ELSEVIER

Computational Geometry 9 (1998) 25–42

Computational Geometry
Theory and Applications

Linear area upward drawings of AVL trees [☆]

P. Crescenzi [*,1], P. Penna, A. Piperno

Dipartimento di Scienze dell'Informazione, Università degli Studi di Roma "La Sapienza", Via Salaria 113, 00198 Roma, Italy

Communicated by G. Di Battista and R. Tamassia; submitted 19 January 1995; revised 31 October 1995 and 12 July 1996

Abstract

We prove that any AVL tree admits a linear-area straight-line strictly-upward planar grid drawing, that is, a drawing in which (a) each edge is mapped into a single straight-line segment, (b) each node is placed below its parent, (c) no two edges intersect, and (d) each node is mapped into a point with integer coordinates. © 1998 Elsevier Science B.V.

Keywords: Graph drawing; Upward drawing; Area requirement

1. Introduction

In several applications, information is better displayed by a graphical representation emphasizing its structure in a readable way: a list of areas in which these applications arise includes software engineering, project management, and knowledge representation. The automatic design of these graphical representations is one of the main motivations for the growing interest in the research area of *graph drawing* whose typical problem is the following: given a graph G, produce a geometric representation of G according to some graphic standards and optimization criteria.

Several graphic standards and optimization criteria have been proposed in the literature depending on the application at hand. The annotated bibliography maintained by Di Battista, Eades, Tamassia and Tollis mentions most of them and refers to more than 300 papers in this research area [3]. In this paper we are interested in *straight-line strictly-upward planar grid drawings*, in short *upward drawings*, of rooted binary trees, that is, drawings in which each edge is mapped into a single straight-line segment, each node is placed below its parent, no two edges intersect, and each node is mapped into a point with integer coordinates. Each of these standards are justified by "readability" considerations. First, straight-line segments are easier to read than chains of segments or curves. Second, the upward requirement

[☆] A preliminary version of this paper was presented at Graph Drawing '94.

[*] Corresponding author. E-mail: piluc@dsi.uniroma1.it.

[1] Research partially supported by the MURST project Efficienza di Algoritmi e Progetto di Strutture Informative.

effectively visualizes the hierarchical structure of the tree. Third, it is natural to assume that if a graph can be drawn without any pair of crossing edges, then we draw it accordingly. Finally, the integer coordinates for vertices constitute a good resolution rule both for the computer display device and for the human eye.

A natural and important optimization criterion for evaluating these drawings is that they take as little area as possible, where the area of a drawing equals the area of the smallest isothetic rectangle bounding the drawing. This criterion belongs to the family of the so-called *aesthetic* criteria which are based on the fact that some drawings are better than others in conveying information regarding the tree.

1.1. Previous results

Most of the known algorithms to produce a straight-line strictly-upward planar grid drawing of a binary tree require quadratic area in the worst case [10,12]. The first $\mathrm{O}(n \log n)$-area algorithm appeared in [11] and recently it has been proved that this algorithm is optimal. In particular, an infinite family of binary trees requiring $\Omega(n \log n)$-area in order to be upward drawn has been shown in [2] (it is worth observing that, if we relax the upwardness requirement, then any binary tree of n nodes admits a linear-area planar grid drawing [13]). In [2] the authors also gave two algorithms producing a linear-area upward drawing of complete and Fibonacci binary trees, respectively. Subsequently, it has been proved that if we allow an edge to be represented by a chain of straight-line segments and a node to be on the same horizontal line as its parent, then *any* binary tree can be drawn in linear area [5].

1.2. Our results

The main result of this paper is that, for any AVL tree t with n nodes, an upward drawing of t can be produced with area $\mathrm{O}(n)$ in time $\mathrm{O}(n)$. In particular, in Sections 2 and 3 we prove that, for any constant $\alpha > 1$, a constant κ exists such that any AVL tree with n nodes can be upward drawn in any rectangle whose shorter side is at least $\log^{\alpha} n$ and whose area is equal to κn. This result can be extended to 2-balanced trees.

Our result improves on that obtained in [2] since both complete trees and Fibonacci trees are AVL trees: indeed, the Fibonacci trees are the AVL trees with the least number of vertices while the complete trees are the AVL trees with the greatest number of vertices. It also improves on the result obtained in [5] (when applied to AVL trees) in two directions. On the one hand, our algorithm produces straight-line strictly-upward drawings, on the other, the bound on the length of the shorter side provides a greater flexibility to applications that need to fit the drawing in a prescribed rectangular region.

On the negative side, the theoretical constant factor in the area bound turns out to be very large. In the case $\alpha = 1.1$, we have that $\kappa \approx 5684$ and this value increases as α decreases. For this reason, in Section 4 we slightly restrict the flexibility of our approach in order to perform a better analysis of the algorithm. In particular, we will be able to prove that any AVL tree of height h with n nodes can be upward drawn in any rectangle whose area is equal to $36n$ and whose shorter side is at least $\log n$ if $h \leqslant 30$, $n^{1/4}$ otherwise.

Finally, in Section 5 we present some experimental results which illustrate how, in practice, the area requirements are much less than those specified by the theoretical results. In particular, we have implemented a slight modification of our algorithm and we have obtained experimental results for complete binary trees, Fibonacci trees, and combinations of these two kinds of trees: these experiments show that the multiplicative factor in the area bound is approximately 6. Moreover, they show that even from a practical point of view our algorithm improves on the one proposed in [5].

1.3. Preliminaries

In this section we give preliminary definitions and results that will be used throughout the paper.

We refer to directed rooted unordered binary trees, in short binary trees. We denote by $\emptyset$ the empty binary tree. Given two binary trees t_1 and t_2, we denote by $t_1 \oplus t_2$ the binary tree whose immediate subtrees are t_1 and t_2. In particular, c_h denotes the *complete binary tree* of height h while F_h denotes the *Fibonacci tree* of height h [7–9]. We recall that these trees are inductively defined in the following way:

$$c_h = \begin{cases} \emptyset, & \text{if } h = 0, \\ c_{h-1} \oplus c_{h-1}, & \text{otherwise}, \end{cases} \tag{1}$$

and

$$F_h = \begin{cases} \emptyset, & \text{if } h = 0, \\ \emptyset \oplus \emptyset, & \text{if } h = 1, \\ F_{h-1} \oplus F_{h-2}, & \text{otherwise}. \end{cases} \tag{2}$$

We denote by $n_c(h)$ and $n_F(h)$ the number of nodes of c_h and F_h, respectively.

