Order 5 (1988), 33–43.

On the Width of an Orientation of a Tree*

M. D. ATKINSON and D. T. H. NG
School of Computer Science, Carleton University, Ottawa, Canada K1S 5B6

Communicated by I. Rival

(Received: 15 September 1987; accepted: 9 December 1987)

Abstract. There are 2^{n-1} ways in which a tree on n vertices can be oriented. Each of these can be regarded as the (Hasse) diagram of a partially ordered set. The maximal and minimal widths of these posets are determined. The maximal width depends on the bipartition of the tree as a bipartite graph and it can be determined in time $O(n)$. The minimal width is one of $\lfloor \lambda/2 \rfloor$ or $\lfloor \lambda/2 \rfloor + 1$, where λ is the number of leaves of the tree. An algorithm of execution time $O(n + \lambda^2 \log \lambda)$ to construct the minimal width orientation is given.

AMS subject classifications (1980). 06A10, 68C05.

Key words. Diagram, tree, orientation, width, algorithm.

1. Introduction

Let T be any (unrooted) tree on n vertices. Each of the $n-1$ edges of T can be oriented in one of two directions and, hence, associated with T is a set of 2^{n-1} directed graphs. Each of these orientations of T can be regarded as the (Hasse) diagram of a partially ordered set. It is interesting to consider the variation over this set of 2^{n-1} related posets of some order theoretic parameter. It was shown in [4] that any poset whose diagram is a tree has dimension at most 3 and so the dimension parameter has very little variation. Other parameters which have been considered are number of comparable pairs, and number of linear extensions [2, 3]. In this paper we consider the width. The width of a poset is the size of a largest anti-chain or, equivalently, the minimal number of chains needed to cover the elements. The orientations of greatest width are easily found but the least width orientations are less trivial. It is natural to relate the width of an orientation to the number of leaves λ of the tree. We shall prove that the smallest possible width is either $\lfloor \lambda/2 \rfloor$ or $\lfloor \lambda/2 \rfloor + 1$. Our methods are algorithmic and allow us to compute an orientation of smallest width in time $O(n + \lambda^2 \log \lambda)$. In all discussions about algorithm execution times we shall use the RAM model of computation with arbitrary precision integers (although the integers used are of maximum value $O(n + \lambda)$).

* This research was partially funded by the National Science and Engineering Research Council of Canada under Grant Number A4219.

2. Orientations of Greatest Width

We begin by giving a simple result about orientations with the greatest width. Every tree T is a bipartite graph. Let A and B be the two sets in the bipartition. There are two canonical orientations of T; in one of them every element of A is maximal and every element of B is minimal, while the other is the dual of this. We shall call these orientations the *win/lose* orientations of T.

THEOREM A. *A win/lose orientation of a tree T maximises the width of the associated poset. The width of this orientation can be determined in time $O(n)$.*

Proof. In a win/lose orientation there are no transitive consequences of the covering relations described by the tree edges. Therefore such an orientation has the largest possible anti-chains. We now consider how to find the width of a win/lose orientation. The first step is to construct a maximal matching of T. Let x be any leaf vertex and let y be the neighbour of x. In a maximal matching exactly one of the edges incident with y is a matching edge, and it is easy to see that the matching may be adjusted so that this edge is (x, y). If the other edges incident with y are deleted the resulting connected components inherit matchings which are also maximal. We can now treat these components in the same way. Thus, a maximal matching can be constructed in time $O(n)$. Once a maximal matching is found a minimal edge cover of the tree is obtained by taking the edges of the maximal matching together with one edge for each vertex not yet covered. For the win/lose orientation an edge cover is the same as a chain cover and so the number of edges in the edge cover is the width.

3. Orientations of Least Width

According to Theorem A, if a tree is oriented to have minimal height, then it has maximal width. It might therefore be supposed that an orientation of minimal width could be found among the orientations of maximal height. This is false, as the tree in Figure 1 demonstrates: each of its orientations of maximal height have width 4 or more, while there is an orientation of width 3.

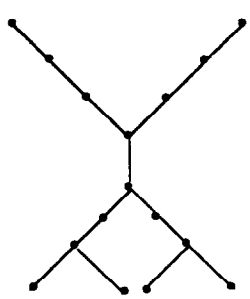


Fig. 1.

