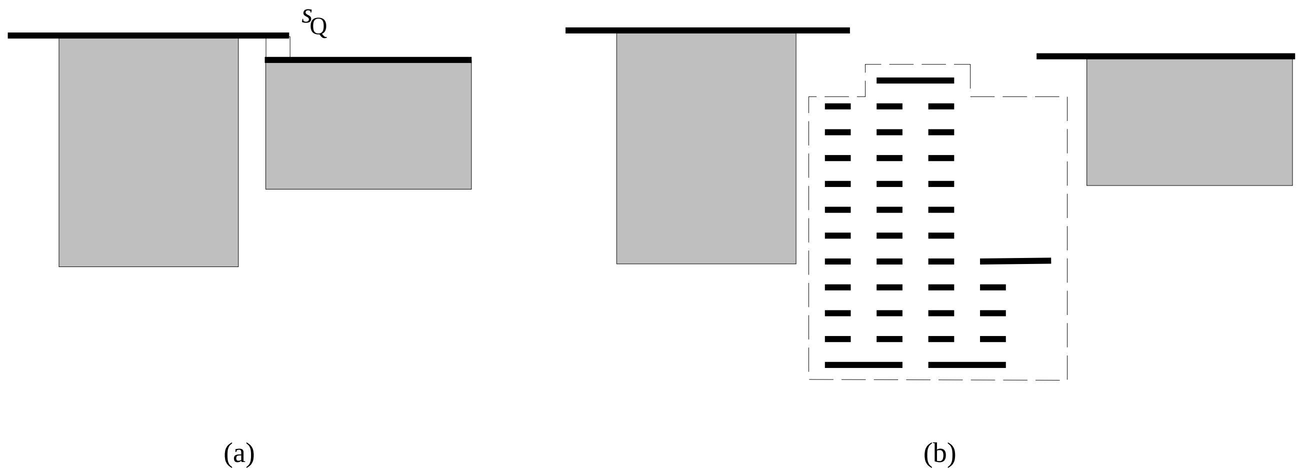

Figure 5: Illustration of Step 4 of the algorithm of Lemma 8, where a unit rectangle s_Q is replaced with $\Gamma(Q)$, (a) before the replacement; (b) after the replacement.

Proof: Let Γ be a rectangle-visibility representation of T. The rectangle of a leaf can see (either horizontally or vertically) only another rectangle. Thus, a leaf contributes at least one unit to both the height and the width of the representation. The rectangle of a degree-2 vertex can see (either horizontally or vertically) only another rectangle. Thus, a degree-2 vertex contributes at least one unit to either the height or the width of the representation. Let k be the number of degree-2 vertices of T. By the above considerations, the area of Γ is $\Omega(l \cdot (k+l))$.

Let p be the number of vertices with degree at least 3, then $p = n - k - l$. Since the sum of the degrees of all the vertices of T is $2(n-1)$, we have $l + 2k + 3p \leq 2(n-1)$, thus $p \leq l - 2$. Hence,

$$2(n-1) \geq l + 2k + 3p \geq 2k + 4p + 2 \geq 4p + 2.$$

This yields

$$p \leq \frac{2n-2-2}{4} = \frac{n}{2} - 1 \leq \lfloor \frac{n}{2} \rfloor.$$

Therefore, $k + l \geq \lceil \frac{n}{2} \rceil$, which implies that the area of Γ is $\Omega(l \cdot n)$. □

Now, we describe an algorithm for constructing a rectangle-visibility representation of a free tree T.

Lemma 10 *Let T be a free tree with n vertices and l leaves. There exists an algorithm that constructs in $O(n)$ time a rectangle-visibility representation of T with $O(l \cdot n)$ area.*

Proof: The algorithm consists of four steps: (1) Root T at an arbitrary vertex v; order arbitrarily the children of each internal vertex, and label the leaves $v_1, \cdots, v_l$, from left to right. (2) Draw leaf v_i $(i = 1, \cdots, l)$ as rectangle $S(i) = ([2i, 2i+1], [2i, 2i+1])$. (3) Label the internal vertices of T in post-order $v_{l+1}, \ldots, v_n$. (4) Draw internal vertex v_j $(j = l+1, \cdots, n)$ as rectangle $S(v_j) = ([x_L(v_{i'}), x_R(v_{i''})], [2j, 2j+1])$, where $v_{i'}$ and $v_{i''}$ are the leftmost and rightmost leaves in the subtree rooted at v_j.