A binary tree is said to be *k-balanced* if, for each node u, the heights of the two immediate subtrees of u differ by at most k. A 1-balanced binary tree is also called an *AVL* tree[2] and it is well known that, for any AVL tree t of height h, $n_F(h) \leqslant n \leqslant n_c(h)$, where n denotes the number of nodes of t [1].

A *straight-line strictly-upward planar grid drawing*, in short *upward drawing*, of a binary tree t is a drawing of t such that:

(i) Edges are straight-line segments.
(ii) Each node has an ordinate greater than that of its parent (we are thus assuming that the y-axis is downward oriented).
(iii) Edges do not intersect.
(iv) Nodes are points with integer coordinates.

The width (respectively, height) of a drawing is the width (respectively, height) of the smallest isothetic rectangle bounding the drawing. We adopt the convention that both the width and the height are measured by the number of grid points, so that any drawing of a nonempty binary tree has both width and height greater than zero. The *area* of a drawing is then defined as the product of the width and the height.

Theorem 1 [2]. *An algorithm exists producing an upward drawing of either a complete binary tree or a Fibonacci binary tree with n nodes in* $\mathrm{O}(n)$ *area.*

In order to prove the above theorem, the notion of *h–v drawing* is introduced. An h–v drawing is an upward drawing in which only rightward-horizontal and downward-vertical straight-line segments

[2] Indeed, AVL trees are usually defined as 1-balanced binary search trees. Since we are not concerned with the labels of the nodes, we are essentially using the term AVL tree to denote the "skeleton" of the tree.

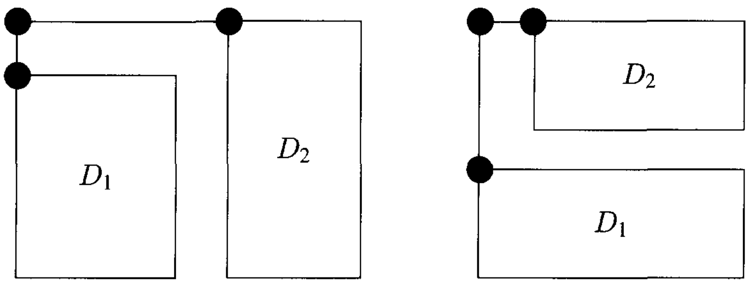

Fig. 1. The two operations of an h–v drawing

are allowed. That is, the notion of h–v drawing is a restriction of that of *orthogonal drawing*[3] in which each edge is a chain of alternating horizontal and vertical segments [3]: on the one hand, each edge can be only one segment, on the other, no leftward-horizontal or upward-vertical segments are allowed.

More precisely, an h–v drawing of a non-empty binary tree $t = t_1 \oplus t_2$ is obtained by one of the two operations illustrated in Fig. 1, where D_1 and D_2 are two h–v drawings of t_1 and t_2, respectively. In the first operation, that is, the horizontal operation, D_2 is translated to the right by as many grid points as the width of D_1 and D_1 is translated to the bottom by one grid point. The semantics of the second operation, that is, the vertical operation, is defined similarly.

The following fact shows that h–v drawings are a powerful tool in dealing with upward drawings.

Proposition 2 [2]. *Any h–v drawing of width w and height h with $w \leqslant h$ can be transformed in linear time into an upward drawing of width w and height at most $w + h$.*

The above result will allow us to devote our attention to h–v drawings only (note that we are interested in upper bounds on the area requirements). In [4], finally, an algorithm is given yielding a minimum area h–v drawing of a binary tree with n nodes in time $\mathrm{O}(n\sqrt{n \log n})$.

2. The technique

Let us reconsider the approach of [2] which mainly uses the inductive definitions of complete and Fibonacci binary trees (see Eqs. (1) and (2)). In particular, according to these definitions c_h must be "combined" just with itself in order to produce c_{h+1} while F_h must be "combined" with F_{h-1} in order to produce F_{h+1} and with F_{h+1} in order to produce F_{h+2}. This suggests that a bottom-up approach to drawing complete and Fibonacci binary trees, that is, an approach consisting of drawing one of these trees by using the drawings produced for the corresponding smaller-height trees, will need one drawing for each c_h and two drawings for each F_h.

Any attempt to extend such a bottom-up construction to the case of AVL trees leads to the necessity of producing an exponential number of drawings for any tree. Indeed, let us fix an AVL tree t of height h along with a drawing of t. An AVL tree of height $h + 1$ having t as one of its subtrees can

[3] In [11] orthogonal drawings are called D_0-L-arrangements.

be obtained in many different ways depending on the choice of the other subtree. If the drawing of this subtree must be produced so that, when combined with the fixed drawing of t, the linear-area requirement is maintained, then it is clear that the same drawing cannot be used in combination with the drawings of all trees of height h, which could be considerably different from t (for example, in the number of nodes).

To overcome this difficulty, we shall instead follow a top-down approach. From the definition of the h–v drawing operations (recall that our attention can be restricted to just h–v drawings), any rectangle including a drawing for an AVL tree t of height h must contain two rectangles including the drawings of the immediate subtrees of t. From an algorithmic point of view, given a rectangle R in which we want to draw t, we must be able to cut it into two rectangles in which it is possible to draw the immediate subtrees of t. Our cut-rule can be roughly described as follows: *cut R in proportion to the number of nodes of the two subtrees*. Two problems arise from this rule. On the one hand, we need to maintain the linear-area requirement, on the other, we must be able to treat "highly rectangular" shapes (for example, consider the case in which one subtree is a Fibonacci tree of height $h-2$ while the other is a complete tree of height $h-1$).

This section is devoted to the study of the conditions under which this construction can be safely carried out. Given an AVL tree, we shall denote by n the number of its nodes and by l and L the length of the shorter and the longer side of the rectangle in which the tree has to be drawn, respectively. Intuitively, we shall prove that, if l and lL are "large enough", then there is a cutting of the rectangle which preserves these same desired properties on the sides of the two obtained subrectangles.

Our proofs will refer to rectangles with real coordinates. However, it is clear that if we can draw a tree within a real-coordinate rectangle R by mapping nodes into points with integer coordinates, then the tree itself can be drawn within the largest integer-coordinate rectangle included in R.

In order to prove the main result of this section, let us first introduce some functions together with properties they are required to satisfy (in the following $\mathbb{N}^+$ and $\mathbb{R}^+$ will denote the set of positive integer numbers and the set of positive real numbers, respectively).

(i) The *factor* function

$$k:\mathbb{N}^+ \to \mathbb{R}^+,$$

that is, the function specifying the multiplicative factor in the area bound, which should be a non-decreasing function which satisfies the following property.

Property 1. *A constant κ exists such that, for any h, $k(h) \leqslant \kappa$.*

(ii) The *shorter side* function

$$l:\mathbb{N}^+ \to \mathbb{R}^+,$$

that is, the function specifying the lower bound for l, which should satisfy the following property.