For brevity we shall call the width of a minimum width orientation of a tree T the *minimum width* of the tree, and denote it by $w(T)$. If a tree has λ leaves then $w(T) \geq \lambda/2$ since, in every orientation, each leaf is either a maximal or minimal element and so among the leaves there is always an antichain of size at least $\lambda/2$. This lower bound on $w(T)$ cannot necessarily be attained, as is shown by the tree in Figure 2.

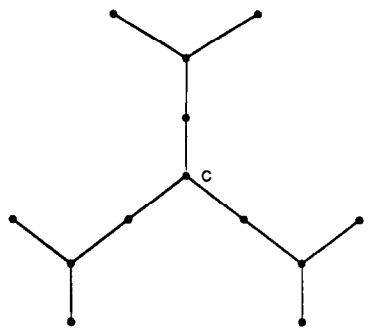


Fig. 2.

Consider any orientation of this tree. Two of the tree edges incident on c must be directed the same way (into c, or out of c). Then the two corresponding subtrees on 4 nodes cannot have comparabilities between them. Each of these has width at least 2, so the total width is at least 4.

A tree is said to be *reduced* if it has the following properties:

1. No two nodes of degree 2 are adjacent,
2. No node of degree 2 is adjacent to a leaf.

Any tree may be converted to a reduced tree by replacing each path of nodes of degree 2 by a single node of degree 2, and by removing each node of degree 2 that is adjacent to a leaf. In proving results about minimum width orientations of a tree, it is sufficient to consider reduced trees in view of Lemma 1.

LEMMA 1. *Let T be any tree on n vertices and let T^* be its reduced form. Then the minimum widths of T and T^* are equal. Moreover, from a minimum width orientation of one of these trees, one can obtain a minimal width orientation of the other in $O(n)$ operations.*

Proof. Let w, w^* be the minimal widths of T, T^*. Suppose that T is oriented to have width w. As T is converted to its reduced form (by removing nodes of degree 2) the width cannot increase. Hence, the orientation of T^* which is ultimately obtained has width no more than w; thus $w^* \leq w$. On the other hand, suppose we have any minimal width orientation of T^*. We can take a chain cover of its vertices with w^* chains with the property that at least one chain flows along each edge incident with a node of degree 1 or 2. From this orientation we can obtain an orientation of T and a cover of the vertices

of T with w^* chains; because T can be recovered from T^* by subdividing edges already covered by chains (and inserting nodes into the chains of the vertex cover). Hence, $w = w^*$. It is clear that only $O(n)$ operations are involved in these processes.

A *pointed* tree is a rooted tree whose root is a leaf. The single edge incident with the root is called the *point* of the tree. If T is any pointed tree we may consider the edges which meet the point to be the points of pointed subtrees of T; the phrase 'the pointed subtrees of T' refers to these pointed trees. The pointed subtrees of T in turn have pointed subtrees, and so on. In this way every pointed tree has a hierarchical structure similar to the hierarchical structure of a rooted tree.

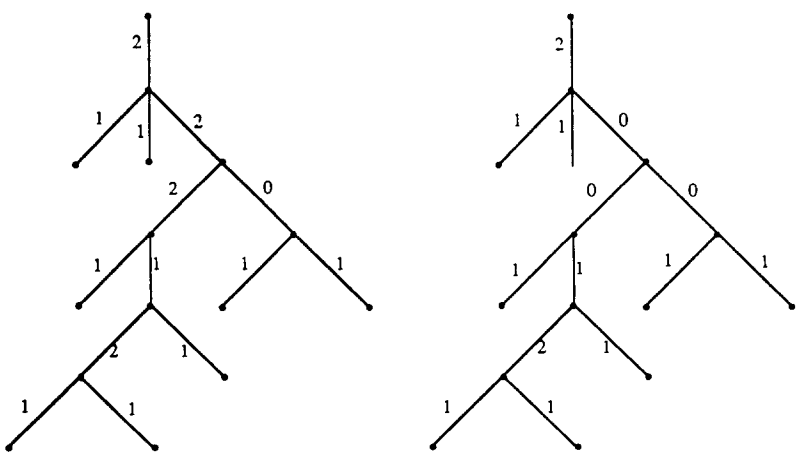
A *perfect labelling* of a reduced pointed tree is a labelling of the edges with nonnegative integers such that

(1) all edges incident with a leaf, with the possible exception of the point, have the label 1,
(2) at every internal vertex not all incident edges have label 0,
(3) at every interval vertex v there is a *flow* condition: namely, a relation $\sum_i s_i a_i = 0$, where the summation is over all edges incident with v, the labels on these edges are $a_1, a_2, \ldots$, and each $s_i = \pm 1$.

