A Linear Algorithm for Compact Box-Drawings of Trees

Masud Hasan*
Department of Computer Science, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada

Md. Saidur Rahman,* Takao Nishizeki
Graduate School of Information Sciences, Tohoku University, Aoba-yama 05, Sendai 980-8579, Japan

In a box-drawing of a rooted tree, each node is drawn by a rectangular box of prescribed size, no two boxes overlap each other, all boxes corresponding to siblings of the tree have the same $x$-coordinate at their left sides, and a parent node is drawn at a given distance apart from its first child. A box drawing of a tree is compact if it attains the minimum possible rectangular area enclosing the drawing. We give a linear-time algorithm for finding a compact box-drawing of a tree. A known algorithm does not always find a compact box-drawing and takes time $O(n^2)$ if a tree has $n$ nodes. © 2003 Wiley Periodicals, Inc.

Keywords: algorithm; graph drawing; tree drawing; box-drawing; compact drawing

1. INTRODUCTION

Automatic drawings of trees on a two-dimensional plane have many applications in VLSI layouts, information visualization such as displaying tree-structured diagrams, and hierarchical file structures [2]. The complexity of algorithms for finding a drawing of a tree significantly depends on the aesthetic conditions considered. The problem of finding a drawing of a binary tree on an integer grid of a minimum area is NP-hard for a certain set of aesthetic conditions [8], whereas for a different set of aesthetic conditions, an $O(n \sqrt{n \log n})$ time algorithm is known for finding a drawing of a binary tree on an integer grid of a minimum area [3]. Reingold and Tilford gave a linear-time algorithm for a tidier drawing of trees with a certain set of aesthetic conditions where the nodes are drawn as points and the drawing does not give optimum compactness [6]. In this paper, we consider

a box-drawing of a rooted tree where each node is drawn by a rectangular box of a prescribed size, no two boxes overlap each other, all boxes corresponding to siblings of the tree have the same $x$-coordinate at their left sides, and a parent node is drawn at a given distance apart from its first child. The size of a node is described by the width and height of a box representing the node.

Figure 1(a) depicts a tree with prescribed width and height for each node; the first integer attached to each node represents the node number, and the two integers written inside the parentheses are the width and height. Two box-drawings of the tree in Figure 1(a) are depicted in Figure 1(b,c), where the $y$-coordinate of the top side of a parent box is one less than that of its first child if the parent has two or more children; otherwise, the top sides of a parent and the child have the same $y$-coordinate. Moreover, the ordering of the children of each node in Figure 1(a) is preserved in both Figure 1(b,c). Note that in the box-drawings in Figure 1(b,c) a box is not drawn in the exact scale of the prescribed size, but the width and the height of a drawn box are less than the prescribed size by one-half of one coordinate unit. This amount of gap is used to show the interconnection among a parent and its children. Box-drawings of trees have practical applications in a VLSI layout where a placement is to be found for modules with a prescribed width and height [4, 7].

We call a box-drawing a compact box-drawing if the smallest rectangle enclosing the drawing has the minimum possible area. The box-drawing in Figure 1(c) is a compact box-drawing, whereas the box-drawing of Figure 1(b) is not a compact box-drawing. In this paper, we give a linear-time algorithm for finding a compact box-drawing of a tree. The algorithm in [5] finds a box-drawing of a tree in time $O(n^2)$ if the tree has $n$ nodes, but the drawing found is not always a compact drawing. Walker [9] gave a linear-time algorithm with a set of aesthetic conditions similar to that of Reingold and Tilford [6], where the sizes of all the nodes are variable in only one direction (i.e., either in the horizontal or in the vertical direction) and fixed in the other direction [9].
straightforward implementation of the algorithms in [6] and [9] for nodes with variable sizes is not trivial and it would lead to unaesthetic drawings [1]. Bloesch [1] gave an algorithm for the box-drawing of trees with a similar type of aesthetic conditions, but its time complexity is not linear in the number of nodes and it does not always give a compact drawing. In contrast, our algorithm is very simple and always gives a compact box-drawing of trees in linear time under the given aesthetic conditions.