Property 2. $l(1) = 1$ *and, for any h,* $l(h+1) \geqslant l(h) + 1$.

(iii) The *area* function

$$A:\mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{R}^+,$$

that is, the function specifying the area of the drawing, is defined as

$$A(h,n) = k(h)n$$

and should satisfy the following property.[4]

Property 3. *For any $h \geqslant 2$ and for any n_1, n_2 such that $n_F(h-1) \leqslant n_1 \leqslant n_2 \leqslant n_c(h)$,*

$$A(h,n_1) \geqslant \sqrt{A(h+1,n)}\, l(h),$$

where $n = n_1 + n_2 + 1$.

Observe that if $l \geqslant l(h)$, then the length L of the longer side is at most

$$L(h,n) = \frac{A(h,n)}{l(h)}.$$

Clearly, if an AVL tree of height h with n nodes exists, then $L(h,n)$ should be at least equal to $l(h)$. This is guaranteed, for $h = 1$, by taking $k(1) \geqslant 1$ and, for $h > 1$, by the following proposition.

Proposition 3. *For any $h \geqslant 2$ and for any n such that $n_F(h) \leqslant n \leqslant n_c(h)$,*

$$L(h,n) > l(h).$$

Proof. By using Property 3 with $n_1 = n_2$, we have

$$\frac{L(h,n)}{l(h)} = \frac{A(h,n)}{l^2(h)} \geqslant A(h,n)\frac{A(h+1,2n+1)}{A^2(h,n)} = \frac{A(h+1,2n+1)}{A(h,n)}.$$

From the assumption of k being non-decreasing, it thus follows that $L(h,n) > l(h)$. □

Finally, the area function should satisfy the following property.

Property 4. *For any h and for any n_1, n_2 with $n_1 \leqslant n_2$,*

$$A(h+1,n) \geqslant A(h,n_1) + A(h,n_2) + \frac{A(h+1,n) - A(h,n_2)}{l(h+1)},$$

where $n = n_1 + n_2 + 1$.

We are now in a position to show that any AVL tree of height h with n nodes can be h–v drawn in any rectangle whose shorter side and whose area are large enough.

Theorem 4. *Let $k(\cdot)$ and $l(\cdot)$ be two functions satisfying Properties 2–4, let $h, n \in \mathbb{N}^+$, and let R be any rectangle whose sides have lengths l and L, respectively, satisfying the following two conditions:*

$$L \geqslant l \geqslant l(h) \quad \text{and} \quad lL = A(h,n).$$

Then any AVL tree of height h with n nodes admits an h–v drawing within the rectangle R.

[4] Since $A(h,n)$ is completely determined by $k(h)$ and n, the next two properties are really conditions on functions $k(\cdot)$ and $h(\cdot)$ rather than on function $A(\cdot,\cdot)$.

Proof. The proof is by induction on h. For $h = 1$, the proof is straightforward.

Let $h \geqslant 1$ and let us assume that the theorem is true for any height less than $h + 1$. Given an AVL tree t of height $h + 1$ with n nodes, let us define

$$l_1 = L - \frac{A(h, n_2)}{l} \quad \text{and} \quad l_2 = L - l_1,$$

where n_2 denotes the number of nodes of the larger immediate subtree of t. Observe that if we denote by n_1 the number of nodes of the smaller immediate subtree of t, then $n = n_1 + n_2 + 1$ and $n_F(h-1) \leqslant n_1 \leqslant n_2 \leqslant n_c(h)$.

The root of t is mapped into the grid point whose coordinates are (x, y), where x and y denote the coordinates of the top leftmost corner of R. Let us assume that the longer side of R is the vertical one (the other case can be proved in a similar way). We then isolate two rectangles R_1 and R_2 within the rectangle R as follows (see Fig. 2). The top leftmost corners of R_1 and R_2 have coordinates $(x+1, y)$ and $(x, y + \lfloor l_1 \rfloor)$, respectively. The vertical side and the horizontal side of R_1 have length l_1 and $l - 1$, respectively, while the vertical side and the horizontal side of R_2 have length l_2 and l, respectively. Since the length of a segment is measured by the number of grid points, we thus have that the y-coordinate of the bottom rightmost corner of R_2 is equal to $y + \lfloor l_1 \rfloor + l_2 - 1$ which is less than or equal to the y-coordinate of the bottom rightmost corner of R, that is, $y + L - 1$.

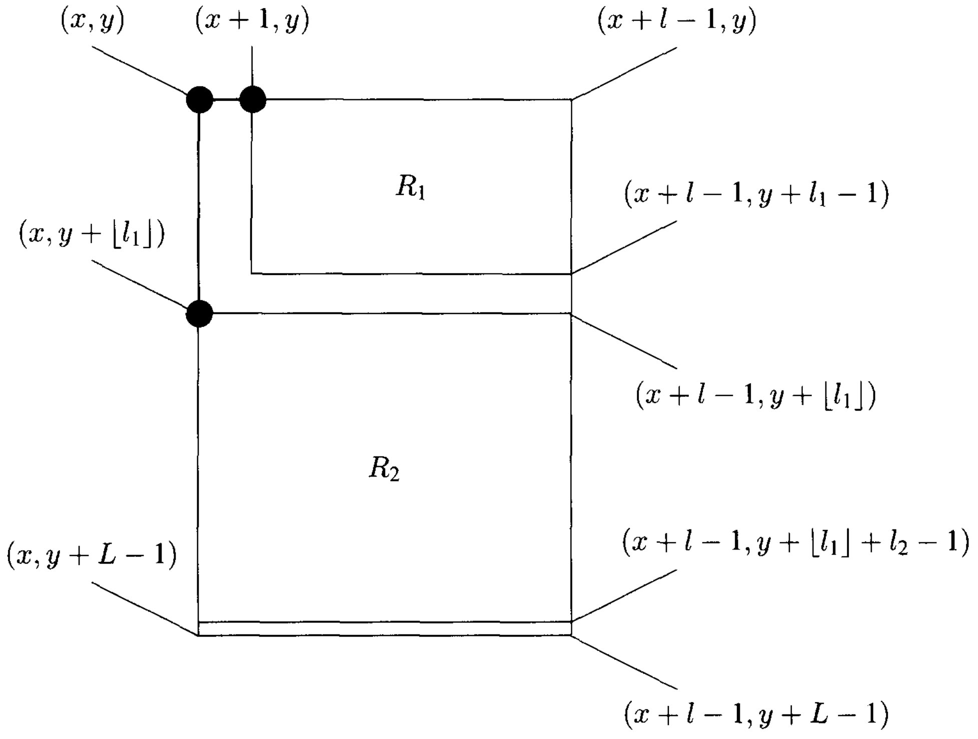
(x, y)
$(x+1, y)$
$(x+l-1, y)$
R_1
$(x+l-1, y+l_1-1)$
$(x, y+\lfloor l_1 \rfloor)$
$(x+l-1, y+\lfloor l_1 \rfloor)$
R_2
$(x, y+L-1)$
$(x+l-1, y+\lfloor l_1 \rfloor + l_2 - 1)$
$(x+l-1, y+L-1)$

Fig. 2. The splitting of a rectangle $R = \langle l, L, (x, y), \mathbf{true} \rangle$ (the exact notation is explained fully in the proof of Theorem 7).