A *quasi-perfect labelling* of a reduced pointed tree is a labelling of the edges with nonnegative integers such that the three conditions for a perfect labelling apply except for precisely one of the following exceptions:

(1*) exactly one of the edges incident with a leaf, but not the point, has label 0 rather than 1,
(2*) at exactly one of the internal vertices all incident edges have label 0.

For example, see Figure 3.



Perfect labelling Fig. 3. Quasi-perfect labelling

The following result follows directly from the definitions of perfect and quasi-perfect labelling.

LEMMA 2. *A labelling of the reduced pointed tree on two vertices (the unique minimal reduced pointed tree) is perfect if the label on the single edge is* 1 *and quasi-perfect if the label is* 0. *If* T *is a reduced pointed tree on more than* 2 *vertices and if* $T_1, T_2, \ldots, T_r$ *are its pointed subtrees then*

(1) *A labelling of* T *is perfect if and only if the induced labellings of* $T_1, T_2, \ldots, T_r$ *are also perfect, and the labels on the points of* $T, T_1, T_2, \ldots, T_r$ *satisfy a flow condition and are not all zero.*

(2) *A labelling of* T *is quasi-perfect if and only if either*

(i) *the induced labellings of* $T_1, T_2, \ldots, T_r$ *are all perfect, and the labels on the points of* $T, T_1, T_2, \ldots, T_r$ *are all zero, or*

(ii) *the induced labellings of* $T_1, T_2, \ldots, T_r$ *are all perfect except for a single tree* T_j *which has a quasi-perfect labelling, and the labels on the points of* $T, T_1, T_2, \ldots, T_r$ *satisfy a flow condition and are not all zero.*

If any T *is any reduced pointed tree then*

$$A(T) = \{x \mid \text{in some perfect labelling of } T \text{ the point is labelled with } x\},$$

and

$$B(T) = \{x \mid \text{in some quasi-perfect labelling of } T \text{ the point is labelled with } x\}.$$

The following lemma is a direct consequence of these definitions and the previous lemma.

LEMMA 3. *Let* T *be any reduced pointed tree with pointed subtrees* $T_1, T_2, \ldots, T_r$. *Then*

(1) $A(T) = \{x \mid x \geqslant 0, x = \sum \pm a_i, a_i \in A(T_i), i = 1, 2, \ldots, r, \text{ not all } a_i = 0\}$,

(2) $B(T) = \{x \mid x \geqslant 0, \ x = \sum \pm a_i, \ a_i \in A(T_i), \ i = 1, 2, \ldots, j-1, \ j+1, \ldots, r, a_j \in B(T_j), \text{ not all } a_i = 0\} \cup Z$, *where* $Z = \{0\}$ *if* $0 \in \bigcap A(T_i)$ *and* $Z = \varnothing$ *otherwise.*

COROLLARY. *If* $r = 1$, *then*

(1) $A(T) = A(T_1) - \{0\}$,

(2) $B(T) = (B(T_1) - \{0\}) \cup Z$.

LEMMA 4. *If* T *is a reduced pointed tree with* $|A(T)| = 1$, *then* $0 \in B(T)$.

Proof. Note first that the lemma is true for a tree on two vertices and so from now on we can suppose that T has pointed subtrees $T_1, T_2, \ldots, T_r, r \geqslant 1$. Note also that $A(T) = \{0\}$ is impossible.

We continue with the notation of the last lemma. By our last remark, each of the $A(T_i)$ contains some nonzero element and, therefore, $r \geqslant 2$ implies $|A(T)| \geqslant 2$;

hence, T has only one pointed subtree T_1. The pointed subtree T_1 must have $|A(T_1)| > 1$ otherwise it too has just one subtree and T would not be reduced. However, the only element it can contain except for the single element of $A(T)$ itself is 0. Now the lemma follows from the last Corollary.

