INFORMATION TO USERS

This reproduction was made from a copy of a document sent to us for microfilming. While the most advanced technology has been used to photograph and reproduce this document, the quality of the reproduction is heavily dependent upon the quality of the material submitted.

The following explanation of techniques is provided to help clarify markings or notations which may appear on this reproduction.

1. The sign or "target" for pages apparently lacking from the document photographed is "Missing Page(s)". If it was possible to obtain the missing page(s) or section, they are spliced into the film along with adjacent pages. This may have necessitated cutting through an image and duplicating adjacent pages to assure complete continuity.

2. When an image on the film is obliterated with a round black mark, it is an indication of either blurred copy because of movement during exposure, duplicate copy, or copyrighted materials that should not have been filmed. For blurred pages, a good image of the page can be found in the adjacent frame. If copyrighted materials were deleted, a target note will appear listing the pages in the adjacent frame.

3. When a map, drawing or chart, etc., is part of the material being photographed, a definite method of "sectioning" the material has been followed. It is customary to begin filming at the upper left hand corner of a large sheet and to continue from left to right in equal sections with small overlaps. If necessary, sectioning is continued again–beginning below the first row and continuing on until complete.

4. For illustrations that cannot be satisfactorily reproduced by xerographic means, photographic prints can be purchased at additional cost and inserted into your xerographic copy. These prints are available upon request from the Dissertations Customer Services Department.

5. Some pages in any document may have indistinct print. In all cases the best available copy has been filmed.

University Microfilms International
300 N. Zeeb Road
Ann Arbor, MI 48106

Jarnagin, Richard March

STUDIES OF BALANCED TREE STRUCTURES

City University of New York

PH.D. 1984

PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark ✓.

1. Glossy photographs or pages _____
2. Colored illustrations, paper or print _____
3. Photographs with dark background _____
4. Illustrations are poor copy _____
5. Pages with black marks, not original copy _____
6. Print shows through as there is text on both sides of page _____
7. Indistinct, broken or small print on several pages ✓
8. Print exceeds margin requirements _____
9. Tightly bound copy with print lost in spine _____
10. Computer printout pages with indistinct print _____
11. Page(s) _____ lacking when material received, and not available from school or author.
12. Page(s) _____ seem to be missing in numbering only as text follows.
13. Two pages numbered _____. Text follows.
14. Curling and wrinkled pages _____
15. Other _____

University
Microfilms
International

STUDIES OF BALANCED TREE STRUCTURES

BY

RICHARD JARNAGIN

A dissertaion submitted to the Graduate Faculty in Mathematics in partial fulfillment of the requirements for the degree of Doctor of Philososphy, The City University of New York.

This manuscript has been read and accepted by the Graduate Faculty in Mathematics in satisfaction of the dissertation requirement for the degree of Doctor of Philosophy.

12/23/83
date

Professor Michael Anshel
Chairman of Examining Committee

12/23/83

Professor Martin Moskowitz
Executive Officer

Professor Alphonse Vasquez

Professor Stanley Kaplan

Supervisory Committee
City University Graduate Center
City Univerity of New York

ACKNOWLEDGMENTS

I owe a great debt of gratitude to my advisor, Professor Michael Anshel, for his mathematical guidence and personal concern for me. I would also like to thank the members of my dissertation committee for their thoughtful comments on drafts of the thesis. Finally, I would like to thank Doctor William Gewirtz and James Toth for their concern, support, and suggestions.

TABLE OF CONTENTS

OVERVIEW

In the twenty years since the discovery of AVL or height-balanced trees a large literature concerning their combinatorial structure has arisen. For example, Knuth [13] has made extensive empirical studies on insertion algorithms on these trees. Alternative structures such as B-trees, k-balanced trees, and weight-balanced trees have been introduced and studied. In a modern development [22] related to AVL trees, rebalancing transformations of B-trees under insertion have been related to Markov processes. Brown [7] subsequently applied the Markov concepts to a closed set of subtrees of height-balanced trees known as M and N subtrees and established bounds for the probability that an insertion into an arbitrary AVL tree with n nodes would imbalance the tree. Brown's bounds were improved by Melhorn [15] who in addition generalized the model for insertions to include deletions of nodes as well. In another direction, the worst-case behavior of insertion and deletion algorithms has been studied in [13] as well as [16].

One of the major directions of this thesis is to examine all the combinatorial ramifications of the Markov connection for AVL trees. The insight travels in both directions. On the one hand Markov analysis allows

us to estimate the density of AVL trees with M M-subtrees and N N-subtrees. On the other, the study of m-way search trees leads us to the concept of Markov processes with random rules and an an algebraic criterion for the stability of a Markov process.

For an introduction to AVL trees the reader is referred to Chapter 1. What follows here is a brief discussion of the fringe analysis of M and N subtrees to motivate some of the results announced below.

If a leaf has another node opposite it in an AVL tree, we refer to it as an N-subtree. If a leaf has a failure node opposite it in the tree, the leaf and its parent are referred to as an M-subtree. In Chapter 3 of this thesis we prove a number of results concerning these subtrees. For example, let F(n,N) enumerate the number of height-balanced trees (generated with multiplicity by insertion) with N N-subtrees. In THEOREM 3.II.1 it is shown that F(n,N) is unimodal for all n. In THEOREM 3.II.1 we relate the $\{M,N\}$, or fringe, structure of an AVL tree to the history of its generation by insertions. Rotations (see Chapter 1 for a definition) can be characterized by their height, or the length of the path from the newly inserted node to the node around which the rotation is made. THEOREM 3.I.2 is used to predict the worst case, as well as the average, number of rotations of height 2 which occur in the generation of a height-balanced tree with n nodes, N N-subtrees, and M M-subtrees. In THEOREM 3.III.1 we generalize this result and apply the methods back to a class of urn models for aftereffect well-known in probability theory. Again, curiously, the probability model for m-way search trees leads us to the concept of a Markov process with random rules.

In Chapter 4 we consider the worst-case performance of insertion

algorithms on AVL trees. For example, we show in THEOREM 4.I.1 that the exact number of rotations which occur in the generation of an AVL tree whose insertion sequence corresponds to the identity permutation is $n - \log_2 n - 1$. In addition, we enumerate subsequences of this sequence at which rotations of height k, $k < \log_2 n$, occur. For each node, we may associate a number, the balance factor, which is the difference in height of the left and right subtrees of the node. When an insertion is made into an AVL tree balance factor adjustments must be made to reflect the new structure of the tree. As a byproduct of this last result, we show that the total number of balance factor adjustments required to insert the identity permutation sequence into an AVL tree is O(n) where the constant can be explictly computed. Next we borrow a well known method used to prove the linearity of the time complexity of the subroutine HEAPIFY in the algorithm HEAPSORT. In THEOREM 4.II.2 we apply a variation of the proof to show that the number of balance factor adjustments for AVL trees generated by a large class of permutations is linear in n, the number of nodes in the tree.

Let F(k) enumerate the number of distinct AVL trees of height k. Then, as will be shown, $F(k) = F(k-1)^2 + 2*F(k-1)*F(k-2)$. In Chapter 5 we repeatedly exploit this relation to prove probabilistic results about rotations on AVL trees of height k. A failure node of an AVL tree will be deemed rotational if an insertion at that failure node will imbalance the tree. In THEOREM 5.I.2 we establish the following result: for any j greater than some fixed positive integer $k > 3$, the probabilty that a random insertion into an arbitrary failure node of an AVL tree of height j is rotational is greater than a fixed strictly positive constant (k) times

$$\alpha \prod_{j=k} \frac{1}{1 + \frac{2^{j-2}}{\gamma^{2^{j-2}}}}$$

where $\gamma = \sqrt[4]{3}$. In addition, the following question is examined: given an arbitrary height-balanced tree of height h, determine the probability that the deletion of an arbitrary node will imbalance the tree (see [11], for a discussion of deletion algorithms for AVL trees). Again, we employ infinite product developments to establish a lower bound for this probability. As an application of these methods we extend our results to obtain probability measures on generalized height-balanced trees called k-balanced trees. K-balanced trees differ from height-balanced trees in that the height difference of subtrees of nodes is allowed to vary in absolute value by some fixed positive integer k rather than one or zero for AVL trees.

A binary search tree is a finite set of nodes that is either empty or consists of a root and two disjoint binary trees called the left and right subtrees of the root. For example

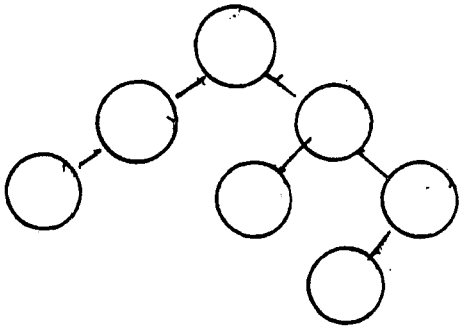


Figure 1.I.1

A sequence of real numbers $A = a_1, a_2, \ldots, a_n$ defines a binary tree T(A) as follows: if n = 0, T(A) is the empty tree; if n > 1, T(A) consists of

the root node containing α_1, a left subtree, $T(A_L)$ and right subtree $T(A_R)$ where

$$A_L = \alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_j}$$

is the subsequence of A consisting of all elements $< \alpha_1$, and

$$A_R = \alpha_{k_1}, \alpha_{k_2}, \ldots, \alpha_{k_m}$$

is the subsequence of A consisting of all elements $> \alpha_1$. For example, the tree in Figure 1.I.1 is defined by the sequence

$$A = \{6,2,1,9,8,13,11\}$$

Large key-ordered data sets (such as census data keyed by the Social Security number) are often maintained as tree structures. We can think of the key values as residing in the nodes of the tree.

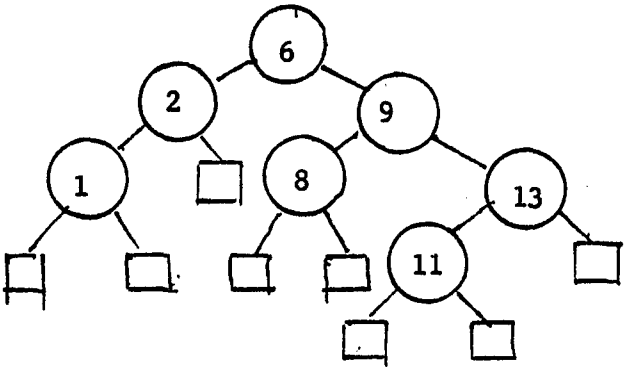


Figure 1.I.2

Note, we include boxes, or failure nodes, to symbolize the empty tree.

AVL, or height-balanced trees, have the following recursive definition: i) the empty tree is height-balanced; ii) a non-empty binary tree is height-balanced if and only if a) the height h_L of the left subtree T_L and the height h_R of the right subtree T_R satisfy $|h_L - h_R| \leq 1$; and b) T_L and T_R are themselves height-balanced. The tree in Figure 1.2 is an example of a height-balanced tree.

Height-balanced trees are useful for maintaining large key-ordered data sets for the following reasons. i) The cost of certain fundamental operations on these data sets such as searching for a key value in the set, and insertion and deletion of key values from the set can be shown to be proportional to the height of the tree. Height-balanced trees tend to minimize the height of the tree as a function of the number of nodes in the tree. ii) Height-balanced trees can easily be maintained as such under the operations of searching, insertion, and deletion. For example, suppose we add the number 0 to our input sequence.

$$A = \{6,2,1,9,8,13,11,0\}$$

To insert a new key value γ we proceed along a path from the root passing to the left subtree of a node $N(\beta)$ containing the value β if $\gamma < \beta$ and passing to the right if $\gamma > \beta$. This process terminates in an empty tree at some leaf and the insertion of $N(\gamma)$ is made there. In the example, the node for 0 would be placed to the left of the node for 1.