Clearly, the area of R_2 is equal to $A(h, n_2)$. Moreover, since

$$L - \frac{A(h,n_2)}{l} = \frac{lL - A(h,n_2)}{l} = \frac{A(h+1,n) - A(h,n_2)}{l} \leqslant \frac{A(h+1,n) - A(h,n_2)}{l(h+1)},$$

we are then guaranteed that the area of R_1 is at least $A(h, n_1)$. Indeed,

$$\begin{aligned} l_1(l-1) &= \left(L - \frac{A(h,n_2)}{l}\right)(l-1) \\ &= A(h+1,n) - A(h,n_2) - \left(L - \frac{A(h,n_2)}{l}\right) \\ &\geqslant A(h+1,n) - A(h,n_2) - \frac{A(h+1,n) - A(h,n_2)}{l(h+1)} \\ &\geqslant A(h,n_1), \end{aligned}$$

where the last inequality is due to Property 4.

Let h_1 and h_2 denote the heights of the two subtrees with n_1 and n_2 nodes, respectively. We now shall prove that the shorter sides of R_1 and R_2 have length at least $l(h_1)$ and $l(h_2)$, respectively. To this aim, we distinguish the following three cases.

(i) $h_1 = h - 1$ and $h_2 = h$.

(a) Rectangle R_1. If $l_1 \geqslant l - 1$, then from Property 2 it follows that

$$l - 1 \geqslant l(h+1) - 1 \geqslant l(h) \geqslant l(h-1).$$

Otherwise,

$$l_1 \geqslant \frac{A(h,n_1)}{l-1} \geqslant l(h) \geqslant l(h-1),$$

where the second inequality follows from the fact that, since $l \leqslant L$ and $lL = A(h+1, n)$, then $l \leqslant \sqrt{A(h+1,n)}$ and from Property 3.

(b) Rectangle R_2. If $l_2 \geqslant l$, then from Property 2 it follows that

$$l \geqslant l(h+1) \geqslant l(h).$$

Otherwise, since $A(h, n_2) \geqslant A(h, n_1)$, we have that

$$l_2 = \frac{A(h,n_2)}{l} \geqslant \frac{A(h,n_1)}{l} \geqslant l(h),$$

where the last inequality follows from the fact that, since $l \leqslant L$ and $lL = A(h+1, n)$, then $l \leqslant \sqrt{A(h+1,n)}$ and from Property 3.

(ii) $h_1 = h$ and $h_2 = h - 1$. The proof is similar to that of case (i): note that in this case we are simply decreasing the upper bound for n_2 and increasing the lower bound for n_1.

(iii) $h_1 = h_2 = h$. The proof is similar to that of case (i): note that in this case we are simply increasing the lower bound for n_1.

In all three cases we have that the inductive hypothesis is satisfied for both a rectangle included in R_1 and a rectangle included in R_2. That is, the AVL subtree of height h_1 with n_1 nodes admits an h–v drawing within the rectangle R_1 and the AVL subtree of height h_2 with n_2 nodes admits an h–v drawing within the rectangle R_2. The theorem thus follows. □

Corollary 5. *If the function $k(\cdot)$ in the previous theorem also satisfies Property 1, then any AVL tree admits a linear-area h–v drawing.*

3. The algorithm

In this section we shall prove the existence of the functions $k(\cdot)$ and $l(\cdot)$ satisfying the four properties of the previous section. As a consequence, we will then be able to give a linear-time algorithm producing a linear-area h–v drawing of an arbitrary AVL tree.

The definition of the functions $k(\cdot)$ and $l(\cdot)$ is quite simple: in fact, $l(h) = h^\alpha$ and $k(h+1) = (1+i(h))k(h)$, where $\alpha > 1$ and $i(h)$ will be specified later. These definitions are motivated by two intuitive reasons: on the one hand, we do not want the shorter side being smaller than the height of the tree, on the other, the higher the tree, the bigger the multiplicative factor should be (even though asymptotically bounded by a constant).

Since $\alpha > 1$, we have that $(h+1)^\alpha \geqslant h^\alpha + 1$, that is, the function $l(h)$ satisfies Property 2.

Observe now that from the definition of $A(h,n)$ it follows that proving Property 4 is equivalent to proving that

$$\big[i(h)A(h,n) + k(h)\big]l(h+1) \geqslant A(h+1,n) - A(h,n_2),$$

where $n = n_1 + n_2 + 1$ and $n_1 \leqslant n_2$. Since $n > 2n_1$, we have that

$$\begin{aligned} &A(h+1,n) - A(h,n_2) \\ &\quad = A(h,n) + i(h)A(h,n) - A(h,n_2) = A(h,n_1) + k(h) + i(h)A(h,n) \\ &\quad < A(h,n)/2 + i(h)A(h,n) + k(h) = i(h)A(h,n)\big[1 + 1/2i(h)\big] + k(h) \\ &\quad \leqslant \big[i(h)A(h,n) + k(h)\big]\big[1 + 1/2i(h)\big]. \end{aligned}$$

We then need to define $i(h)$ so that $1 + 1/2i(h) \leqslant l(h+1)$. From Property 2 it follows that we can simply define $i(h) = 1/2l(h)$.