Let T be any tree with an orientation. A *free chain* in the oriented tree is a totally ordered path of at least two vertices beginning and ending at a leaf. The connection between tree labellings, orientations and width is furnished by the following result.

LEMMA 5. *Let T be any reduced pointed tree. If T has a perfect labelling with point labelled x then T has an orientation in which there is a set of free chains which covers all the nonroot vertices,covers each nonroot-leaf exactly once, and covers the root x times. If T has a quasi-perfect labelling with point labelled x, then T has an orientation in which there is a set of free chains which covers all but one of the nonroot vertices, covers each nonroot-leaf at most once, and covers the root x times.*

Proof. We use induction on the number of vertices in the tree to prove both statements simultaneously. If the tree has two vertices only, the result is obvious. We shall therefore assume that T has more than two vertices, and that the result is true for the pointed subtrees $T_1, T_2, \ldots, T_r$ *of* T. Let Q be the common root of all these subtrees.

If the labelling of T is perfect, then the induced labellings of each of $T_1, T_2, \ldots, T_r$ are perfect, and not all labels $a_1, a_2, \ldots, a_r$ on the points of $T_1, T_2, \ldots, T_r$ are zero. By the inductive hypothesis $T_1, T_2, \ldots, T_r$ have orientations and chain covers satisfying the first part of the lemma. The label x on the point of T satisfies $x = \sum \pm a_i$ and we may write $x = b - c$, where b is the sum of those a_i which have coefficient +1, and c is the sum of those a_i which have coefficient −1. By reversing the orientations of some of the T_i if necessary we can assume that b chains flow into Q and c chains flow out. We orient the point of T so that the direction is from Q to the root of T. Then we can obtain a chain cover of the entire set of nonroot vertices of T as follows. The c chains flowing out of Q are appended to c of the chains flowing into Q and the other $b - c = x$ chains flowing into Q are continued along the point of T (thereby covering the root of T x times). Notice that the vertex Q itself is covered since at least one of $a_1, a_2, \ldots, a_r$ is nonzero.

If the labelling of T is quasi-perfect there are two situations to consider:

(i) the induced labellings of $T_1, T_2, \ldots, T_r$ are perfect but their labels $a_1, a_2, \ldots, a_r$ are all zero; in this case we argue exactly as above, the only difference being that Q is now not covered by the chains,

(ii) with one exception the induced labellings of $T_1, T_2, \ldots, T_r$ are perfect, the exceptional subtree has a quasi-perfect labelling, and not all labels a_1,

$a_2, \ldots, a_r$ are all zero; here the argument is again similar but the exceptional subtree will contain one vertex not covered by the chains.

LEMMA 6. *If T is any reduced pointed tree and if $1 \leqslant x \leqslant \mathrm{Max}(A(T))$, then $x \in A(T) \cup B(T)$.*

Proof. We argue by induction on the number of nodes of T. The result is true for the unique minimal reduced pointed tree on two nodes so we shall assume that T has more than two nodes and that the result is true for each of the pointed subtrees $T_1, T_2, \ldots, T_r$ of T.

Let $1 \leqslant x \leqslant \max(A(T))$ and, in order to reach a contradiction, assume that $x \notin A(T) \cup B(T)$.

Let $m_i = \max(A(T_i))$, $i = 1, 2, \ldots, r$.

If $r = 1$, then we may appeal to the corollary to Lemma 3: $1 \leqslant x \leqslant \max(A(T_1))$, so $x \in A(T_1) \cup B(T_1)$ and therefore $x \in A(T) \cup B(T)$. Thus we may take $r \geqslant 2$.