The rest of the paper is organized as follows: Section 2 introduces some definitions and known results. Section 3 gives our main result on compact box-drawings of trees. Finally, Section 4 is the conclusion.

2. PRELIMINARIES

In this section, we give some definitions and present some known results.

In this paper, a tree $T$ means the so-called rooted ordered tree, in which there is an ordering of the children of each node in $T$.

We call a drawing of a tree a box-drawing if each node of the tree is drawn as an axis-parallel rectangular box and the drawing satisfies the following three aesthetic conditions:

(C1) No two boxes overlap each other;
(C2) All boxes corresponding to siblings have the same x-coordinate at their left sides, and the x-coordinate is equal to that of their parent plus the width of the parent; and

(C3) A parent node is drawn at a given distance vertically apart from its first child: The y-coordinate of the top side of a parent is smaller than that of its first child by a given integer.

Similarly as in [5], a box is not drawn in the exact scale of the prescribed size, but the width and the height of a drawn box are less than the prescribed size by one-half of one coordinate unit. This amount of gap is used to show the interconnection among a parent and its children. A box is placed in the grid in such a manner that the top-left corner has an integral coordinate value but the bottom-right corner has a half-integral coordinate value. The interconnection among a parent and its children is represented by horizontal and vertical line segments as illustrated in Figure 1(b,c). A horizontal line starts from the point which is on the right side of a parent box and is one-fourth of one coordinate unit lower from the top-right corner of the box, and ends on a vertical line at the middle of the gap. From the vertical line, each child box is connected by a horizontal line segment at one-half of one coordinate unit. This amount of gap is used to show the interconnection among a parent and its children.

3. COMPACT DRAWING

In this section, we give a linear-time algorithm for finding a compact box-drawing of a tree \( T \). It suffices to give a linear-time algorithm for finding a compact box-drawing from an arbitrary box-drawing \( \Gamma \) of \( T \), for example, the initial box-drawing which can be found in linear time by the first phase of the algorithm in [5].

Any box-drawing is compact in the \( x \)-direction. Therefore, our task is to compact the box-drawing \( \Gamma \) in the \( y \)-direction. Our idea is as follows: Let \( l \) be the number of leaves of tree \( T \), and let \( p_1, p_2, \ldots, p_l \) be the ordering of all leaves of \( T \) obtained by a preorder tree-traversal. For the tree \( T \) in Figure 1(a), \( l = 14 \) and all the leaves are ordered as follows: nodes 22, 23, 18, 11, 12, 6, 24, 26, 27, 14, 20, 21, 16, and 9. We decompose tree \( T \) into \( l \) node-disjoint paths \( P_1, P_2, \ldots, P_l \). \( P_i \) starts at the root and ends at leaf \( p_i \); and each path \( P_i \), \( 2 \leq i \leq l \), starts at the furthest ancestor of \( p_i \) not in any of paths \( P_1, P_2, \ldots, P_{i-1} \) and ends at leaf \( p_i \). In Figure 1, path \( P_1 \) contains boxes 1, 2, 5, 10, 17, and 22; paths \( P_2, P_3, P_4, P_5, \) and \( P_6 \) contain boxes 23, 18, 11, 12, 6, respectively; and path \( P_7 \) contains boxes 3, 7, 13, 19, and 24. For each \( i, 1 \leq i \leq l \), we denote by \( T_i \) the subtree of \( T \) induced by the \( i \) paths \( P_1, P_2, \ldots, P_i \). The relative positions of all boxes on each path \( P_i \) are fixed in all box-drawings of \( T \), due to Conditions (C2) and (C3). Therefore, the positions of all boxes on \( P_i \) in \( \Gamma \) need not be changed in a compact drawing. We determine the positions of boxes on \( P_2, P_3, \ldots, P_l \) in this order. For each \( i \geq 2 \), all boxes on \( P_i \) should be moved upward by the same amount of distance for compaction because their relative positions are fixed. The distance is denoted by \( \text{dist}(P_i) \). To compute \( \text{dist}(P_i) \), one wishes to know the positions of boxes in the subtree \( T_{i-1} \) which can be seen from boxes on \( P_i \). We construct a “visibility graph” for this purpose.