In order to prove Property 3, we have to show that, for any $h \geqslant 2$,

$$k^2(h)n_1^2 \geqslant A(h+1,n)h^{2\alpha} = k(h)n\left(\frac{2h^\alpha + 1}{2h^\alpha}\right)h^{2\alpha},$$

where $n = n_1 + n_2 + 1$ and $n_F(h-1) \leqslant n_1 \leqslant n_2 \leqslant n_c(h)$. This is clearly true if

$$k(h)\frac{n_1^2}{n} \geqslant \frac{5}{4}h^{2\alpha}.$$

Since n_1^2/n is minimum when n_1 is minimum and n_2 is maximum, we thus have to prove that

$$k(h)\frac{1}{\rho(h)} \geqslant \frac{5}{4}h^{2\alpha},$$

where, for any $h > 1$,

$$\rho(h) = \frac{n_F(h-1) + n_c(h) + 1}{n_F^2(h-1)}.$$

Table 1
The values of $\rho(h)/\rho(h+1)$ and $2(h+1)^{2\hat{\alpha}}/(2h^{2\hat{\alpha}}+h^{\hat{\alpha}})$ for $h=2,\ldots,7$

	h					
	2	3	4	5	6	7
$\rho(h)/\rho(h+1)$	2.00	2.00	1.57	1.50	1.42	1.39
$2(h+1)^{2\hat{\alpha}}/(2h^{2\hat{\alpha}}+h^{\hat{\alpha}})$	2.00	1.66	1.49	1.39	1.32	1.28

We shall prove the above inequality by induction on h and by suitably choosing α and $k(1)$. For $h=2$, if we set $k(1)$ equal to $\frac{25}{6}\cdot 2^{2\alpha}$, then

$$k(2)\frac{1}{\rho(2)} = \frac{3}{2}k(1)\frac{1}{5} = \frac{5}{4}2^{2\alpha}.$$

Let us assume that the inequality is satisfied for any height less than $h+1$. Then

$$\frac{k(h+1)}{\rho(h+1)} = \left(1+\frac{1}{2h^{\alpha}}\right)\frac{k(h)}{\rho(h+1)} \geqslant \left(1+\frac{1}{2h^{\alpha}}\right)\frac{5}{4}h^{2\alpha}\frac{\rho(h)}{\rho(h+1)}.$$

In order to satisfy the inequality for $h+1$, we must then define α so that

$$\left(1+\frac{1}{2h^{\alpha}}\right)h^{2\alpha}\frac{\rho(h)}{\rho(h+1)} \geqslant (h+1)^{2\alpha},$$

that is,

$$\frac{\rho(h)}{\rho(h+1)} \geqslant \frac{2(h+1)^{2\alpha}}{2h^{2\alpha}+h^{\alpha}}.$$

If $h=2$, then $\rho(h)/\rho(h+1)=2$ and α can be any value between 1 and $\widehat{\alpha}$, where $\widehat{\alpha}\approx 1.111$ is the solution of the following equation: [5]

$$2\cdot 2^{2x}+2^{x}-3^{2x}=0.$$

Since, for any h, $(2(h+1)^{2\alpha})/(2h^{2\alpha}+h^{\alpha})$ is an increasing function in the variable α, we then have to prove that

$$\frac{\rho(h)}{\rho(h+1)} \geqslant \frac{2(h+1)^{2\widehat{\alpha}}}{2h^{2\widehat{\alpha}}+h^{\widehat{\alpha}}}.$$

Table 1 shows the values of the left-hand side and the right-hand side of this inequality, respectively, for $h=2,\ldots,7$. Observe that, for $h\geqslant 7$, the right-hand side is smaller than 1.3; the following fact shows that the left-hand side is always greater than 1.3.

Fact 6. *For any h, $\rho(h)/\rho(h+1) > 1.3$.*

[5] The computation of this solution and other mathematical manipulations have been performed using *Mathematica*, a trademark of Wolfram Research Inc.

Finally, it remains to show that the function $k(h)$ satisfies Property 1. To this aim, first observe that, for any $h > 1$,

$$k(h) = \frac{25}{6} \cdot 2^{2\alpha} \cdot \prod_{i=1}^{h-1} \left(1 + \frac{1}{2i^\alpha}\right).$$

Moreover, we have that

$$\ln \prod_{i=1}^{h-1} \left(1 + \frac{1}{2i^\alpha}\right) = \sum_{i=1}^{h-1} \ln\left(1 + \frac{1}{2i^\alpha}\right) \leqslant \frac{1}{2} \sum_{i=1}^{h-1} \frac{1}{i^\alpha} < \frac{1}{2}\zeta(\alpha),$$

where the first inequality follows from $\ln(1+x) \leqslant x$ for all x and ζ denotes the Riemann zeta function defined as

$$\zeta(\alpha) = \sum_{i=1}^{\infty} \frac{1}{i^\alpha}.$$

We have thus shown that

$$k(h) < \frac{25}{6} \cdot 2^{2\alpha} \cdot \mathrm{e}^{\zeta(\alpha)/2}.$$

Since it is well-known that, for any $\alpha > 1$, $\zeta(\alpha) < \infty$ [6] (even though no closed form for the Riemann zeta function is known), it thus follows that $k(h)$ satisfies Property 1.

We are now ready to prove the main result.

Theorem 7. *For any constant $\alpha > 1$, a constant κ exists such that any AVL tree with n nodes can be h–v drawn in any rectangle whose shorter side is at least $\log^\alpha n$ and whose area is equal to κn. Moreover, this drawing can be produced in time $\mathrm{O}(n)$.*

Proof. For any constant $\alpha > 1$, let us define

$$\kappa = \begin{cases} \frac{25}{6} \cdot 2^{2\alpha} \cdot \mathrm{e}^{\zeta(\alpha)/2}, & \text{if } \alpha \leqslant \widehat{\alpha}, \\ \frac{25}{6} \cdot 2^{2\widehat{\alpha}} \cdot \mathrm{e}^{\zeta(\widehat{\alpha})/2}, & \text{otherwise.} \end{cases}$$

Our algorithm is then Algorithm 1 (see also Fig. 2), where a rectangle is specified by the lengths l and L of its sides, by the coordinates x and y of its top leftmost corner which is always assumed to be a point with integer coordinates, and by a Boolean flag b which indicates the orientation of the longer side (if b is true then the longer side is vertical, otherwise it is horizontal).

From the proof of Theorem 4 and from Corollary 5, it follows that, for any AVL tree t with n nodes and for any rectangle R whose shorter side is at least $\log^\alpha n$ and whose area is equal to κn, the output of $BT(R, t)$ is an h–v drawing of t within R.