Choose y such that $y > x$, y has a representation in the form $y = \sum \varepsilon_i m_i$, $\varepsilon_i = \pm 1$, and subject to this y is minimal. At least one of the signs ε_i is positive, so assume $\varepsilon_1 = +1$. Then

$$m_1 + \sum_{i \neq 1} \varepsilon_i m_i = y > x > -m_1 + \sum_{i \neq 1} \varepsilon_i m_i$$

(the last inequality following because y was minimal). The set of positive integers among $\{\pm z + \sum_{i \neq 1} \varepsilon_i m_i \mid 1 \leqslant z \leqslant m_1\}$ are all members of $A(T) \cup B(T)$ (by Lemma 3). Since x is not a member of this set, we have $x = \sum_{i \neq 1} \varepsilon_i m_i$. In particular there must be another positive sign ε_2, say. Then, repeating the same argument we have $x = \sum_{i \neq 2} \varepsilon_i m_i$; thus $m_1 = m_2$.

If $|A(T_1)| = 1$ then, by Lemma 4, $0 \in B(T_1)$, and, using Lemma 3, $x = 0 + \sum_{i \neq 1} \varepsilon_i m_i$ belongs to $B(T)$, a contradiction. On the other hand, if $A(T_1)$ contains a member z in addition to m_1, we have the expression for x, $x = z + (m_1 - z) + \sum_{i \geqslant 3} \varepsilon_i m_i$, and, as $1 \leqslant m_1 - z \leqslant m_2$, $m_1 - z \in A(T_2) \cup B(T_2)$, by induction; but then, by Lemma 3, $x \in A(T) \cup B(T)$, a final contradiction.
From this lemma the following result is immediate.

COROLLARY. *Every reduced pointed tree has a perfect or quasi-perfect labelling in which the point is labelled with* 1.

THEOREM B. *Let T be any tree. Then one of the following conditions holds.*

(1) *T has an orientation and a vertex cover by free chains $C_1, C_2, \ldots, C_r$ such that every leaf is the source or sink of exactly one chain,*
(2) *T has an orientation and a vertex cover by free chains $C_1, C_2, \ldots, C_r$, and a chain D, such that every leaf is the source or sink of at most one C_i, and the set of vertices not covered by $\bigcup C_i$ is equal to D.*

Proof. We shall first prove that, for a reduced tree T^*, the theorem holds with D consisting of one node only. Let T^* be reduced and by choice of any

leaf consider T^* to be a reduced pointed tree. By the last corollary we know that T^* has a perfect or quasi-perfect labelling with point labelled 1. Now, from Lemma 5, it follows that T^* has a cover by chains with the properties claimed.

The theorem now follows for an arbitrary tree T by applying the result just proved to the reduced version T^* of T.

THEOREM C. *Every tree with λ leaves has an orientation of width $\lfloor \lambda/2 \rfloor$ or $\lfloor \lambda/2 \rfloor + 1$.*

Proof. The required orientation is given by the previous theorem. In case 1, $\lambda = 2r$ and r chains suffice to cover all vertices while in case 2, $\lambda = 2r$ or $\lambda = 2r+1$ and $r+1$ chains suffice.

4. Complexity Analysis

In this section we show how the existence result, Theorem C, can be made constructive by giving an efficient algorithm for constructing the minimal width orientation of a tree. If $A_1, A_2, \ldots, A_r$ are sets of integers then we denote the set $\{\sum s_i a_i \mid s_i = \pm 1 \text{ and } a_i \in A_i\}$ by $\sum \pm A_i$ or by $\pm A_1 \pm A_2 \pm \cdots \pm A_r$.

LEMMA 7. *Let $A_1, A_2, \ldots, A_r$ be sets of integers and suppose that every element of $\sum \pm A_i$ is of absolute value at most λ. Then the set $\sum \pm A_i$ can be computed using $O(r\lambda \log \lambda)$ operations.*

Proof. Consider the expression

$$x^{-\lambda} \prod_{i=1}^{r} \sum_{a_i \in A_i} (x^{a_i+\lambda} + x^{-a_i+\lambda}) = \sum_{i=-\infty}^{\infty} m_i x^i.$$

Clearly, $m_k \neq 0$ precisely when $k \in \sum \pm A_i$. By performing each of the $r-1$ polynomial multiplications on the left-hand side of this equation using the fast algorithm based on the finite Fourier transform [1], we obtain the operation count $O(r\lambda \log \lambda)$ since each polynomial has degree at most 2λ.