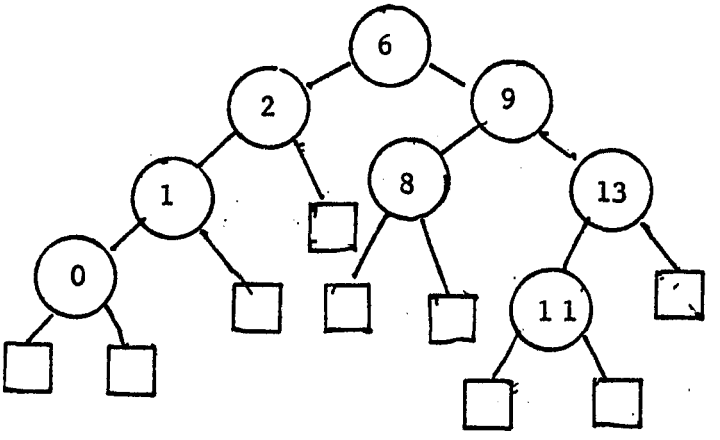


Figure 1.I.3

Note that after the insertion, the subtree rooted by the key value 2 no longer is height-balanced. To rebalance the tree a procedure known as a rotation is required. After the rotation the tree has the following shape:

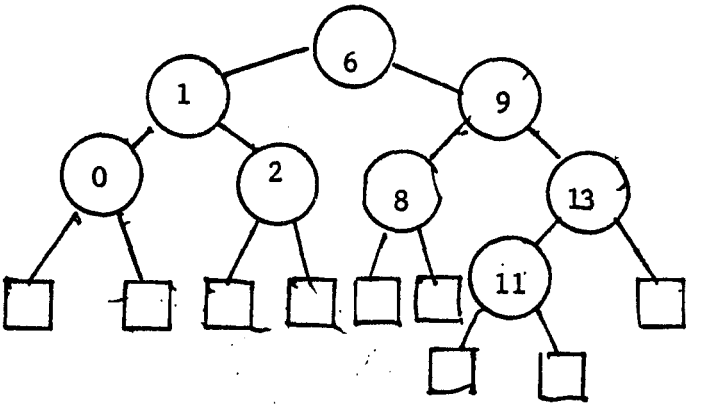


Figure 1.I.4

A rotation is essentially a finite number of parent-child pointer

changes, and based upon the location of the newly inserted leaf with respect to its nearest ancestor whose balance factor become greater than zero in absolute value, there are a finite number of rotation types which will always rebalance the tree. Let γ be inserted into an AVL tree and let $A(\gamma)$ be the ancestor of whose balance factor becomes greater than 1 in absolute value . Then we have the following characterization of rotation types

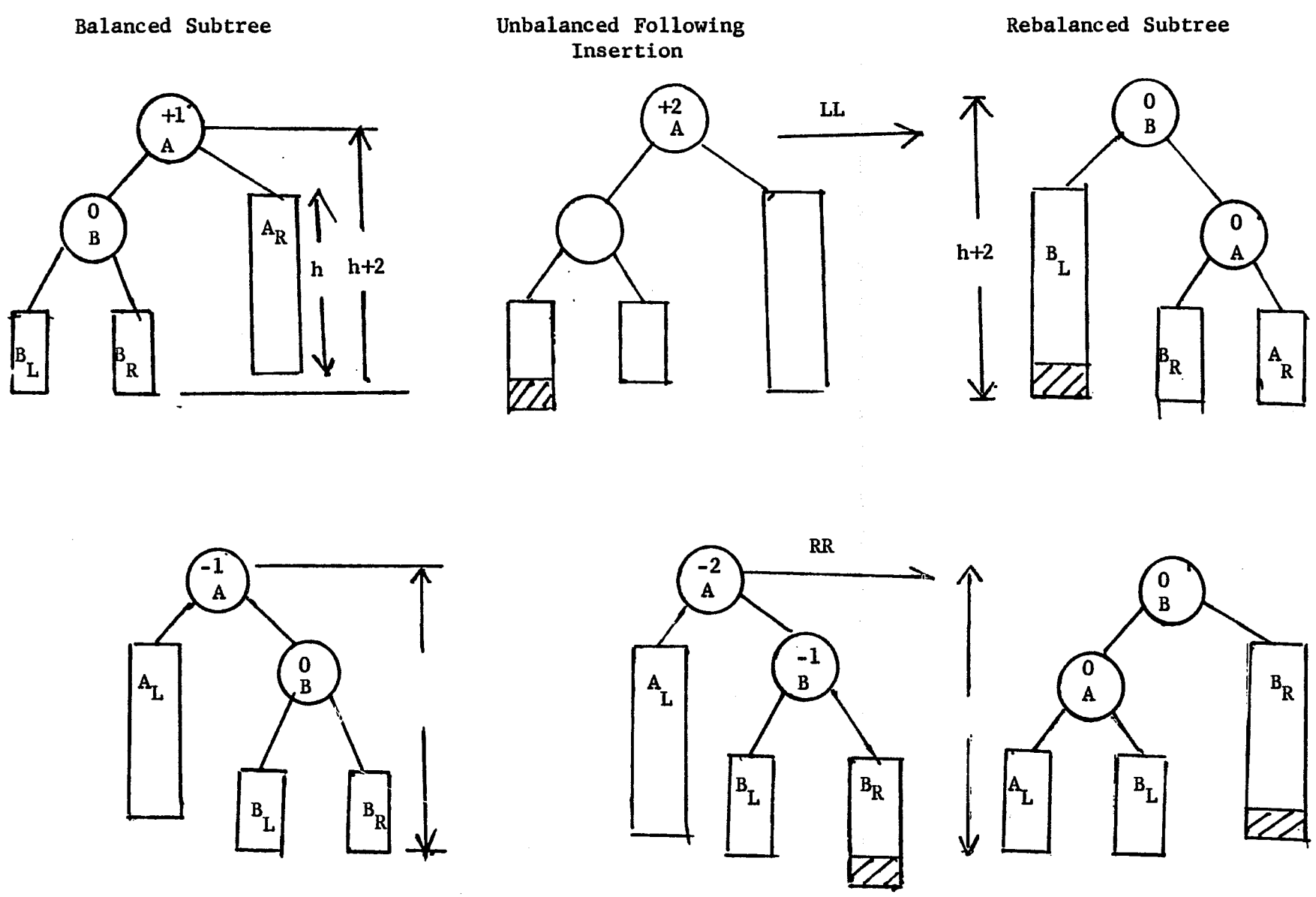
LL : the node for γ is inserted in the left subtree of the left subtree of $A(\gamma)$

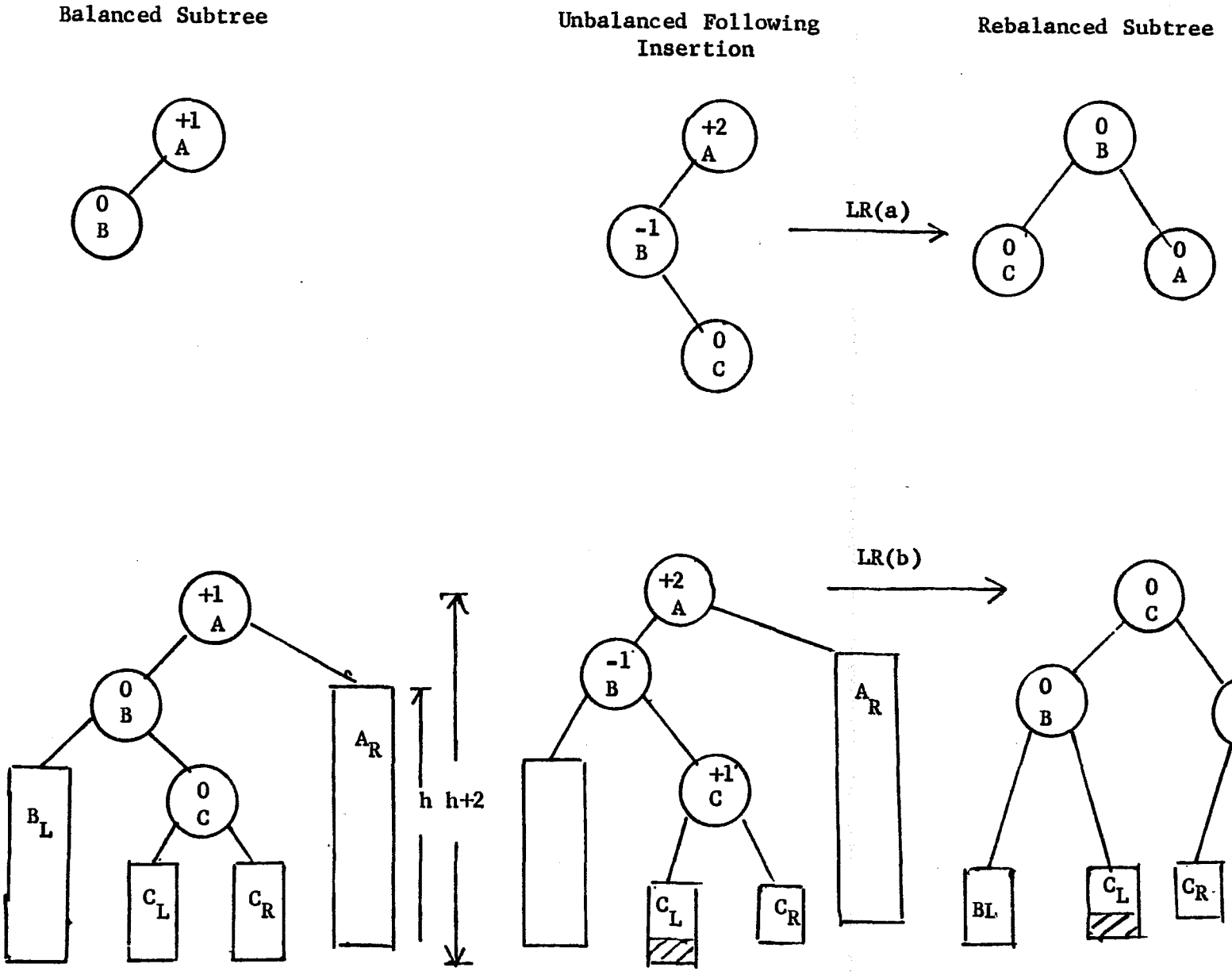
LR : the node for γ is inserted in the right subtree of the left subtree of $A(\gamma)$

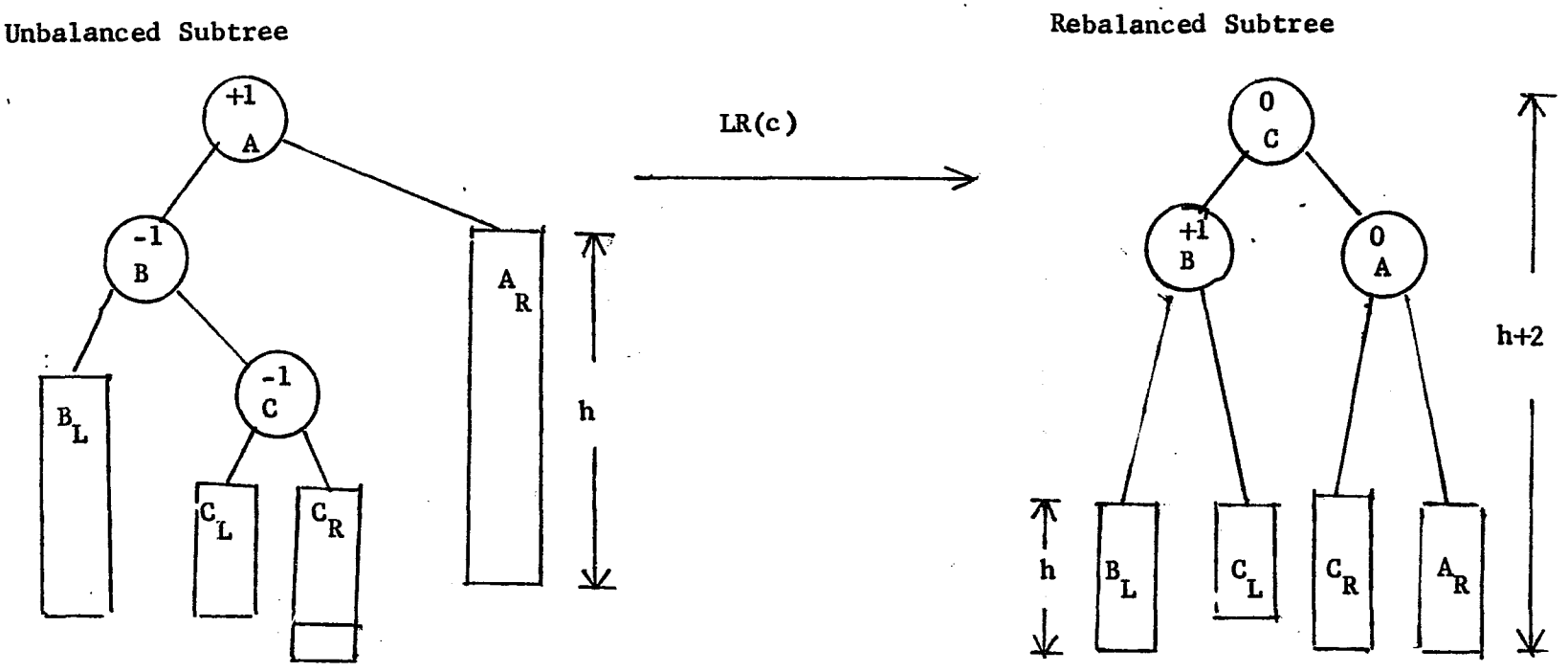
RR : the node for γ is inserted in the right subtree of the right subtree of $A(\gamma)$

RL : the node for γ is inserted in the left subtree of the right subtree of $A(\gamma)$

Below are a sequence of representations of these rotation types which demonstrate that they do indeed rebalance the tree. For RR and LL rotations the node A is the first ancestor of the newly inserted node with a non-zero ancestor and in the case of LR and RL rotations C is the the first ancestor of the newly inserted node with a non-zero balance factor. The cost of an insertion then divides into two parts, the cost of search and the cost of rebalancing. The cost of the search is proportional to the height of the tree since the same processing (a compare and a jump down to the next node along the path if a match is not found) occurs along each node of the path of search. More will be said of the cost of rebalancing in Chapter 2.