In order to analyze the time complexity of the algorithm, we may assume that a preprocessing has been performed to compute, for any node u, the height and the number of nodes of the subtree rooted at u. Clearly, this preprocessing requires linear time. After that, it is easy to see that the algorithm visits each node of the tree exactly once and that, for any node, it performs a constant number of operations. It thus follows that the time complexity of the algorithm is $\mathrm{O}(n)$. □

Algorithm 1. The algorithm to draw an AVL tree

```
procedure $BT(R = \langle l, L, (x,y), b\rangle, t)$;
{construct upward drawing of AVL tree $t$ within rectangle $R$}
begin
    $h \leftarrow$ height of $t$;
    $t_1 \leftarrow$ subtree of $t$ with the smaller number of nodes;
    $t_2 \leftarrow$ subtree of $t$ with the larger number of nodes;
    $n_2 \leftarrow$ number of nodes of $t_2$;
    $l_1 \leftarrow L - A(h-1, n_2)/l$;
    $l_2 \leftarrow L - l_1$;
    if $b$ then $(x_1, y_1) \leftarrow (x+1, y)$ else $(x_1, y_1) \leftarrow (x, y+1)$;
    if $l_1 > l - 1$ then $R_1 \leftarrow \langle l-1, l_1, (x_1, y_1), b\rangle$
                    else $R_1 \leftarrow \langle l_1, l-1, (x_1, y_1), \textbf{not } b\rangle$;
    if $b$ then $(x_2, y_2) \leftarrow (x, y + \lfloor l_1 \rfloor)$ else $(x_2, y_2) \leftarrow (x + \lfloor l_1 \rfloor, y)$;
    if $l_2 > l$ then $R_2 \leftarrow \langle l, l_2, (x_2, y_2), b\rangle$
                else $R_2 \leftarrow \langle l_2, l, (x_2, y_2), \textbf{not } b\rangle$;
    map the root of $t$ into $(x, y)$;
    if $t_1 \neq \emptyset$ then $BT(R_1, t_1)$;
    if $t_2 \neq \emptyset$ then $BT(R_2, t_2)$;
end.
```

The next corollary follows from the above theorem and from Proposition 2 (observe that since we are considering unordered trees, we can always assume that the shorter side is the horizontal one).

Corollary 8. *For any constant $\alpha > 1$, a constant κ exists such that any AVL tree with n nodes can be upward drawn in any rectangle whose shorter side is at least $\log^{\alpha} n$ and whose area is equal to κn. Moreover, this drawing can be produced in time $O(n)$.*

4. Improving on the constant factor

According to the results of the previous section, we have that, for any $\alpha \geqslant \widehat{\alpha}$, the constant factor in the area bound for h–v drawings is equal to

$$\tfrac{25}{6} \cdot 2^{2\widehat{\alpha}} \cdot e^{\zeta(\widehat{\alpha})/2} \approx 2842$$

(observe that if α is smaller than $\widehat{\alpha}$ then this bound increases since the Riemann function converges more slowly). To obtain an upward drawing, according to Proposition 2, this bound must be doubled.

In order to improve this bound and to perform a better analysis of our algorithm, we will proceed along two directions: on the one hand, we will weaken its flexibility for AVL trees of big height, on the other hand we will deal with AVL trees of very small height as special cases (observe that from the definition of the function $k(h)$ it follows that these trees require an enormous area since $k(1) = \frac{25}{6}2^{2\alpha}$).

Formally, the new functions $l(\cdot)$ and $k(\cdot)$ are defined as follows:

$$l(h) = \begin{cases} h, & \text{if } h \leqslant 30, \\ 2^{h/6}, & \text{otherwise,} \end{cases}$$

and

$$k(h+1) = \begin{cases} k(h)\left(1 + \frac{1}{2l(h)}\right), & \text{if } h \geqslant 4, \\ 5.511, & \text{if } h = 3, \\ h, & \text{otherwise.} \end{cases}$$

By reasoning similarly to the previous section, we can show that these two functions satisfy Properties 2 and 4 for any $h \geqslant 4$. Moreover, function $k(\cdot)$ satisfies Property 1: in particular, we have that, for any h,

$$\begin{aligned} \ln k(h) &< \ln\left[k(31)\prod_{i=31}^{\infty}\left(1 + \frac{1}{2 \cdot 2^{i/6}}\right)\right] = \ln k(31) + \sum_{i=31}^{\infty}\ln\left(1 + \frac{1}{2 \cdot 2^{i/6}}\right) \\ &\leqslant \ln k(31) + \frac{1}{2}\sum_{i=31}^{\infty}\frac{1}{2^{i/6}} < \ln k(31) + \frac{1}{2}\frac{1/2^{31/6}}{1 - 1/2^{1/6}} < 2.76 + 0.13 = 2.89. \end{aligned}$$

Then, for any h, $k(h) < \mathrm{e}^{2.89} < 18$.

In a certain sense, Property 3 is also satisfied but in a slightly different way: this is stated by the following result whose proof is postponed to Appendix B.

Proposition 9. *For any $h \geqslant 4$, the following inequalities hold.*

(i) *For any $n = n_1 + n_2 + 1$ with $n_F(h) \leqslant n_1 \leqslant n_2 \leqslant n_c(h)$,*

$$\frac{A(h+1,n) - A(h,n_2)}{\sqrt{A(h+1,n)}} \geqslant l(h).$$

(ii) *For any $n = n_1 + n_2 + 1$ with $n_F(h-1) \leqslant n_1 \leqslant n_2$ and $n_F(h) \leqslant n_2 \leqslant n_c(h)$,*

$$\frac{A(h+1,n) - A(h,n_2)}{\sqrt{A(h+1,n)}} \geqslant l(h-1).$$

(iii) *For any $n = n_1 + n_2 + 1$ with $n_F(h-1) \leqslant n_1 \leqslant n_2$ and $n_F(h) \leqslant n_2 \leqslant n_c(h)$,*

$$\frac{A(h,n_2)}{\sqrt{A(h+1,n)}} \geqslant l(h).$$

By reasoning similarly to the proof of Theorem 4, we can show that, for any AVL tree t of height $h \geqslant 4$ with n nodes and for any rectangle R whose sides have lengths l and L, respectively, satisfying the following two conditions:

$$L \geqslant l \geqslant l(h) \quad \text{and} \quad lL = A(h,n),$$

t admits an h–v drawing within the rectangle R. Indeed, the only significant difference consists of the basis of the inductive proof, which may be established by exhaustively considering all binary tree of height 4.

Theorem 10. *Any AVL tree of height h with n nodes can be upward drawn in any rectangle whose area is equal to* $36n$ *and whose shorter side is at least* $\log n$ *if* $h \leqslant 30$, $n^{1/4}$ *otherwise.*

Proof. Observe that AVL trees of height smaller than 4 clearly satisfy the theorem. The result for AVL trees of height at least 4 follows from the above discussion, from the fact that $k(h) < 18$, and from Proposition 2. □

5. Practical considerations, extensions, and open questions

In this section we shall discuss practical consequences and extensions of the results of the previous sections.