In the main application of Lemma 7 it is sometimes necessary to recognise when $0 \in \sum \pm A_i$ has a single representation as $0 + 0 + \cdots + 0$, with each $0 \in A_i$. Since m_0 is the number of representations of 0 as $\sum s_i a_i$, this is straightforward.

LEMMA 8. *Let $A_1, A_2, \ldots, A_r$ be sets of integers and suppose that every element of $\sum \pm A_i$ is of absolute value at most λ. If $x \in \sum \pm A_i$, then a representation of x as $x = \sum s_i a_i$, where $s_i = \pm 1$ and $a_i \in A_i$ can be computed in $O(r\lambda \log \lambda)$ operations.*

Proof. By the method of the previous lemma, we compute each of the sets

$$S_1 = \pm A_1, \qquad S_i = S_{i-1} \pm A_i, \quad i = 2, 3, \ldots, r.$$

This requires $O(r\lambda \log \lambda)$ operations. We also sort each set A_i; in all, this requires a further $O(r\lambda \log \lambda)$ operations.

Now put $x_r = x$ and express x_r as

$$x_r = \pm x_{r-1} \pm a_r \quad \text{with } x_{r-1} \in S_{r-1} \quad \text{and} \quad a_r \in A_r.$$

This is done by considering each element y of S_{r-1} in turn and, by binary search, determining whether $\pm x_r \pm y \in A_r$. Thus, the decomposition of x_r takes $O(\lambda \log \lambda)$ steps to calculate. A similar computation is now iterated to decompose x_i, $i = r-1, \ldots, 2$ as $x_i = \pm x_{i-1} \pm a_i$, with $x_{i-1} \in S_{i-1}$, $a_i \in A_i$ until the full decomposition of x is found.

A small variant of Lemma 8 is sometimes needed. Suppose that $x = 0$ and that it is known that x has a representation as $x = \sum s_i a_i$ with not all $a_i = 0$. Such a representation may be found by considering the elements $y \in S_{r-1}$ (and S_{r-2}, $S_{r-3}, \ldots$) in decreasing order of absolute value. This can be accomplished by sorting each S_i; the extra cost is only $O(r\lambda \log \lambda)$ operations.

LEMMA 9. *Let $A_1, A_2, \ldots, A_r$ be sets of integers and suppose that every element of $\sum \pm A_i$ is of absolute value at most λ. Let B_i, $i = 1, 2, \ldots, r$ be sets of integers of absolute value at most the maximum absolute value occurring in A_i. Suppose that an integer x is known to lie in one of the following sets*

$$C_j = \pm A_1 \pm A_2 \pm \cdots \pm A_{j-1} \pm B_j \pm A_{j+1} \pm \cdots \pm A_r, \quad j = 1, 2, \ldots, r.$$

Then, in time $O(r\lambda \log \lambda)$, an index j can be found for which $x \in C_j$, and the representation of x as

$$x = \pm a_1 \pm a_2 \pm \cdots \pm a_{j-1} \pm b_j \pm a_{j+1} \pm \cdots \pm a_r$$

can also be determined.

Proof. As in the previous lemma we form the sets

$$S_1 = \pm A_1, \qquad S_i = S_{i-1} \pm A_i, \quad i = 2, 3, \ldots, r.$$

and

$$U_r = \pm A_r, \qquad U_i = U_{i+1} \pm A_i, \quad i = r-1, r-2, \ldots, 1.$$

The time needed for this is $O(r\lambda \log \lambda)$.

Next, for each j, $1 \leqslant j \leqslant r$, we form $C_j = \pm S_{j-1} \pm B_j \pm U_{j+1}$ until a value j is found for which $x \in C_j$. This requires $O(r\lambda \log \lambda)$ operations.

Finally, by the method of Lemma 8, we find the required representation of x.