Balanced Subtree
Unbalanced Following Insertion
Rebalanced Subtree
+1
A
0
B
A_R
h
h+2
B_L
B_R
+2
A
LL
h+2
0
B
B_L
0
A
B_R
A_R
-1
A
A_L
0
B
B_L
B_R
RR
-2
A
A_L
-1
B
B_L
B_R
0
B
0
A
B_R
A_L
B_L


Balanced Subtree
+1
A
0
B
Unbalanced Following
Insertion
+2
A
-1
B
0
C
LR(a)
Rebalanced Subtree
0
B
0
C
0
A
+1
A
0
B
0
C
B_L
C_L
C_R
A_R
h
h+2
+2
A
-1
B
+1
C
C_L
C_R
A_R
LR(b)
0
C
0
B
-1
A
BL
C_L
C_R
A_R


Unbalanced Subtree
+1
A
-1
B
-1
C
B_L
C_L
C_R
A_R
h
LR(c)
Rebalanced Subtree
0
C
+1
B
0
A
h+2
h
B_L
C_L
C_R
A_R

In this chapter we survey some analytic and statistical results pertaining to AVL trees as well as generalizations of and alternatives to these data structures in database systems.

2.I ANALYSIS OF SEARCHING COSTS

From the discussion of Chapter 1 we conclude that the worst-case performance of search algorithms on height-balanced trees is proportional to the height of the tree. In this section we show that the height of any AVL tree with n nodes is $O(\log_2 n)$. We will also examine the implied constant which is of interest as we wish to know how far an arbitrary height-balanced tree is from the optimum. Here we follow [13] .

THEOREM 2.I.1 (Adelson-Vel'skii, Landis,[1]). The height of a height-balanced tree with n nodes always lies between $\log_2(n+1)$ and $1.4404*\log_2(n+2) - 0.328$.

Proof: A binary tree of height h obviously cannot have more than 2^h failure nodes; so $n+1 < 2^h$, as there are n+1 failure nodes in a tree with n nodes; or equivalently, $h > \log_2(n+1)$. In order to find the maximum value of h, consider the problem of finding the miminum number of nodes possible in a height-balanced tree of height h. Let T_h be such a tree of height h with the fewest number of nodes. Then one of the subtrees of the root, say the left subtree, has height h-1 or h-2. Since we wish T_h to have the minimum number of nodes, we may assume that the left subtree of T_h is T_{h-1}, which is minimal for height h-1, and that the right subtree is T_{h-2}, which is minimal for height h-2. Then if N(k) counts the number of nodes in a minimal tree of height k, N(k) satisfies the recursion

$$\begin{aligned} N(1) &= 2 \\ N(2) &= 4 \\ N(k) &= N(k-1) + N(k-2) \qquad k > 3, \end{aligned}$$

or equivalently, N(h) = F(h+2) where F(h) is the Fibonacci sequence. Again

$$F(h+2) - 1 = \phi^{h+2} / 5 - 2$$

where $\phi = (1+\sqrt{5})/2$. For a proof of this inequality see ([13], vol. 2, p. 343, and vol. 1, sec. 1.2.8).

To make a more refined analysis for seaching costs on height balanced trees consider the following argument. Let B_{nh} be the number of height-balanced trees of height h with n non-failure nodes. Then we compute the generating function $B_h(z) = \sum_{n=0} B_{nh}z^n$ for small h from the relations:

$B_0(z) = 1$

$B_1(z) = z$

$B_2(z) = z*B_h(z)*(B_h(z) + 2*B_h(z))$

Hence

$$B_2(z) = 2*z^2 + z^3$$

$$B_3(z) = 4*z^4 + 6*z^5 + 4*z^6 + z^7$$

$$B_4(z) = 16*z^7 + 32*z^8 + 44*z^9 + \ldots + 8*z^{14} + z^{15}$$

and in general $B_h(z)$ has the form

$$2^{F_{h+1}-1}*z^{F_{h+2}-1} + 2^{F_{h+1}-2}*L_{h-1} + \text{complicated terms} + 2^{h-1}*z^{2^h-2} + z^{2^h-1}$$

where $L_k = F_{k+1} + F_{k-1}$ for $h > 3$. The total number of height-balanced trees with height h is $B_h = B_h(1)$ which satisfies the recurrence

$B_0 = 1$

$B_1 = 1$

$B_{h+1} = B_h + 2*B_h*B_{h-1}$

Assume that each of the B_h height-balanced trees of height h is equally likely. Then we use an argument due to Kzidan [13] to show

THEOREM 2.I.2 The average number of nodes in a height-balanced tree of height h is

$$B'(1)/B(1) \sim (0.70118)*2^h$$

where B' denotes the derivative of B.

Proof: Let $b_h = B_h'(1)/B_h(1) + 1$ and let ε_h be the very small quantity $2*B_h*B_{h-1}*(b_h - b_{h-1})/B_h$. Then $b_h = 2*b_{h-1} - \varepsilon_h$; backsubstituting in this formula we obtain $b_h = 2^h*(1 - \varepsilon_1/2 - \varepsilon_2/4 - \ldots) + r = \varepsilon_{h+1}/2^{h+1} + \varepsilon_{h+2}/2^{h+2} + \ldots$ is extremely small for large h.

This indicates that the height of a balanced tree with n nodes is usually much closer to $\log_2 n$ than $\log_\varphi n$. Unfortunately, this result cannot be directly applied to insertion algorithms on AVL trees since the mechanism of these algorithms seems to make some trees much more likely than others. Note that there are 7! = 5040 possible orderings in which seven orderings in which seven keys can be inserted into a height-balanced tree and the perfectly balanced "complete" tree

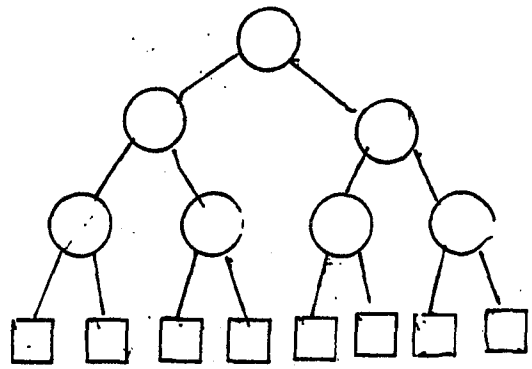


Figure 2.I.1

is obtained 2160 times. By contrast, the minimal tree of height 3 with seven nodes

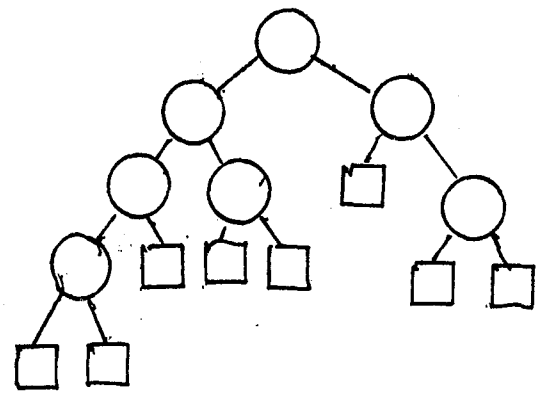


Figure 2.I.2

occurs only 144 times, and the similar tree

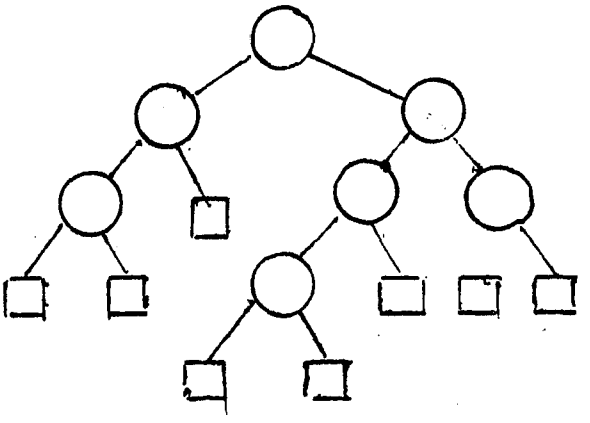


Figure 2.I.3

occurs 216 times.

The fact that the perfectly balanced tree appears to be obtained with such high probability - together with THEOREM 2.I.2 - makes it extremely plausible that the average search time for a height-balanced three with nodes is about $\log_2 n + c$ for some small constant c. Empirical evidence supports this conjecture: the average number of comparisons needed to insert the n'th element seems to be approximately $\log_2 n + .25$ for large n [13].

II ANALYSIS OF REBALANCING

From the discussion of Chapter 1, we conclude that the rebalancing procedure for insertion of a new element into an AVL tree has two cost components: 1) the cost of the pointer changes from the newly inserted node to

its first ancestor (before the insertion) with a non-zero balance factor; and 2) the cost of the pointer changes for the subtrees rotated around that ancestor. In this section we recount some partial analytic results due to Brown[7] on the costs of rebalancing.

Define an internal (non-failure) node in a height-balanced tree to be a fringe node if at least one of its offspring is a failure node; the set of all fringe nodes is called the fringe of the tree. When the non-failure nodes are removed from an AVL tree, the fringe becomes a collection of disjoint subtrees, each one having one of the forms shown in figure 2.II.1. An M-subtree is rooted by a fringe node which is unbalanced, while an N-subtree is simply a balanced fringe node.

While the analysis in this section is performed in terms of M-subtrees and N-subtrees rather than directly in terms of balanced and unbalanced fringe nodes, it is clearly possible to translate between the two views.

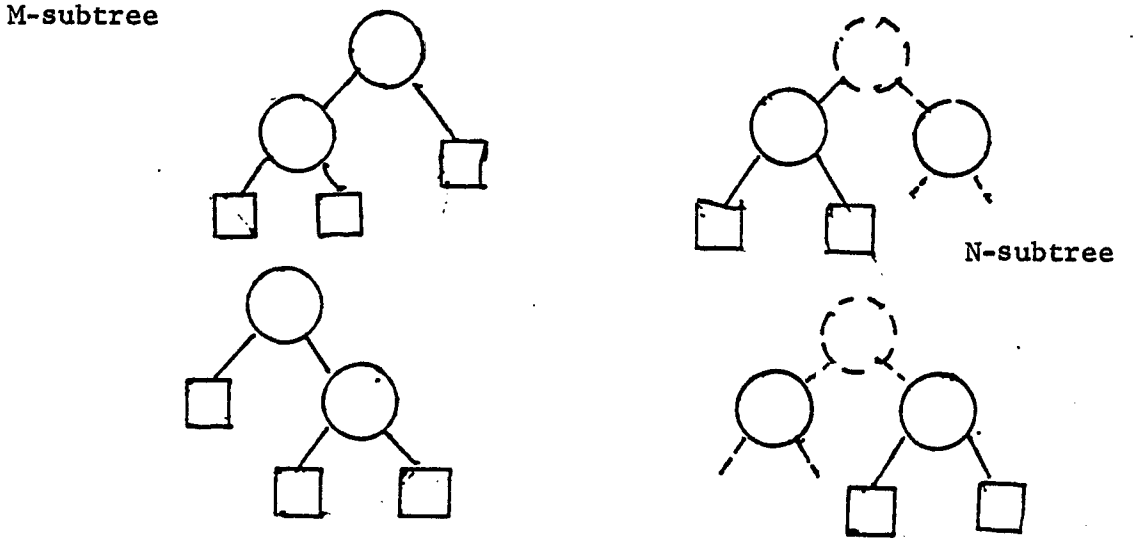


Figure 2.II.1

LEMMA 2.II.1 Let T be an n node height-balanced tree with M M-subtrees and N N-subtrees where n > 0. Then

$$3*M + 2*N = n+1$$

Proof: Each M-subtree has three failure nodes, and each N-subtree has two; hence the left-hand side of the equation counts the number of failure nodes in T. But a simple proof by induction shows thatan n-node height-balanced tree has n+1 failure nodes.