First of all, the theoretical upper bound on the area required to upward draw an AVL tree can be, in practice, substantially improved. The idea is to use Algorithm 1 in order to compute, for each node, the *h–v operation* to be performed at that node. More formally, this can be done by simply replacing in the algorithm the instruction

map the root of t into (x, y)

by the instruction

label the root of t with b

(if a node is labeled with a true value, then its operation is the vertical operation, otherwise its operation is the horizontal one). Once this preprocessing has been realized, the drawing of the tree is obtained by simply performing, for each node, the corresponding operation. We have implemented such a modification. Table 2 presents experimental results obtained for complete binary trees, Fibonacci trees, and combinations of these two kinds of trees, that is, trees of height h whose immediate left subtrees are complete binary trees of height $h-1$ and whose immediate right subtrees are Fibonacci trees of height $h-2$. Note that from these experiments it can be conjectured that the real constant factor in the area bound is approximately 6 (recall that going from h–v drawings to upward drawings causes a doubling of the constant factor). We leave this conjecture as an open problem.

Secondly, observe that the bound on the length of the shorter side allows a very great flexibility to applications that need to draw a binary tree in a prespecified rectangular region: indeed, the only requirement is essentially that the length of the shorter side is a little bit greater than the height of the tree. Moreover, in the case of trees of height smaller than 31 a logarithmic bound on the length of the shorter side is also obtainable. It is still an open question, however, whether this is true for trees of greater height.

Finally, the technique we presented can be easily applied to the class of 2-balanced binary trees. In this case, for any node u of the tree, the number n of nodes in the subtree rooted at u satisfies the following inequality:

$$n_2(h) \leqslant n \leqslant n_c(h),$$

where h denotes the height of the tree rooted at u, and n_2 is defined as

$$n_2(h) = \begin{cases} h, & \text{if } h \leqslant 2, \\ n_2(h-1) + n_2(h-3) + 1, & \text{otherwise.} \end{cases}$$

Table 2
Experimental results on the h–v drawings produced by the modification of Algorithm 1 starting with $l = L = \sqrt{A(h,n)}$ for comparison with the theoretical upper bounds

Tree t	Nodes n	Width of the drawing	Height of the drawing	Area A	Ratio A/n
c_4	15	6	5	30	2.000
c_6	63	12	12	144	2.286
c_8	255	24	28	672	2.635
c_{10}	1023	48	60	2880	2.815
c_{12}	4095	96	119	11424	2.790
c_{14}	16383	192	239	45888	2.801
c_{15}	32767	267	349	93183	2.844
F_6	20	6	5	30	1.5
F_9	88	12	13	156	1.773
F_{11}	232	20	22	440	1.897
F_{13}	609	34	37	1258	2.066
F_{16}	2583	70	74	5180	2.005
F_{17}	4180	94	100	9400	2.249
$c_4 \oplus F_3$	20	6	7	42	2.100
$c_6 \oplus F_5$	76	16	12	192	2.526
$c_8 \oplus F_7$	289	28	29	812	2.810
$c_{10} \oplus F_9$	1112	53	61	3233	2.907
$c_{12} \oplus F_{11}$	4328	104	125	13000	3.004
$c_{14} \oplus F_{13}$	16993	201	253	50853	2.993
$c_{16} \oplus F_{15}$	67132	396	509	201564	3.003

Our proofs can then be modified in order to prove that any 2-balanced binary tree admits a linear-area upward drawing. However, it is easy to see that in this case the constant factor in the area bound increases substantially. Even though one might think of a different algorithm, the behavior of our procedure seems to be sufficiently natural: the more unbalanced the tree, the bigger must be the area of the drawing. This is in accordance with the lower bound of [2] which refers to a family of binary trees which are highly unbalanced. The question still remains open whether other types of balanced trees, such as k-balanced trees with $k > 2$, red–black trees and weight-balanced trees, admit linear-area upward drawings. More generally, it would be interesting to know whether our results can be extended to any family of trees with logarithmic height.

Acknowledgements

We thank Shimon Even for pointing out connections to the work by Shiloach [11]. We are also grateful to the anonymous referees for several useful suggestions.

Appendix A. The proof of Fact 6

For $h \leqslant 7$, the inequality follows from Table 1. Assume then that $h > 7$. From the definition of $\rho(h)$, we have that

$$\begin{aligned}\frac{\rho(h)}{\rho(h+1)} &= \frac{n_F(h-1)+n_c(h)+1}{n_F^2(h-1)}\,\frac{n_F^2(h)}{n_F(h)+n_c(h+1)+1}\\ &> \frac{n_F(h-1)+n_c(h)+1}{n_F^2(h-1)}\,\frac{n_F^2(h)}{2(n_F(h-1)+n_c(h)+1)}\\ &= \frac{1}{2}\,\frac{n_F^2(h)}{n_F^2(h-1)},\end{aligned}$$

where the inequality is due to the fact that, for any h, $n_c(h+1) = 2n_c(h)+1$ and $n_F(h) \leqslant 2n_F(h-1)$ (see Eqs. (1) and (2)).

It is easy to prove by induction on h that $n_F(h) = f_{h+2} - 1$, where f_h denotes the hth Fibonacci number. Moreover, it is well known (see [6, p. 286]) that, for any h,

$$\frac{f_h}{f_{h-1}} = \frac{1+\sqrt{5}}{2} + \frac{1}{f_{h-1}}\left(\frac{1-\sqrt{5}}{2}\right)^{h-1}.$$

It thus follows that

$$\frac{n_F(h)}{n_F(h-1)} = \frac{f_{h+2}-1}{f_{h+1}-1} > \frac{f_{h+2}}{f_{h+1}} = \frac{1+\sqrt{5}}{2} + \frac{1}{f_{h+1}}\left(\frac{1-\sqrt{5}}{2}\right)^{h+1}.$$

In order to prove that $\rho(h)/\rho(h+1) > 1.3$, we then need to show that, for any $h > 7$,

$$\frac{1+\sqrt{5}}{2} + \frac{1}{f_{h+1}}\left(\frac{1-\sqrt{5}}{2}\right)^{h+1} > \sqrt{2.6}.$$

Since $(1+\sqrt{5})/2 > 1.618$ while $\sqrt{2.6} < 1.613$, the above inequality is clearly true whenever h is odd. If h is even, we simply have to observe that

$$\frac{1}{f_{h+1}}\left(\frac{1-\sqrt{5}}{2}\right)^{h+1} > \frac{1}{f_9}\left(\frac{1-\sqrt{5}}{2}\right)^9 = \frac{1}{34}\left(\frac{1-\sqrt{5}}{2}\right)^9 > -0.005.$$