Two boxes \( u \) and \( v \) in \( \Gamma \) are said visible from each other if a vertical line segment can be drawn so that it connects \( u \) and \( v \) but does not intersect or touch any other box in \( \Gamma \). The upward visibility graph \( G_{up} \) of \( \Gamma \) is defined as follows: Each box \( b \) in \( \Gamma \) corresponds to a vertex \( v_b \) in \( G_{up} \) and there is a directed edge connecting vertex \( v_b \) to vertex \( v_{b'} \) in \( G_{up} \) if and only if in \( \Gamma \) box \( b' \) is visible from box \( b \) and the y-coordinate of the top side of \( b' \) is greater than that of \( b \).

The upward visibility graph \( G_{up} \) is directed because boxes are drawn with a given distance vertically apart. The upward visibility graph of any box-drawing of a tree is planar. We now have the following lemma:

**Lemma 1.** The upward visibility graph \( G_{up} \) of a box-drawing \( \Gamma \) of a tree \( T \) can be constructed in linear time.

**Proof.** We give an algorithm to construct \( G_{up} \) in linear time as follows:

We first find the paths \( P_1, P_2, \ldots, P_l \). This can be done in linear time. We then traverse the paths \( P_1, P_2, \ldots, P_l \) in this order and construct the visibility graph \( G_{up} \). For each \( i, 1 \leq i \leq l \), we maintain a list of all boxes in \( T_i \) that can see
Theorem 1. A compact box-drawing of a tree can be obtained in linear time.

Proof. The box-drawing $\Gamma$ of tree $T$ is optimally compacted in the $x$-direction. Our task is to compact $\Gamma$ in the $y$-direction. For this, we give a linear-time algorithm as follows:

We first find paths $P_1, P_2, \ldots, P_l$. This can be done in linear time. The positions of all boxes on $P_1$ in $\Gamma$ need not be changed in a compact drawing. We now assume that the positions of all boxes in $T_{i-1}$, $i \geq 2$, have already been determined for a compact drawing. We now determine the positions of all boxes on $P_i$.

Let $u$ be a box on $P_i$, let $k$ be the out-degree of $u$ in $G_{\text{up}}$, and let $v_1, v_2, \ldots, v_k$ be the head nodes of edges going out from node $u$ in $G_{\text{up}}$. Let $\text{dist}(u, v_i)$ be the vertical distance from the top side of box $u$ to the bottom side of box $v_i$, $1 \leq i \leq k$. Then, the possible upward movement $\text{um}(u)$ of box $u$ such that $u$ will not overlap any box in $T_{i-1}$ is $\min\{\text{dist}(u, v_1), \text{dist}(u, v_2), \ldots, \text{dist}(u, v_k)\}$. The distance $\text{dist}(P_i)$ of the possible upward movement of boxes on $P_i$ is equal to the minimum of possible upward movements of all boxes $u$ on $P_i$; $\text{dist}(P_i) = \min\{\text{um}(u)\}$ $u$ is a node on $P_i$. We uniformly move upward all boxes on $P_i$ by the distance $\text{dist}(P_i)$.