LEMMA 10. *Let T be a reduced tree with n vertices and λ leaves. Then $n \leqslant 3\lambda - 5$.*

Proof. Let d_i be the number of vertices of degree i in the tree. Then

$$2(n-1) = d_1 + 2d_2 + 3d_3 + 4d_4 + \cdots \geqslant d_1 + 2d_2 + 3(d_3 + d_4 + \cdots) = d_1 + 2d_2 + 3(n - d_1 - d_2)$$

from which it follows that

$$n \leqslant 2d_1 + d_2 - 2.$$

The fact that T is reduced gives (by counting, in two ways, edges with one end of degree at most 2 and the other of degree at least 3)

$$d_1 + 2d_2 \leqslant 3d_3 + 4d_4 + \cdots = 2(n-1) - 2d_2 - d_1$$

or

$$d_2 \leqslant \frac{n-1}{2} - \frac{d_1}{2}.$$

Now it follows that

$$n \leqslant 2d_1 + \frac{n-1}{2} - \frac{d_1}{2} - 2,$$

giving the result.

THEOREM D. *Let T be any tree with n vertices and λ leaves. Then a minimal width orientation of T can be found in $O(n + \lambda^2 \log \lambda)$ operations.*

Proof. We shall give an algorithm for reduced trees of execution cost $O(\lambda^2 \log \lambda)$. The result for arbitrary trees then follows from Lemma 2.

Let T be any reduced tree. Choose any leaf and regard T as a pointed tree. The first step is to compute the set $A(T)$. This is done by repeated use of Lemma 3 working up the tree from leaves to root using the method of Lemma 7. The total cost of all these computations is $\sum r_v \lambda \log \lambda$, the sum being over all internal vertices v, where r_v is the number of pointed subtrees rooted at v. Since $\sum r_v = O(n) = O(\lambda)$ (by Lemma 10) this cost is $O(\lambda^2 \log \lambda)$.

In the second step we construct a perfect or quasi-perfect labelling of the tree and an orientation satisfying Lemma 5 in which the point is labelled with 1. If $1 \in A(T)$, the labelling will be perfect, otherwise it will be quasi-perfect. Initially the point is labelled with 1. The remainder of the labelling is carried out by working down the tree. A typical step is to use the label x and the direction on the point of some pointed subtree S in order to define labels $a_1, a_2, \ldots, a_r$ and directions on the points of the pointed subtrees $S_1, S_2, \ldots, S_r$ of S. There are three cases:

(1) If $x \in A(S)$, then $x = \sum_i s_i a_i$, with $a_i \in A(S_i)$, not all zero, and $s_i = \pm 1$; then the a_i are the required labels on the points $S_1, S_2, \ldots, S_r$ and the s_i give the directions on these points (ensuring that the total flow into the common root of $S_1, S_2, \ldots, S_r$ equals the total outward flow);

(2) If $x \in B(S)$, then $x = \sum_i s_i a_i$, with all $a_i \in A(S_i)$, except for one value $a_j \in B(S_j)$, and $s_i = \pm 1$; again the a_i are the required labels on the points S_1, $S_2, \ldots, S_r$ and the s_i give the directions on these points.

(3) If $x \in B(S)$, $x = 0$, and $0 = a_i \in A(S_i)$ for all i; then the a_i are the required

labels on the points $S_1, S_2, \ldots, S_r$ and the directions can be taken arbitrarily.

The first case is handled by Lemma 8, the second by Lemma 9, the third requires a simple check only, and the total costs involved amount to $\sum r_v \lambda \log \lambda = O(\lambda^2 \log \lambda)$.

This concludes the proof of Theorem D. As a footnote we observe that the problem of simply determining the minimal width (one of $\lambda/2$ and $\lfloor \lambda/2 \rfloor + 1$) can be answered in $(n + \lambda^2)$ steps. The technique again uses fast polynomial multiplication with the Fourier transform but the processing is done almost entirely in transform space.

Acknowledgements

We thank Professor W. H. Cunningham for information about matchings in trees, and the referees for some helpful suggestions.

References

1. A. V. Aho, J. E. Hopcroft, and J. D. Ullman (1974) *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass.
2. M. D. Atkinson (1988) The complexity of order, *Proc. NATO Advanced Study Institute on Algorithms and Order* (ed. I. Rival), Kluwer Academic Publishers, Dordrecht.
3. M. D. Atkinson (1988) Partially ordered sets and sorting, *Proc. IMA Conference on Computers in Mathematics*, Oxford University Press, Oxford.
4. W. T. Trotter and J. I. Moore (1977) The dimension of planar posets, *J. Comb. Theory (b)* **22**, 54–67.