LEMMA 2.II.2 An insertion which falls into a failure node of an M-subtree reduces the number of M-subtrees by one and increases the number of N-subtrees by two.

Proof: By observation of Figure 2.II.1 and the rotation diagrams of Chapter 1, section IV.

LEMMA 2.II.3 An insertion which falls into an failure node of an N-subtree reduces the number of N-subtrees by one and increases the number of M-subtrees by one.

Proof: When an insertion falls into either of the failure nodes of an N-subtree, the N-subtree is transformed into an M-subtree , but rebalancing may take place higher up the tree since the root of an N-subtree is balanced. It remains to determine what effect rebalancing will have upon the fringe.

Again, if we refer to the rotation diagrams of Chapter 1, section IV, we observe that rebalancing has no effect on nodes which lie outside of the rebalanced subtree, and if the fringe of the rebalanced subtree is contained entirely in the subtree, then we can see that rebalancing has no effect on the fringe. We can also see that the only case in which the fringe of the rebalanced tree is not totally contained in this subtree is the case of an RL rotation or an LR rotation in a subtree of height 3. Fortunately, there is only one such tree (and its mirror image) in which this case occurs. The possible insertions which cause a rotation are shown in Figure 2.II.2. In both situations the net effect on the entire fringe is to eliminate one N-subtree and to introduce one M-subtree.

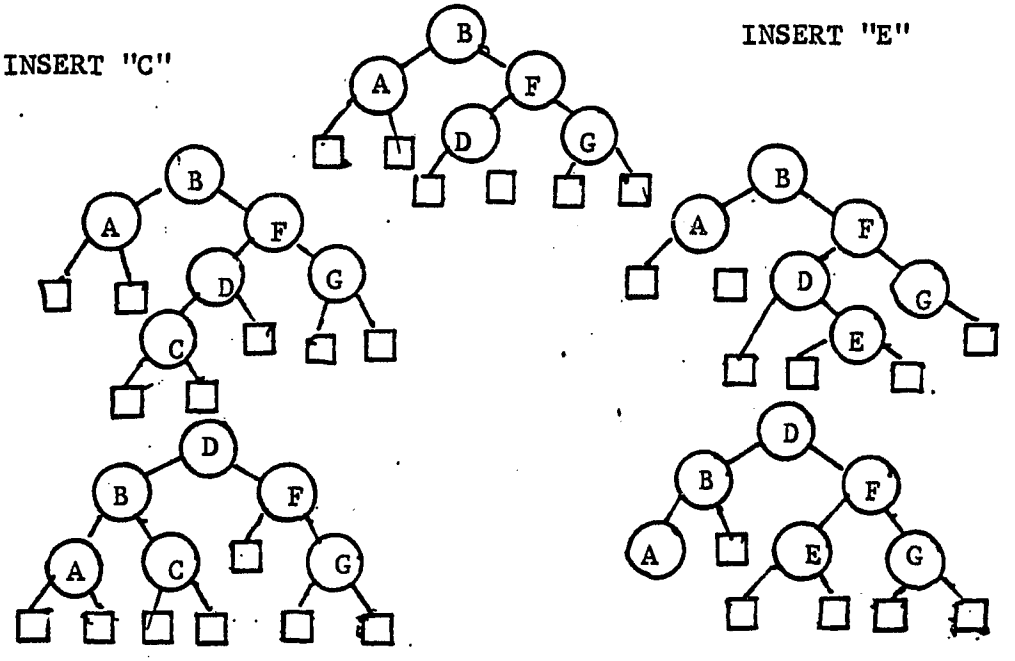


Figure 2.II.2

Now it is easy to determine the average number of M and N subtrees generated by a random permutation of an input set of size n. Let $P_n(N,M)$ denote

probability that a random height-balanced tree with n nodes contains M M-subtrees and N N-subtrees in the fringe. If we define

$$N(n) = \sum_{M,N} N * P_n(N,M)$$

then N is just the average number of N-subtrees in the fringe of a random height-balanced tree with n nodes. The same quantity for M is defined analogously.

THEOREM 2.II.1 N(n) = (2/7)*(n+1) and M(n) = (1/7)*(n+1) for n > 6.

Proof: Let T be an n node AVL tree with M M-subtrees and N N-subtrees in the fringe for some n > 0. Then by LEMMAs 2.II.2 and 2.II.3 the next insertion into T changes the number of N-subtrees to N-1 or N+2 depending on whether the insertion falls into an N-subtree or an M-subtree. For a random insertion into T (that is, all failure nodes are equally likely candidates for the new insertion), these events occur with probabilities 2*N/(n+1) and 3*M/(n+1) = ((n+1) - 2*N)/(n+1) respectively so

$$N(n) = \sum_{M,N} P_n(N,M)*((2*N/(n+1))*(N-1) + (1-(2*N)/(n+1))*(N+2))$$

$$\sum_{M,N} P_n(N,M)*(N-(6*N/(n+1)) + 2)$$

$$= \quad (1 - 6/(n+1))*N(n) + 2.$$

A proof by induction shows that $N(1) = 1$ and $N(n) = (2/7)*(n+1)$ for $n > 6$.

This theorem allows us to confirm the accuracy of some empirical results on random height-balanced tree given by Knuth [13]. In section 3,6.2.3, Table 1., it is given that the probability that a random insertion into a large random height-balanced tree falls into an M-subtree and causes either 1) no rebalancing, 2) an LL or RR rotation or 3) an LR or RL rotation is appproximately .144 in each case. The following corollary shows that 1/7 is the exact formula for every insertion after the sixth.

COROLLARY 2.II.1 The probability that a random insertion into a random height-balanced tree of size n falls into an M-subtree is 3/7 for $n > 6$.

Proof: Since an M-subtree has three failure nodes, the probability is

$$3/(n+1)*(\sum_{M,N} M*P_n(N,M)) = (3/(n+1))*(\sum_{M,N} M*P_n(N,M))$$

$$= \quad (3/(n+1))*M(n) = 3/7$$

for $n > 6$ by THEOREM 2.II.1

III ALTERNATIVES TO HEIGHT-BALANCED TREES

In the wake of the Adelson Velskii-Landis paper [1] other classes of binary trees have been suggested as data structures for FIND-ADD-DELETE operations which like AVL trees require at most logarithmic time. C.C. Foster [10] has suggested generalized height-balanced trees which arise when we allow the difference of subtrees to vary by as much as some fixed positive integer k (see Chapter 5.III for an analysis of k-balanced trees).

Weight-balanced trees have been studied by J. Nievergelt, E. Reingold, and C.K. Wong [18]. Instead of considering the the height of binary trees, they require that the subtrees rooted by all nodes of the tree satisfy

$$\sqrt{2}-1 < \frac{\text{no. of nodes in left subtree}}{\text{no. of nodes in right subtree}} < \sqrt{2}+1$$

The weight-balance of a tree can be maintained under insertion using the rotations for rebalancing height-balanced tree, but after a single insertion the number of rotations required to rebalance the tree may be as many as $\log_2(n)$ where n is the number of nodes in the tree.

The tree search methods discussed so far were developed primarily for searching a computer file maintained entirely within a computer's high speed main memory. When the file is maintained on an external storage device such as a disk, the cost of the disk access during the search procedure becomes important.

If one disk access per node of the height-balanced tree were required during the search process and each disk access took an average of .25 seconds, then the seach procedure for large data sets could become very time consuming. An alternative tree structure for large data sets which minimizes the number of disk accesses in the search process was discovered in 1970 by R. Bayer and E. McCreight [6]. Their data structure, called a B-tree of order m, satisfies the following properties.

i) Every node has $<$ m sons.

ii) Every node, except for the root and leaves, has $>$ m/2 leaves.

iii) The root has at least 2 sons (unless it is a leaf).

iv) All leaves appear on the same level.

v) A non-leaf node with k sons contains k-1 key-variables values.

Below is an example of a B-tree of order 3.

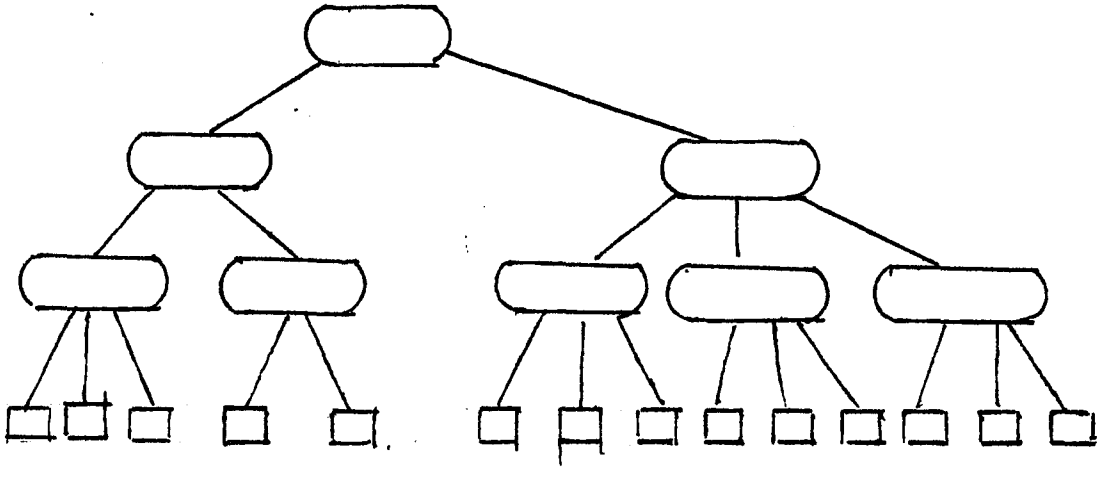
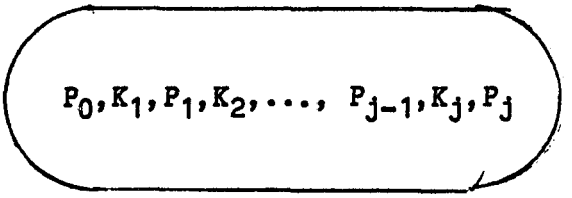


Figure 2.III.1

A node which contains j keys and j+1 pointers can be represented as



1)

Figure 1.III.2

where $K_1 < K_2 < \ldots < K_j$ and P_k points to the subtree for keys between K_k and K_{k+1}, k= 0, ...,j. Then searching a B-tree is quite straightforward. After node 1) has been fetched into main memory from the disk, the algorithm searches for the given argument among the keys K_1, K_2, ..., K_j. Sequential rather than binary search would be appropriate for small j. If the key value being searched lies in the set K_1, K_2, ..., K_j, the search procedure terminates. Otherwise, the key value in question lies between some K_r and K_{r+1} and we procede to search the keys of the subtree to which P_r points.

If we wish to insert a new item into a B-tree of order m, where all the leaves are at level q, we insert the next key J into the appropriate node at level q-1. If the node now contains m keys so that it has the form 1) with j=m, an attempt is made to displace J in the parent node of its node via the following method: an attempt is made first to move J into its parent node at the appropriate position and to move the key M_i immediately at its left (if it exists) to the rightmost key position in the root of the subtree to which M_i originally pointed. If that node is full a similar attempt is made to move J into the parent node moving M_{i+1} into the root of the subtree to which it points. If neither of these transfers is feasible the node N containing J splits into N and N'

and the key J is inserted into the father of the original node. Note the pointer P_{i+1} in the father node is replaced by the sequence P_{i+1}, P'_{i+1}, where P_{i+1} points to the split node N and P'_{i+1} points to N'. With this change the father of the original node may itself obtain too many keys. In this event it must split in the same fashion its offspring split. The splitting and re-insertion process may continue all the way up to the root of the tree. If the root becomes full in this process, the algorithm splits it and creates a new root with the single key value V, (see [11], pp. 496-510, for an illustration of this phenomenon). For a discussion of deletion procedures for B-trees see [11], pp. 511-517.