By Lemma 1, the visibility graph $G_{\text{up}}$ can be constructed in time $O(n)$. Using $G_{\text{up}}$, one can compute the possible upward movement $\text{um}(u)$ of a box $u$ on $P_i$ in time proportional to the out-degree of $u$ in $G_{\text{up}}$. Since $G_{\text{up}}$ is planar, the sum of out-degrees of all vertices in $G_{\text{up}}$ is at most $3n$, and, hence, the possible upward movements of all boxes in $\Gamma$ can be computed in time $O(n)$. One can compute $\text{dist}(P_i)$, $1 \leq i \leq l$, from the possible upward movements of all boxes on $P_i$ in time $O(\{|V(P_i)|\})$; and, hence, one can compute $\text{dist}(P_i)$ for $i = 1, 2, \ldots, l$ in time $O(\sum_{i=1}^{l}\{|V(P_i)|\}) = O(n)$. Thus, the algorithm takes time $O(n)$.

Clearly, the algorithm above obtains a compact box-drawing of $T$. 

We then have the following theorem:

FIG. 2. (a) A box-drawing $\Gamma$ of a tree and (b) its upward visibility graph $G_{\text{up}}$. 

---

$-\infty$ in the $y$-direction; the list contains boxes in the increasing order of $x$-coordinates of their left sides and the list is called the lower envelope $\text{LE}(T_i)$ of $T_i$. An entry in the lower envelope is not always a “full” box, but may be a part of a box which can see $-\infty$ in the $y$-direction. Clearly, the visibility graph of a drawing of the subtree $T_i$ consists of isolated vertices corresponding to the boxes on $P_i$, and $\text{LE}(T_i)$ consists of all (full) boxes on $P_i$.

Assume that we have constructed the visibility graph of a drawing of the subtree $T_{i-1}$, $i \geq 2$, and found the lower envelope $\text{LE}(T_{i-1})$. We now find the edges of $G_{\text{up}}$ going out from boxes on $P_i$. Let $u$ be the starting box of $P_i$, and let $v$ be the sibling of $u$ immediately above $u$. Clearly, $v$ is in $\text{LE}(T_{i-1})$. We find all edges of $G_{\text{up}}$ going out from boxes on $P_i$ by comparing the $x$-coordinates of top-left and top-right corners of boxes on $P_i$ with those of (full or partial) boxes in $\text{LE}(T_{i-1})$ starting from $v$ in a manner of comparing two sorted lists for merging. We add the found edges to the visibility graph for $T_{i-1}$ to obtain the visibility graph for $T_i$. Simultaneously, we update the lower envelope $\text{LE}(T_{i-1})$ to $\text{LE}(T_i)$; all (full) boxes on $P_i$ appear in the lower envelope $\text{LE}(T_i)$ and several consecutive boxes starting from $v$ in $\text{LE}(T_{i-1})$ and possibly a left part of the last one may be hidden by the boxes on $P_i$ and must disappear in $\text{LE}(T_i)$. For the box-drawing in Figure 1(b), the lower envelope $\text{LE}(T_{10})$ of $T_{10}$ is 1, 3, 7, 14, 13, 19, 25, 27, and 24. When we update $\text{LE}(T_{10})$ to $\text{LE}(T_{11})$, boxes 15 and 20 appear in the list and boxes 14 and 13 disappear in $\text{LE}(T_{11})$.

All the boxes in $\text{LE}(T_{j-1})$ that are compared once disappear in $\text{LE}(T_i)$ except the last compared box, a right part of which may remain in $\text{LE}(T_i)$. Therefore, the number of comparisons required throughout the execution of the algorithm is $O(\sum_{i=1}^{l}(|V(P_i)| + 1)) = O(n + l) = O(n)$, where $n$ is the number of nodes in $T$ and $|V(P_i)|$ is the number of nodes in $P_i$. Thus, we can construct the visibility graph $G_{\text{up}}$ in time $O(n)$.

We then have the following theorem:
4. CONCLUSIONS

In this paper, we gave a linear-time algorithm for finding a compact box-drawing of a tree following some aesthetic conditions. The best-known previous algorithm under the same set of aesthetic conditions runs in $O(n^2)$ time [5], but does not always find a compact box-drawing.

REFERENCES