The algorithm can be clearly implemented to run in linear time. It is easy to see that the area of the representation is $(2l-1) \cdot (2n-1)$. To show that the representation is indeed a rectangle-visibility representation, we observe that $S(v_i)$ lies in the horizontal strip between ordinates $2i$ and $2i+1$. Hence, no two rectangles can see each other in the horizontal direction, so that visibility is only in the vertical direction. Using an argument similar to the one in the proof of Lemma 2, we conclude that the representation is a rectangle-visibility representation. □

We refer to the algorithm in the proof of Lemma 10 as Algorithm Rectangle-Visibility. Figure 6 shows an example of a rectangle-visibility representation constructed by Algorithm Rectangle-Visibility.

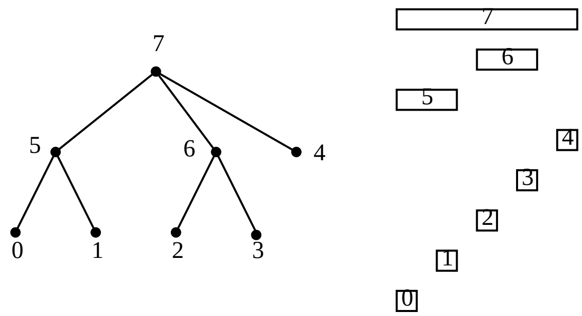

Figure 6: Illustration of Algorithm Rectangle-Visibility.

By combining Lemmas 9 and 10, we obtain the following theorem.

Theorem 3 *Let T be a free tree with n vertices and l leaves. The area required by a rectangle-visibility representation of T is $\Omega(l \cdot n)$. Also, a rectangle-visibility representation of T with $O(l \cdot n)$ area can be computed in $O(n)$ time.*

Acknowledgments

This research is a consequence of the authors' participation to the Workshop on Visibility Representations of Graphs organized by Sue Whitesides and Joan Hutchinson at the Bellairs Research Institute of McGill University, Feb. 12–19, 1993. We thank the participants of the Workshop for useful discussions.

References

[1] P. K. Agarwal, N. Alon, B. Aronov, and S. Suri. Can visibility graphs be represented compactly? *Discrete Comput. Geom.*, 12:347–365, 1994.

[2] T. Andreae. Some results on visibility graphs. *Discrete Appl. Math.*, 40:5–17, 1992.

[3] F. J. Brandenburg, editor. *Graph Drawing (Proc. GD '95)*, volume 1027 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.

[4] A. M. Dean and J. P. Hutchinson. Rectangle-visibility representations of bipartite graphs. In R. Tamassia and I. G. Tollis, editors, *Graph Drawing (Proc. GD '94)*, volume 894 of *Lecture Notes in Computer Science*, pages 159–166. Springer-Verlag, 1995.

[5] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. Algorithms for drawing graphs: an annotated bibliography. *Comput. Geom. Theory Appl.*, 4:235–282, 1994.

[6] G. Di Battista, R. Tamassia, and I. G. Tollis. Area requirement and symmetry display of planar upward drawings. *Discrete Comput. Geom.*, 7:381–401, 1992.

[7] A. Garg, M. T. Goodrich, and R. Tamassia. Area-optimal upward tree drawings. *Internat. J. Comput. Geom. Appl.*, 6(3):333–356, 1996.

[8] S. K. Ghosh and D. M. Mount. An output-sensitive algorithm for computing visibility graphs. *SIAM J. Comput.*, 20:888–910, 1991.

[9] J. O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, New York, NY, 1987.

[10] P. Rosenstiehl and R. E. Tarjan. Rectilinear planar layouts and bipolar orientations of planar graphs. *Discrete Comput. Geom.*, 1(4):343–353, 1986.

[11] R. Tamassia and I. G. Tollis. A unified approach to visibility representations of planar graphs. *Discrete Comput. Geom.*, 1(4):321–341, 1986.

[12] R. Tamassia and I. G. Tollis, editors. *Graph Drawing (Proc. GD '94)*, volume 894 of *Lecture Notes in Computer Science*. Springer-Verlag, 1995.

[13] S. K. Wismath. Characterizing bar line-of-sight graphs. In *Proc. 1st Annu. ACM Sympos. Comput. Geom.*, pages 147–152, 1985.

[14] S. K. Wismath. *Bar-Representable Visibility Graphs and Related Flow Problems*. Ph.D. thesis, Dept. Comput. Sci., Univ. British Columbia, 1989.