The worst case performance of search algorithms for B-trees involves a simple analysis. Suppose that there are N keys and the N+1 leaves appear at level q. Then the number of nodes on levels 1,2,3, ... is at least $2, \lceil 2*m/2 \rceil, \lceil 2*(m/2)^2 \rceil, \ldots$; hence

$$N+1 > \lceil 2*(m/2)^q \rceil$$

or equivalently

$$q < \lceil \log_{m/2}(N+1)/2 \rceil$$

Now we estimate the average number of splittings required to insert n keys

into a B-tree of order m. If there are p nodes then there are at least $(\lceil m/2 \rceil - 1)(p-1)$ keys; hence

$$p < \frac{N-1}{\lceil m/2 \rceil - 1}$$

Then the average number of times splitting is required is less than $1/(\lceil m/2 \rceil - 1)$ per insertion.

CHAPTER 3 FRINGE ANALYSIS

In this chapter we examine many of the combinatorial implications of fringe analysis for AVL trees. For example we show that the fringe distribution function is unimodal. Furthermore we estimate and obtain recursive formulae for sums of the type $P_n(N,M)*P(N,M)$ where P(N,M) is a polynomial in N and M

I Cumulative Analysis of Fringe Sets

It is possible to relate fringe analysis for algorithms on AVL trees to a cumulative analysis of algorithms on these data structures. That is, given an arbitrary height-balanced tree with n nodes and N N-subtrees and M M-subtrees generated by a sequence of random insertions into an initially empty tree, determine bounds for the maximal number of rotations (of special types) which could posssibly have arisen in the generation of the tree. We will characterize a rotation as being of height k, if the path, after the insertion but before the rotation, of the newly inserted node to its first ancestor with a non-zero balance factor is of length k.

THEOREM 3.I.1 Given a height-balanced tree T with M M-subtrees and N N-subtrees at most N+M rotations of height 2 could have occurred in the generation of the tree and at most N + 2*M rotations of height 3 or greater could have occurred.

REMARK. This last bound is nontrivial since by LEMMA 2.II.1 3*M + 2*N = n+1.

Proof: By LEMMA 2.II.2 an insertion which lands in an M-subtree increases the number of N-subtrees by 2 and decreases the number of M-subtrees by 1. Call this an M-insertion. By LEMMA 2.II.3 an insertion which lands in an N-subtree increases the number of M-subtrees by 1 and decreases the number of N-subtrees by 1. Call this an N-insertion. Inspection of the rotation diagrams in Chapter 1 shows that a rotation of height 2 can be the consequence of an M-insertion only. Let MI count the number of M-insertions in the generation of the tree (in this case the number of N-subtrees increases) and let NI count the number of N-insertions (in this case the number of M-subtrees increases) Assume that every M-insertion causes a rotation of height 2. Then NI counts the number of rotations of height 2. By the discussion above we have the following relations

$$2*MI - NI = N$$

and

$$NI - MI = M$$

Substituting for MI in the first equation we obtain

$$MI - M = N$$

or equivalently

$$MI = N + M$$

Now assume that every N-insertion generates a rotation of height 3 or greater. Then the same equations give a bound for the maximal number of rotations of height 3 or greater.

$$NI = 2*M + N$$

This proves the following

COROLLARY 3.I.1 The maximal number of rotations of height 2 which can arise in the generation of a height-balanced tree with n nodes is less than (n+1)/2 and the maximal number of rotations of height 3 or greater which can occur is less than 2*(n+1)/3.

Proof:By LEMMA 2.II.1 we have the relation 2*N + 3*M = n+1. Hence we maximize the expression N+M in THEOREM 3.I.1 by taking M=0 and we maximize the expression N + 2*M by taking N = 0.

Suppose a sequence of insertions generates a height-balanced tree with n nodes, N N-subtrees and M M-subtrees. Assume further that at any stage of the insertion process the failure nodes of the current tree are equally likely to be the location of the next insertion. In particular, if the new element is to be inserted into an M-subtree, it is twice as likely that the insertion will force a rotation of height 2 than that it will cause a no rotation at all (note that there are two failure nodes in the taller of the subtrees of an M-subtree and there is one failure node in the shorter of the subtrees). From the previous formulae we know that in the generation of a height-balanced tree with N N-subtrees and M M-subtrees N + M of the insertions generation T landed in M-subtrees. This fact coupled with our probability assumptions allows us to obtain explicit formulae for the expected number of rotations of height 2 occurring in the generation of T.

THEOREM 3.I.2 Let T be a height-balanced tree with n nodes N N-subtrees, and M M-subtrees. Then the expected number of rotations of height 2 which would take place in the generation of T is $(2/3)*(N+M)$.

Proof: The function

$$b(k,N+M,2/3) = \begin{cases} \binom{N+M}{k} *(2/3)^{k}*(1/3)^{N+M-k} & \text{for } k = 0,1, \ldots, N+M \\ 0 & \text{otherwise.} \end{cases}$$

is the probability mass function in question. Then if we follow [19], p.208, its mean is given by

$$\sum_{k=0}^{N+M} k*p(k) = \sum_{k=0}^{N+M} k*\binom{N+M}{k} *(2/3)^{k}*(1/3)^{N+M-k}$$

$$= \sum_{k=0}^{N+M} \binom{N+M-1}{k-1} * (2/3)^{k-1}*(1/3)^{(N+M-1)-(k-1)}$$

$$= (N+M)*(2/3)*(2/3 + 1/3)^{N+M-1} = (2/3)*(N+M)$$

THE FRINGE DISTRIBUTION FUNCTION II

Now consider the following probabilistic question: for M and N such that $3*M + 2*N = n+1$, what is the probability that n insertions made into an initially empty height-balanced tree will generate a tree with M M-subtrees and N N-subtrees? This is the density function $P_n(N,M)$ Brown uses to establish the densities of M- and N-subtrees in trees with n nodes. In this section we shall establish that its distribution is unimodal.

For the first result the following LEMMA is required.

LEMMA 3.II.1 The function $P_n(N,M)$ described above satifies the following relations:

$$P_n(0,0) = 0$$

$$P_n(0,1) = 1$$

$$P_n(1,0) = 2$$

$$P_n(N,M) = 2*((N+1)*P_n(N+1,M-1) + 3*(M+1)*P_n(N-2,M+1))/(n+1) \quad M > 1, \quad N > 2$$

$$= 3*P_n(N-2,M+1)*3(M+1)/(n+1) \quad N > 2, \; M = 0$$

$$= 2*P_n(N+1,M-1)*2*(N+1)/(n+1) \quad N = 0,1 \; M > 1$$

Proof: Given an arbitrary height-balanced tree with n nodes generated by insertions, the n'th insertion had one of two outcomes: i) the number of N-subtrees increased by two and the number of M-subtrees decreased by one (the insertion landed in an M-subtree) ; or ii) the number of N-subtrees decreased by one and the number of M-subtrees increased by one (the insertion landed in an N-subtree). For case i) $N > 2$, $M > 1$, the term $3*(M+1)*P_n(N-2,M+1)$ counts the number of distinct insertion sequences which could have generated a predecessor tree with n-1 nodes, N-2 N-subtrees and M+1 M-

subtrees. Note the factor M+1 derives from the possible candidate M-subtrees for the n-insertion and the factor 3 reflects the fact that an insertion landing in any of the tree failure nodes of an M-subtrees will transform it into two N-subtrees. For case ii), N > 2, M > 1, the term $2*(N+1)*P_n(N+1,M-1)/(n+1)$ counts the density of sequences which could have generated a predecessor tree with N+1 N-subtrees and M-1 M-subtrees times the number of ways (2*(N+1)) the n'th insertion could land in an N-subtree and generate an M-subtree. The border condition formula (M = 0) follows from the observation that a tree with no M-subtrees could only have been generated by an n'th insertion landing in the sole M-subtree of its predecessor tree in the insertion sequence. Similarly, the border condition (N = 0,1) follows from the observation that a height-balanced tree with 0 or 1 n-subtrees could only have been generated by an n'th insertion landing in an N-subtree of a tree with either two (N = 1) or one (N = 0) N-subtrees.

Let F(N,n) equal $P_n(N,M)$ over all permissable values of N and M for fixed n. Now we can prove the following

PERMISSIBLE VALUES OF N

n k mod 6	N = ...
0	2, 5, 8, ..., (n-2)/2
1	1, 4, 7, ..., (n+1-6)/2
2	0, 3, 6, ..., (n-2)/2
3	2, 5, 8, ..., (n+1-6)/2
4	1, 4, 6, ..., (n-2)/2
5	0, 3, 6, ..., (n+1-6)/2

Table 3.II.1

THEOREM 3.II.2 For all $n > 0$, function $F(n,N)$ over permissable N is unimodal.

Proof: For each of the congruence classes the initial sequence for $F(n,N)$ is given below

n	F(n,N)*n = ...
1	1
2	2
3	6
4	24
5	48,72
6	720

Table 3.II.2

The Figure serves as a basis for a proof by induction on n. For the induction step, we assume that for $n > 6$, the sequence is unimodal. Then we construct a map M : $F(n,N) \longrightarrow F(n+1,N)$ which preserves the unimodality of F(n,N). To facilitate the proof we will demonstrate the unimodality of sequences F(n,N) which differ from F(n,N) by the presence of single zeroes attached at the beginning or the end of the sequence depending on the residue of n mod 6.

By the induction hypothesis for some $n > 6$, the sequence F(n,N) is unimodal. Let F(n+1,k) be the k'th member of the sequence F(n+1,N). Then by LEMMA 3.II.1

$$F(n+1,k) = (F(n,k+1)*(2*k + 2) + F(n,k-1)*(n - 1 - 2*k))/(n+1)$$

$$\text{for } 0 < k < n$$

$$= F(n,1)*2/(n+1)$$

for $k = 0$

$$= F(n,(n-1)/2)*(n-1)/(n+1)$$

for $k = n$

Then we can write

$$F(n+1,k) = \alpha * F(n,k+1) + \beta * F(n,k)$$

where $\alpha + \beta = 1$. Then F(n+1,k) lies between the values of F(n,k+1) and F(n,k-1) as it is a convex combination of their values. By the induction hypothesis, the sequence F(n,k) is unimodal; hence the sequence F(n+1,k) is also unimodal. Note that an extra zero must be appended either to the beginning or to the end of the derived series for F(n+1,k) as one of the zeroes will be lost in the transition.

Curiously, the fringe analysis model for height-balanced trees corresponds to a specific case of a class of urn models for aftereffect well-known in probability theory. From [9], we have the following description of these models: "Consider an industrial plant liable to accidents. The occurrence of an accident might be pictured as the result of a superhuman game of chance: Fate has in store an urn containing red and black balls; at regular time intervals a ball is drawn at random, a red ball signifying an accident. If the chance of an accident remains constant in time, the composition of the urn is always the same. But it is conceivable that each

accident has an aftereffect in that it either increases or decreases the chance of a new accident. This corresponds to an urn whose composition changes according to certain rules that depend upon the outcome of successive drawings."

Many of the cases are covered by the following model: an urn contains b black balls and r red balls. A ball is drawn at random. It is replaced. If it is red then c_1 red balls are added and d_1 black balls are added. If it is black, then d_2 black balls are added and c_2 red balls are added. Note that c_1, c_2, d_1, and d_2 are arbitrary integers and may be chosen negative. In terms of Brown's fringe analysis model the failure nodes of M-subtrees would correspond to red balls and the failure nodes of N-subtrees would correspond to black balls.

III FRINGE POLYNOMIALS

In his fringe analysis for height-balanced trees Brown obtains explicit formulae for expressions of the form

$$\sum_{M,N} P_n(M,N) * N$$

$$\sum_{M,N} P_n(M,N) * M$$

where $P_n(M,N)$ is the density of height-balanced trees with n nodes

containing M M-subtrees and N N-subtrees. In this section we indicate a method to evaluate sums of the form

$$\sum P_n(M,N) * \mathcal{P}(M,N)$$

where $\mathcal{P}$ is a polynomial M and N. As an illustration of the method we will prove the following

THEOREM 3.III.1 $\sum_{N,M} P_n(N,M)N^2 > (4/49)*(n+1)^2$.

Proof: The variance of N is

$$\sum_{N,M} P(N,M)*(N-N(n))^2$$

We can expand it as

$$\sum_{N,M} P_n(N,M)*N^2 + \sum_{N,M} P_n(N,M)*N(n)^2 + -2(\sum_{N,M} P_n(N,M)*N*N(n))$$

By THEOREM 2.II.1 $N(n) = (2/7)*(n+1)$ and $\sum P_n(N,M) = 1$ so the second term equals $(4/49)*(n+1)^2$. Applying the same result we obtain that the third sum equals $-4*((n+1)/7)*(\sum P_n(N,M)*N)) = -8*(n+1)^2/49$. But the variance must be positive so

$$\sum P_n(N,M)*N^2 > (4/49)*(n+1)^2$$

THEOREM 3.III.2 Let

$$N^2(n) = \sum_{M,N} P_n(M,N)*N^2$$

$$M^2(n) = \sum_{M,N} P_n(M,N)*M^2$$

and

$$NM(n) = \sum_{M,N} P_n(M,N)*N*M$$

Then

$$N^2(n+1) = 3*N^2(n) + (16/7)*n * 12*NM(n)$$

where NM(n) can itself be expressed as linear combinations of $N^2(n-1)$, $M^2(n-1)$, NM(n-1) and n.