The fact thus follows. □

Appendix B. The proof of Proposition 9

Let us first consider the case $h \geqslant 30$ (observe that in this case $2^{h/6} > h$). First we prove that, for any $n = n_1 + n_2 + 1$ with $n_F(h-1) \leqslant n_1 \leqslant n_2$ and $n_F(h) \leqslant n_2 \leqslant n_c(h)$,

$$\frac{A(h+1,n) - A(h,n_2)}{\sqrt{A(h+1,n)}} \geqslant l(h).$$

Indeed,

$$\begin{aligned}
&\frac{A(h+1,n)-A(h,n_2)}{\sqrt{A(h+1,n)}}\\
&\quad=\frac{k(h+1)n-k(h)n_2}{\sqrt{k(h+1)n}}=\frac{k(h+1)(n_1+n_2+1)-k(h)n_2}{\sqrt{k(h+1)(n_1+n_2+1)}}\\
&\quad\geqslant\frac{k(h+1)n_2-k(h)n_2}{\sqrt{k(h+1)n_2}}=\frac{[k(h+1)-k(h)]n_2}{\sqrt{k(h+1)n_2}}\\
&\quad=\frac{k(h)n_2/(2l(h))}{\sqrt{k(h+1)n_2}}=\frac{\sqrt{k(h)}\sqrt{n_2}/(2l(h))}{\sqrt{1+1/(2l(h))}}\geqslant\frac{\sqrt{k(h)}\sqrt{n_2}/(2l(h))}{\sqrt{1+1/60}},
\end{aligned}$$

where the last inequality is due to the fact that, for any $h\geqslant 30$, $l(h)\geqslant 30$. Since $n_2\geqslant n_F(h)$, then

$$\frac{A(h+1,n)-A(h,n_2)}{\sqrt{A(h+1,n)}}\geqslant\frac{\sqrt{k(h)}\sqrt{n_F(h)}/(2l(h))}{\sqrt{1+1/60}}\geqslant\frac{\sqrt{k(h)}2^{h/3}/(2l(h))}{\sqrt{1+1/60}},$$

where the second inequality is due to the fact[6] that, for any $h\geqslant 3$, $n_F(h)\geqslant 2^{2h/3}$. Since for any h, $l(h)\leqslant 2^{h/6}$, then

$$\frac{A(h+1,n)-A(h,n_2)}{\sqrt{A(h+1,n)}}\geqslant\frac{\sqrt{k(h)}2^{h/6}}{2\sqrt{61/60}}>2^{h/6},$$

where the last inequality is due to the fact that, for any h, $k(h)>5.511$.

Similarly, we can show that, for any $n=n_1+n_2+1$ with $n_F(h-1)\leqslant n_1\leqslant n_2$ and $n_F(h)\leqslant n_2\leqslant n_c(h)$,

$$\frac{A(h,n_2)}{\sqrt{A(h+1,n)}}\geqslant l(h)$$

(in this case we make use of the fact that $n\leqslant 2n_2+1$).

The case $h<30$ will justify our choice of $k(4)$. Let us prove that, for any $n=n_1+n_2+1$ with $n_F(h-1)\leqslant n_1\leqslant n_2$ and $n_F(h)\leqslant n_2\leqslant n_c(h)$,

$$\frac{A(h,n_2)}{\sqrt{A(h+1,n)}}\geqslant l(h).$$

To this aim, it suffices to show that

$$\frac{k(h)n_F(h)}{\sqrt{k(h)(1+1/(2h))(2n_F(h)+1)}}\geqslant h. \tag{B.1}$$

In other words, we have to choose $k(h)$ for $4\leqslant h<30$ so that

$$k(h)\geqslant h^2\frac{(1+1/(2h))(2n_F(h)+1)}{n_F^2(h)}.$$

[6] The proof of this fact easily follows from the well-known identity $n_F(h)=(\frac{1+\sqrt{5}}{2})^{h+2}-(\frac{1-\sqrt{5}}{2})^{h+2}-1$ (see [6, p. 285]).

Table 3
The values of $h^2(1+1/(2h))(2n_F(h)+1)/n_F^2(h)$ for $h=4,\dots,13$

4	5	6	7	8	9	10	11	12	13
5.51	4.774	3.998	3.23	2.542	1.954	1.474	1.093	0.799	0.577

Table 4
The lower bounds for $k(h)$ for $h=4,\dots,13$

4	5	6	7	8	9	10	11	12	13
3.501	3.997	4.179	4.083	3.811	3.428	2.993	2.545	2.114	1.717
3.6	4.847	5.507	5.727	5.551	5.117	4.523	3.86	3.193	2.567

Table 3 shows the values of the right-hand side for $h=4,\dots,13$: the values for $h>15$ are all smaller than these values. By choosing $k(4)=5.511$ it follows that the inequality (B.1) is satisfied (recall that $k(h)$ is an increasing function).

Similarly, we can deal with the first two conditions of the proposition. In particular, it is not difficult to prove that we have to choose $k(h)$ so that it is bigger than the values shown in the Table 4: by taking $k(4)=5.511$ this is clearly true. □

References

[1] G.M. Adelson-Velskii, E.M. Landis, An algorithm for the organization of information, Soviet Math. Dokl. 3 (1962) 1259–1262.
[2] P. Crescenzi, G. Di Battista, A. Piperno, A note on optimal area algorithms for upward drawings of binary trees, Computational Geometry: Theory and Applications 2 (4) (1992) 187–200.
[3] G. Di Battista, P. Eades, R. Tamassia, I.G. Tollis, Algorithms for drawing graphs: an annotated bibliography, Computational Geometry: Theory and Applications 4 (5) (1994) 235–282.
[4] P. Eades, T. Lin, X. Lin, Minimum size h–v drawings, in: Proceedings International Workshop AVI '92, 1992, pp. 386–394.
[5] A. Garg, M.T. Goodrich, R. Tamassia, Area-efficient upward tree drawing, in: Proceedings ACM Symposium on Computational Geometry, 1993, pp. 359–368.
[6] R.L. Graham, D.E. Knuth, O. Patashnik, Concrete Mathematics, Addison-Wesley, Reading, MA, 1989.
[7] Y. Horibe, An entropy view of Fibonacci trees, Fibonacci Quarterly 20 (1982) 168–178.
[8] Y. Horibe, Notes on Fibonacci trees and their optimality, Fibonacci Quarterly 21 (1983) 118–128.
[9] D.E. Knuth, The Art of Computer Programming: Sorting and Searching, Addison-Wesley, Reading, MA, 1975.
[10] E. Reingold, J. Tilford, Tidier drawing of trees, IEEE Trans. Software Engrg. 7 (1981) 223–228.
[11] Y. Shiloach, Linear and planar arrangements of graphs, Ph.D. Thesis, Department of Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel, 1976.
[12] K.J. Supowit, E. Reingold, The complexity of drawing trees nicely, Acta Informatica 18 (1983) 377–392.
[13] L. Valiant, Universality considerations in VLSI circuits, IEEE Trans. Comput. 30 (2) (1981) 135–140.