Proof: Let T be an arbitrary height-balanced tree with N N-subtrees and M M-subtrees. Then the coefficient of T in $\sum P_n(M,N)*N^2$ is N^2. Suppose the

n+1'st insertion into T lands in an N-subtree. Then the coefficient of the image T' of T under this insertion map is $(N-1)^2$. There are n+1 ways this can occur. If the n+1'st insertion lands in an M-subtree the coefficient of T' is $(N+2)^2$. Then

$$N^2(n+1) = \sum_{M,N} P_n(N,M)*(3*M*(N+2)^2/(n+1))+(2*N*(N-1)^2)/(n+1))$$

$$= \sum_{M,N} P_n(N,M)((3*M+2*N)*N^2/(n+1)+12*\sum_{M,N} P_n(N,M)*M*N/(n+1)$$

$$- 4*\sum_{M,N} P_n(N,M)*N^2/(n+1) + 12*\sum_{M,N} P_n(N,M)*M/(n+1)$$

$$+ 2*\sum_{M,N} P_n(N,M)*N/(n+1)$$

$$= N^2(n) + 12*MN/(n+1) - 4*N^2(n)/(n+1) + 16/7$$

if we use THEOREM 2.II.1 to evaluate the expressions for $P_n(M,N)*N$ and $P_n(M,N)*M$. We also have a recursive formula for NM(n).

$$NM(n+1)=\sum_{M,N} P_n(N,M)*(3*M*(N+2)*(M-1)/(n+1)+2*N*(N-1)*(M+1)/(n+1)$$

$$=\sum_{M,N} P_n(N,M)*N*M + \sum_{M,N} P_n(N,M)*6*M /(n+1)$$

$$-5*\sum_{N,M} P_n(N,M)*N*M/(n+1) \quad - \quad 6*\sum_{N,M} P_n(N,M)*M^2/(n+1)$$

$$-2*\sum_{N,M} P_n(N,M)*N/(n+1) \quad + \quad 2*\sum_{N,M} P_n(N,M)N^2/(n+1)$$

$$= \quad NM(n) + \quad 6*M^2(n)/(n+1) \; -5*NM(n)/(n+1)$$

$$+ \; 2*N^2(2)/(n+1) \quad -10/7$$

Note the constant -10/7 follows from the terms $-6\sum P_n(N,M)*M/(n+1)$ $= -(6/7)*(n+1)/(n+1)$ and $-2\sum P_n(N,M)*N/(n+1) = (4/7)*(n+1)/(n+1)$. A similar derivation gives the following recursive formula for $M^2(n)$.

$$M^2(n+1) = M^2(n) + 4*NM(n)/(n+1) - 6*M^2(n)/(n+1) + 1$$

IV DOUBLE AVERAGING FRINGE ANALYSIS

In section 3.I the expected number of rotations (2/3)*(N+M) of height 2 occuring in the generation of a height-balanced tree with n nodes , N N-

subtrees, and M M-subtrees was calculated. Suppose now we wish to obtain the expected number of rotations over all AVL trees with n nodes.

THEOREM 3.IV.1 The expected number of rotations of height two occuring in the generation of an AVL tree with n nodes is (2/7)*(n+1)

Proof: Let $T_n(N,M)$ enumerate AVL trees with N N-subtrees and M M-subtrees and let T(n) count all AVL trees with n nodes. Then the average number of rotations of height 2 occurring in the generation of trees with n nodes is

$$\left(\sum_{N,M} T_n(N,M)*(2/3)*(N+M)\right)/T(n)$$

$$= (2/3)*\left(\sum_{N,M} P_n(N,M)(N+M)\right)$$

as $T_n(N,M)/T(n) = P_n(N,M)$. By THEOREM 2.II.1 and COROLLARY 2.II.1 the expression $\sum P_n(N,M)*(N+M)=(3/7)*(n+1)$. Reducing the fraction we obtain the result.

We can extend this averaging method to a general Markov model with certain restrictions on its rules of transition.

Let $\mathcal{U}$ be an urn model for aftereffect containing balls of colors b_1, b_2, ..., b_k. For each i, i= 1, ..., k, when a ball of color b_i is chosen from theCurn the following transitions occur: if a ball of color b_i is chosen n_{i1} balls of color b_1 are added to the urn, n_{i2} balls of color b_2 are chosen, ...,

and n_{ik} balls of color b_k are added for i= 1, ..., k. Furthermore, $\sum_j n_{ij} = k$ for all j= 1, ..., k. Let the matrix $[a_{ij}]$ represent the transitions in the composition of η under the various selections of balls. Let $N_i(n)$ represent the average number of balls of color b_i in urn compostions generated by insertions from an initial state S. Assume, again, that there exist real constants $c_1, c_2, \ldots, c_k$ such that for some r, $N(r) = c_r * n$, r= 1, ...,k and the vector $(c_1, c_2, \ldots, c_k)$ is an eigenvector with eigenvalue unity of the matrix $[a_{ij}/k]$. Then we have the following result.

THEOREM 3.IV.2 Let η be defined as above with initial state S, $|S| = k$. Then exist constants $c_1, c_2, \ldots, c_k$ such that for any n greater than some fixed integer r, $N_i(n) = c_i * n$.

Proof: Let $P_n(N_1, N_2, \ldots, N_k)$ denote the density of urns with compositions consisting of N_1 balls of color b_1, N_2 balls of color b_2, ..., and N_k balls of color n_k. Then

$$N_i(n+1) = \sum\sum P_n(N_1, N_2, \ldots, N_k) * N_j * (N_i + a_{ij})/k * n$$

$$= \sum_j \sum P_n(N_1, N_2, \ldots, N_k) * N_i * (N_j)/k * n$$

$$+ \sum_j \sum_j P_n(N_1, N_2, \ldots, N_k) * a_{ij} * N_i/k * n$$

$$= N_i(n) + \sum a_{ij} N_j(n)/k * n$$

By the induction hypothesis, $N_i(n) = c_i(n)$, $i = 1,\ldots,k$, and the vector $(c_1, c_2, \ldots, c_k)$ is an eigenvector of $[a_{ij}/k]$ with eigenvalue 1. Hence the summation $\sum_j a_{ij}N_j(n)/k^n = c_i$. This implies $N_i(n+1) = c_i^*(n+1)$, $i = 1,\ldots,k$.

V FRINGE ANALYSIS FOR RANDOM BINARY TREES

It is possible to apply the methods of fringe analysis for AVL trees to random binary trees (see Overview for an introduction to random binary trees). Failure nodes of leaves of random binary trees will be referred to as leaf failure nodes and failure nodes of interior nodes will be referred to as interior failure nodes. Let $P_n(L,I)$ represent the density of random binary trees (generated with multiplicity) containing exactly L leaf failure nodes and I interior failure nodes. Let L(n) represent the average density of leaf failure nodes over all trees wil n nodes and let I(n) represent the average density of interior failure nodes over all trees with n nodes. When a new insertion lands in a leaf failure node there is a net increase of one interior failure node in the tree; when a new insertion lands in in an interior failure node there is a net loss of one interior failure node and a net increase of two leaf failure nodes. Then I(n) =

$$\sum_{L,I} P_n(L,I)^*I$$

and

$$I(n+1) = \sum_{L,I} P_n(L,I)*((I-1)*I/N)+((I+1)*L/N))$$

$$= I(n) - I(n)/n + L(n)/n$$

Taking the case n = 2 as a basis (where the density of interior failure nodes is 1/3), we can prove via induction that the solution to the above recurrence is 1/3*(n+1).

THEOREM 3.V.1 The average density of interior failure nodes in random binary trees is (1/3)*(n+1) for n > 2.

Note that there is a one-one correspondence between interior failure nodes and unbalanced nodes in random binary trees. This allows us to establish the following

COROLLARY. 3.V.1 The average number of unbalanced nodes in a random binary tree with n nodes is greater than (1/3)*(n+1).

REMARK. It is possible to apply the cumulative fringe analysis of section 3.I and 3.IV to the fringe model for random binary trees and obtain similar results.

The fringe model for random binary trees can be extended to larger classes of subtrees than those described above. Let $\{S_{k_w}\}$ be the set of subtrees of random binary trees split from the tree below the first ancestor rooting a subtree of height greater that some fixed integer k. If no such

ancestor exists an imaginary ancestor of the root with an imaginary subtree of height k opposite the subtree in question will serve to determine the class. We claim that such classes of subtrees are closed under insertion. To observe this note that an insertion into a subtree class member will have one of two of the following effects: the insertion will send a tree of one class into a subtree of another class; or it will split a subtree of a certain class into two subtrees of the original each having its own classification. The latter phenomenon occurs when an insertion lands in a subtree of height k and increases its height to k+1.

Let $N_v(n)$, v= 1, ..., j be the average density of the subtree class $\{T_{k_v}\}$. Then $N_v(n)$ satisfies the following relation

$$N_v(n) = \sum N_v * P_n(N_1, N_2, \ldots, N_j)$$

where $P_n(N_1, N_2, \ldots, N_j)$ is the density of random binary trees of size n with N_1 subtrees of class $\{T_{k_1}\}$, N_2 subtrees of class $\{T_{k_2}\}$ ..., and N_j subtrees of class $\{T_{k_j}\}$. Since we can generate all random binary trees of size n+1 by considering the trees generated by all possible insertions at the failure nodes of binary trees of size n, we can analyze the impact of such insertions on the right side of the above equation.

Suppose $\{T_{k_p}\}$ is a subtree class representative such that an insertion into a tree of size n lands in $\{T_{k_p}\}$. Let c_{pq} count the net change in the number of subtrees of class T_{k_q} , q = 1, ..., j, over insertions landing in any of the possible failure nodes of T_{k_p} weighted by the density of the failure nodes in T_{k_p} which effect the same transition. Then

$$N_v(n+1) = \sum P_n(N_1,N_2,\ldots,N_j)*(N_v+\sum c_{iv}*N_i/(n+1)$$

$$= \sum P_n(N_1,N_2, \ldots, N_j)*N_v +$$

$$\sum P_n(N_1,N_2, \ldots,N_k)*(\sum N_i*c_{iv}/(n+1))$$

$$= N_v n) + \sum c_{iv}*N_i(n)/(n+1)$$

The recursion form suggests that possibly the $N_v(n)$ each equal $d_v(n)$ for some fixed constants $\{d_i\}$ and a computer program would have to be used to come up with a set of canditates. At the least the formula above provides a computational method for computing the $N_v(n)$ which is linear in $n*j$.

WORST CASE ANALYSIS CHAPTER 4

In this chapter we make a number of worst-case analyses of insertion and deletion algorithms on AVL trees and other balanced tree structures. With certain constraints on the class of AVL trees to be considered, we can for instance obtain non-trivial bounds on the number of balance factor adjustments required to generate a tree of the class. For specific permutation sequences such as the identity permutation an exact formula for the number of rotations required to insert this sequence into an AVL tree is obtained. Exact formula for the number of rotations of height k for such sequences are also derived. Finally, we adduce formulae for the number of balance factor adjustments required for insertion sequences and general bounds are given.

I THE IDENTITY PERMUTATION

THEOREM. 4.I.1 The number of rotations required to insert n elements a_1, $a_2, \ldots, a_n$ corresponding to the identity permutation, i.e. $a_1 < a_2 < \ldots <$

α_n, into a height-balanced tree is

$$n - \lceil \log_2 n \rceil - 1, \text{ for } n > 1$$

The result follows form the following three LEMMAs. In the proofs, a leaf L of a height-balanced tree will be referred to as dominant if for each of the trees $T(A_j)$ rooted by ancestors $\{A_j\}$ of depth j , j > 1, of L, lies in a subtree of $T(A_j)$ of height j - 1.

LEMMA 4.I.1 For the input sequence $\alpha_1, \ldots, \alpha_n$ corresponding to the identity permutation, the node containing α_n is always a leaf.

Proof: If the insertion of the node of α_n requires no rebalancing, then there is nothing to prove. If a rotation occurs, we may conclude from an examination of the rotation diagrams on pp. 10-12 that after a rebalancing, the leaf containing α_n remains a leaf.

LEMMA 4.I.2 Given the input sequence $\alpha_1, \alpha_2, \ldots, \alpha_n$ corresponding to the identity permutation, the leaf of α_n is always dominant.

Proof: by induction for n > 2.

Basis step: we begin with a tree consisting of a single node containing α_1.

When α_2 is inserted to the right, we note that it lies in the taller of the subtrees of the root i.e. it is dominant.

Induction step: for $n > 3$ we consider the effect of the insertion of α_n in the cases i) where no rotation occurs, and ii) where a rotation does occur. We know by LEMMA 4.I.1 that α_n is a leaf. Now assume by the induction hypothesis that α_n is dominant. If the insertion of α_{n+1} requires no rebalancing and α_n is dominant then α_n must initiate a zero path.(i.e. before the insertion of α_{n+1} all the ancestors of the node containing α_n had balance factors identically zero). Then α_{n+1} must lie in the taller of the subtrees of all trees rooted by its ancestors. Hence the dominance property extends to identity sequences of size n+1 where the insertion of α_{n+1} corresponds to case i). For case ii) we note that if a rotation occurs it must always be of type RR (see the rotation diagrams on pp. 10-12 for a description of RR rotations). This must be the case since α_{n+1} is inserted into the right subtrees of all trees rooted by its ancestors. If we observe the diagram for the RR rotation, we note that by the induction hypothesis the insertion of α_{n+1} makes its leaf dominant in B_R. After the rotation the new root B of α_{n+1} has balance factor zero. Since the remainder of the tree is not disturbed by the rotation, we conclude that the leaf of α_{n+1} is dominant with respect to the trees rooted by all its ancestors.

LEMMA 4.I.3 Given the height-balanced tree constructed by the identity permutation sequence for $n > 3$, the set of all left subtrees of nodes along the

path from the leaf containing α_{n+1} to the root are conplete binary trees.

Proof: by induction for $n > 3$.

Basis: by observation for $n = 3$.

Induction step: Assume the LEMMA true for some $n > 3$. If we insert $_{n+1}$ and no rotation occurs, then certainly the property is preserved. Suppose now a rotation occurs after the insertion of α_{n+1}. Here we follow the notation for the RR diagram on pp. 10-12. By the induction hypothesis, the subtree B_L of the right subtree of A is a complete binary tree. Within B_L the left subtrees of subtrees containing α_{n+1} remain complete after its insertion. We note also that by the induction hypothesis, the subtrees A_L and B_L are complete and hence after the rotation the tree rooted by A with subtrees A_L and B_L is complete as well. Finally, we observe that the rotation leaves the rest of the tree undisturbed and hence the left subtrees of all trees rooted by ancestors of α_{n+1} remain complete.

Then from LEMMA 4.I.2, we conclude that a rotation can fail to take place if and only if the leaf of α_{n+1} initiates zero path. But if this is the case, by LEMMA 4.I.3, all the left subtrees of this tree must be complete and hence the entire tree is complete. A binary tree can only be complete for $n = 2^k - 1$, $k > 1$, and by a simple application of LEMMA 4.I.3, we can show that the identity permutation sequence of size n constructs all complete binary trees of height $< \lceil \log_2 n \rceil - 1$. Hence the formula $n - \lceil \log_2 n \rceil - 1$.

We can characterize rotations by the distance from the newly inserted node to its first ancestor with a non-zero balance factor. This distance will be referred to of the height of the rotation. Now we may adapt the machinery of the previous LEMMA's and THEOREM 4.I.1 to give explicit formulae for those $i < n$ in the identity permutation sequence at which a rotation of height $k < \log_2 n - 1$ occurs.

THEOREM 4.I.2 For the sequences

$$n_{k_1} = 2^{k-1} \qquad k > 1$$

$$n_{k_j} = n_{k_{j-1}} + 2^k \qquad j \geq 1$$

and $n_{k_j} < n$, the insertion of the n_{k_j}'th element of the identity permutation sequence causes a rotation of height k+1.

Proof: by induction on $k > 1$, $j > 1$.

Basis step: k=1, j=1. We observe the RR-rotation diagrams for n = 3 to see that a rotation of height 2 occurs.

Induction step: $k = 1$, $j > 1$. We must show that rotations of height 2 occur precisely at the odd integers > 3. Suppose $j > 3$ is odd. Then j-1 is even and

by the induction hypothesis either i) a rotation of height > 2 occurred at the j-1'st insertion or ii) no rotation occurred at all. Situation ii) could only have occurred, by THEOREM 4.I.1, if the j-1'st insertion were made into the complete binary tree leaving the setup of Figure 4.I.1

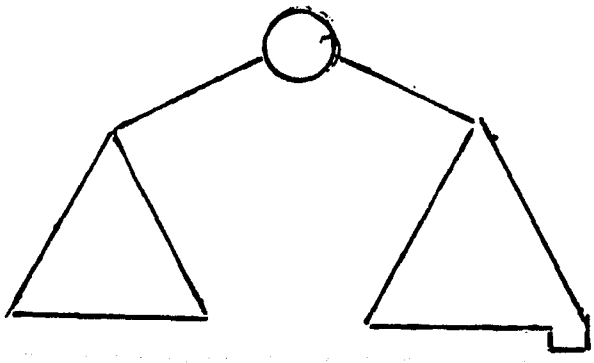


Figure 4.I.1

Then the rightmost leaf has no left sibling in this situation and the insertion of the j'th element to the far right necessitates a rotation of height 2. For case i) we note by an exmination of the RR-rotation diagram that the rightmost leaf in subtree B has again no left sibling and hence the insertion of a new leaf to the right of it will cause a rotation of height 2.

Basis step: k>1, j=1. We must prove that for k > 1 that at the n_{k_1}'st $(2^k + 2^{k-1})$ insertion the tree in question has the shape

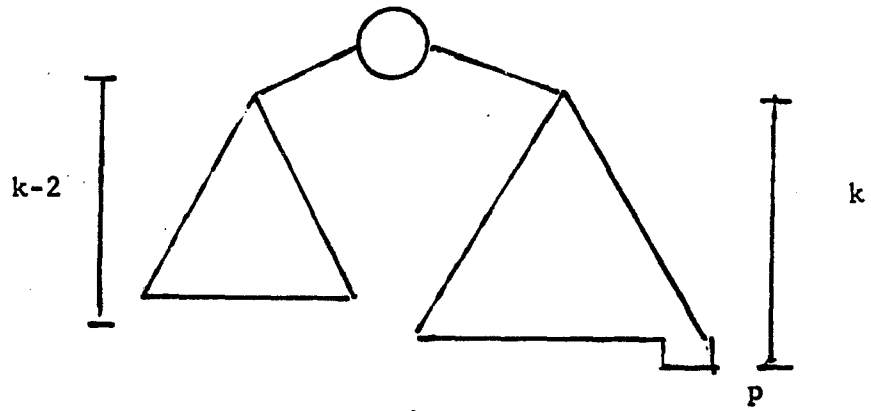


Figure 4.I.2

where E is complete of height k-2, F less P is complete of height k-1 and the insertion of P causes a rotation of height k+1. By LEMMA 4.I.3 E is complete. It must be of height at least k-2 as otherwise F would contain too many nodes to maintain balance at G. Then E must contain either 2^k-1 nodes or $2^{k-1}-1$ nodes. Suppose E contains $2^{k-1} - 1$ nodes. By LEMMA 4.I.3 the subtrees opposite the parent P are complete and number including the parent of P

$$\sum_{i=0}^{q-2} 2^i = 2^{q-1} - 1$$

Then since F must have $2^{k-1} - 1$ nodes q must equal k-2. But by LEMMA 4.I.2 the parent of P had to be dominant in the tree after its insertion and clearly if the height of F were k-2 this could not be the case. Hence E must be complete of height k-2, F minus P must be complete of height k-1 and the insertion of P causes a rotation of height k+1.

Proof: by induction j>1, k>1,. Suppose for some j > 1 a rotation of height k>1 occurs in the manner described in the previous step j=1. The after the insertion the rightmost subtree of height k has the shape

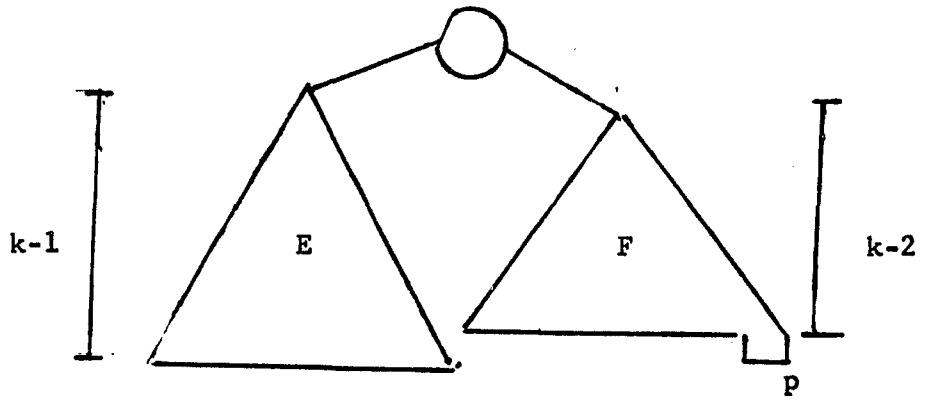


Figure 4.I.3

where E is complete of height k-1 and F minus p is complete of height k-2 and F including P is of height k-1. A simple proof by induction on k shows that after $2^{k-1} - 1$ additional insertions the subtree has the shape:

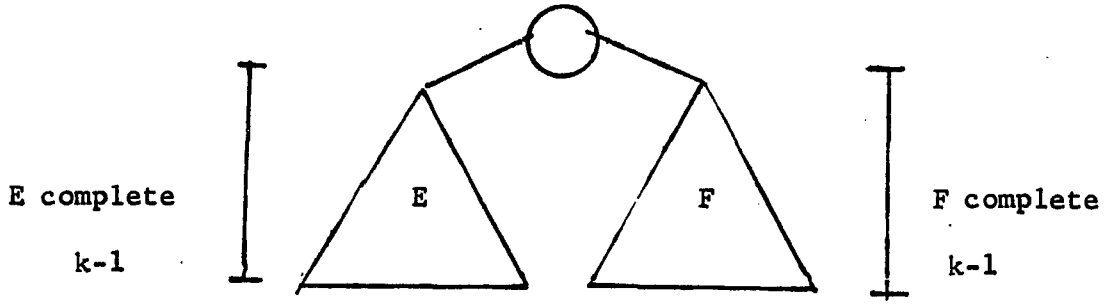


Figure 4.I.4

Whether or not the next insertion causes a rotation up the tree beyond H or not, F, the rightmost subtree of height k, will have the shape:

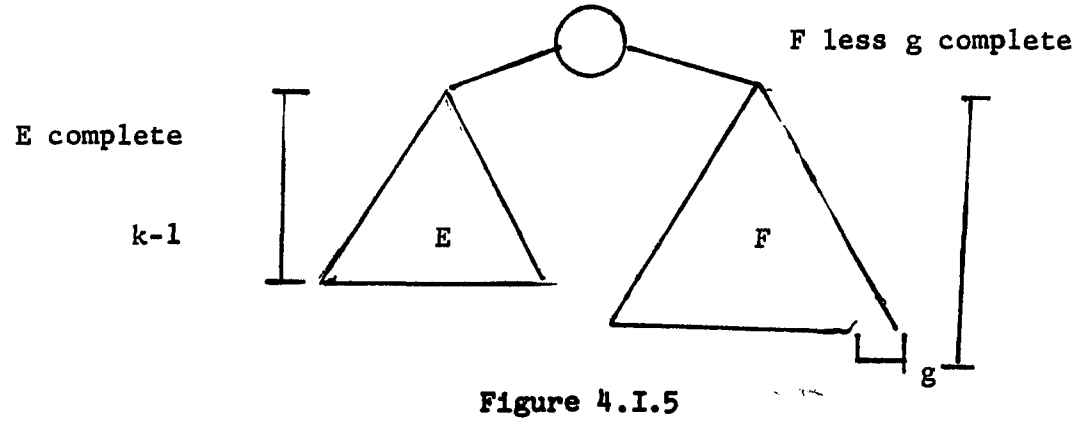


Figure 4.I.5

After $2^{k-1} - 1$ additional insertions F_2 becomes complete of height k-1 and we have the following setup:

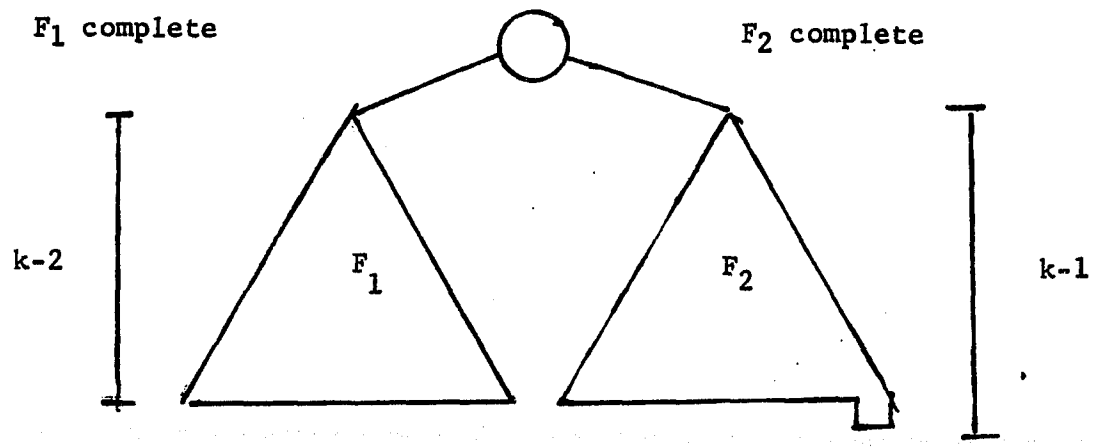


Figure 4.I.6

The next insertion, the $n_{k_{j+1}}$'st, will cause a rotation of height k+1.

We note that whenever a rotation occurs the balance factors along the path from the newly inserted node to its first ancestor which, before insertion, held a non-zero balance factor. In addition a maximum of three additional balance factor adjustments may occur around that ancestor during the rotation process (see the rotation diagrams on pp.10-12). The result just proved allows us to obtain an explicit formula for the number of balance factor adjustments required to insert the identity permutation into a height-balanced tree. The formula is linear in the size of the sequence.

THEOREM 4.I.3 The number of balance factor adjustments required to insert n elements corresponding to the identity permutation into a height-balanced tree is

$$\sum_{k=2}^{\lceil \log_2(n) \rceil} \frac{n*(k-1)}{2^k} - (k-1) \quad + \quad \sum_{i=0}^{\lceil \log_2(n) \rceil} i+1$$

$$+ n - \lceil \log_2(n) \rceil - 1$$

$<c*n$ for some fixed effectively computable constant c independent of n

Proof: we account for each term $((n/2^k)*(k-1) - (k-1)$ from the formulae for the sequences n_{k_j} at which a rotation of height k+1 occurs . We account for the second summation by noting that at indices 1,2, ..., 2^j no rotation occurs but nevertheless j balance factor adjustments are made. If we follow the notation of the diagram for the RR-rotation on p.12, we may account for the last term by noting that every time a rotation occurs node A is shifted down the tree and its balance factor changes. To get the linear inequality we observe that the second sum is $\lceil \log_2(n) \rceil 2 + \lceil \log_2(n)/2 \rceil$ and that

$$\sum_{k=2}^{\log_2(n)} \frac{n*(k-1)}{2^k} - (k-1)$$

$$> n * \left(\sum_{k=2}^{\log_2(n)} 2^{\frac{k-1}{k}} - (k-1) \right)$$

$$> n * \left(\sum_{k=2}^{\infty} 2^{\frac{k-1}{k}} - (k-1) \right)$$

THEOREM 4.I.1 demonstrates the effectiveness of the following upper bound on the number of rotations which can occur in the generation of a height-balanced tree with n nodes.

THEOREM 4.I.4 A maximum of $n - \log_2(n) - 1$ rotations can occur in the generationof a height-balanced tree with n nodes.

Proof: Inspection of the rotation diagrams on pp. 10-12 shows that the height of the tree cannot increase when an insertion causes a rotation. But at least $\log_2(n) + 1$ insertions must increase the height of the tree as its final height must be at least $\log_2(n)$.

There are a number of corollaries to THEOREMs 4.I.1 and 4.I.2, a few of which we present here.

COROLLARY 4.I.1 For n even, the sequence 2,1,4,3, ..., n,n-1 requires exactly $n/2 - \lceil \log_2(n) \rceil$ rotations to insert it into a height-balanced tree.

Proof: The proof follows exactly as in the proof THEOREM 4.I.1 if we note that odd-indexed sequence entries are always inserted into the shorter subtree of an M-subtree (see Chapter 2 for a definition of an M-subtree) and hence do not cause rotations of height 2 as in the case of the identity permuation. Otherwise the rotation behavior of the two sequences is identical.

COROLLARY 4.I.2 There is a positive effectively computable constant c such that the total number of balance factor adjustments required to insert the sequence 2,1,4,3, ..., n,n-1, n even, into a height-balanced tree is less that $c*n$ as $n \to \infty$.

Proof: As noted above the balance factor adjustments for this sequence are identical to those of the sequence 1,2, ..., n except at the odd integers, where instead of causing a rotation of height 2, the insertion lands in the shorter subtree of an M-subtree. But in both cases exactly one balance factor adjustment is made so the bound in THEOREM 4.I.3 applies to the sequence 2,1,3,4,...,n,n-1 as well.

COROLLARY.4.I.3 For n odd the sequence n/2 + 1, n, 2, n-1, ..., n/2 - 1, n/2 + 2, requires exactly $n - 2*\lceil \log_2(n/2) \rceil - 3$ rotations to insert it into a height-

balanced tree.

Proof: A simple proof by induction shows that a member of the subsequence 1,2,3, ..., n/2 -1 can never migrate to the right subtree of the tree and no member of the subsequence n,n-1, ..., n/2 + 2 can migrate to the left subtree of the tree. Then the insertion process for these two subsequences goes on independently as if the two sequences were being inserted into two disjoint trees. By THEOREM 4.I.1 each of the subsequences requires $n/2 - \lceil \log_2(n/2) \rceil - 1$ rotations to insert it into its respective subtree. Finally, we observe that no rotation is required to insert the first element into the tree.

To continue this worst case analysis of insertion algorithms, consider the following problem: over all permutations of the input sequence, determine bounds for the maximal number of rotations around the root of the tree required to insert n elements into an AVL tree.

THEOREM 4.I.5 Let T be an AVL tree of height k. Then at most k root rotations could have occurred in the generation of T.

Proof: A rotation around the root of a tree will occur if and only if both of the following conditions are met: i) the balance factor of the root is non-zero; and ii) the insertion is made at a failure node initiating a zero path in the taller of the subtrees of the root. Inspection of the rotation diagrams

on pp. 10-12 reveals that after a rotation at the root the balance factor of the new root is zero. This implies that for condition i) for a root rotation to obtain, the height of one of the subtrees of the root must increase by one. If a tree has a balance factor of zero in the root and the height of one of the subtrees increases by one, then the height of the tree must increase by one as well. This follows by a simple proof by induction. To complete the proof we note that under insertion the height of a tree is monotone increasing at most one root rotation can be charged to each of the height-increases obtained in the generation of the tree.

COROLLARY 4.I.4 A node of a height-balanced tree generated by insertions from an empty tree can migrate during this process from one subtree of the root to another a maximum of k times where k is the height of T.

Proof: A node can move from one subtree of the root to another only if a rotation occurs at the root.

There is analogy to THEOREM 4.IV.4 for sequences of deletions from an initial height-balanced tree T. We adopt the following terminology: the deletion of a node which causes a rotation around the root of the tree will be referred to as a root deletion rotation. For a complete discussion of deletion algorithms on height-balanced trees see [1] .

THEOREM 4.I.6 Let T be an AVL tree of height k. Then a sequence of deletions

of nodes from T giving rise to the empty tree can cause at most k root deletion rotations.

Proof: An inspection of the deletion rotation diagrams shows that after a deletion at the root of the tree the height of the tree must decrease by one.

II REBALANCINGS IN CLASSES OF AVL TREES

There are interesting analogies to THEOREM 4.I.3 for measuring the number of balance factor adjustments incurred in insertion algorithms on AVL trees. Again we must apply certain restrictions on the types of AVL trees to which the results apply.

THEOREM 4.II.1 Let T be a height-balanced tree with n nodes of height $\lceil \log_2(n) \rceil + k$ in which j, fixed independent of n, rotations have occurred. Then there exists a constant c which depends only on j and k such that the total number of balance factor adjustments required to generate T is less than $c*n$.

Proof: Suppose Y is inserted at a leaf of the tree. Following the rotation diagrams on pp.10-12, we let A be the first ancestor of the newly inserted node with a non-zero balance factor. Let B be the root of the subtree of A containing the node N(Y) and let T(B) be the subtree rooted by B. Let C be the

root of the subtree in T(B) containing N(∫) and let C_R and C_L be the left and right subtrees of T(C), the subtree rooted by C. We adopt a charging scheme which relates the final level of a node to the number of balance factor adjustments charged against it. If no rotation occurs the balance factors of each node along the path from N(∫) to A must be adjusted and accordingly the depth of each node along the path increases by one with the exception of A. To account for the adjustment of A without the corresponding depth increase 2 charges are made to B (in the case where A is the root, its depth in the tree increases also and it is not necessary to make an addditional charge to B). In the case of an RR or an LL rotation the depth of all nodes along the path from N(∫) to B increase in depth and one balance factor adjustment is charged to each of them. The balance factor of A also changes so an additional charge is made to B. In the case of an RL or an LR rotation the balance factors in C_R and C_L change and for each change the depth of the node position charged increases by one in the tree. A decreases in depth and so a possible charge related to the previous depth of A might be lost. To account for this as well as the charge for changing the balance factor of A in the rotation three balance factor charges are made to C whose depth increases by 2.

Now consider in the resultant tree T a node of level m+k. For such a node at least m+k balanced factor adjustments must be charged to account for its height in the tree. Also, if such a node were a node of type B in a non-rotational insertion, or in an RR or LL rotation or in the case it was a node of type C in an LR or RL rotation as many as 2*(m+k) charges could be laid to it. Furthermore, if during a rotation, the node had been shifted down the tree, certain charges to the node may not be reflected by its depth. However, since

only j rotations occurred at most 2*j charges to a node would not be reflected in its depth. Since there are at most $n/2^m$ nodes at level m+k, the total charges which can be laid to this level is less than

$$\frac{n*2*(k+j)}{2^m}$$

Summing over all levels greater than k we obtain

$$n*\left(\sum_{p=k}^{\lceil \log_2(n) \rceil + k} \frac{2*(j+p+k)}{2^{p-k}}\right)$$

$$< n*\left(\sum_{p=k}^{\infty} \frac{2*(j+p+k)}{2^{p-k}}\right)$$

$$< c_1*n$$

Now a maximum of $2*(k+j)*n < c_2*n$ charges can be made against nodes of level less than k. Hence the total number of balance factor adjustments required to generate the tree is less than

$$c_1*n + c_2*n$$

$$< c*n$$

THE GENERAL CASE FOR BALANCE FACTOR ADJUSTMENT COSTS III

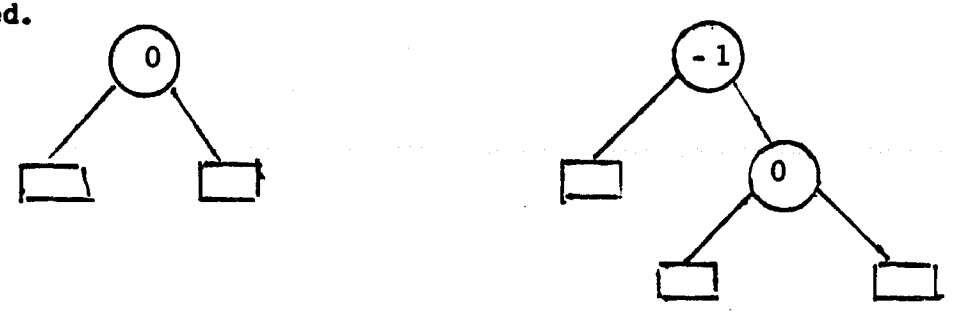
There is a method to obtain bounds for the maximal and average number of balance factor adjustments for an arbitrary permutation sequence generating an AVL with n nodes.

THEOREM 4.III.1 The number of balance factor adjustments required to insert n elements into an initially empty AVL tree T is less than $2*n + U(T)$, where U(T) is the number of unbalanced nodes in T.

Proof: It must be shown that for an arbitrary insertion into an AVL tree the total number of balance factor adjustments is less than the net change in the number of nodes in the tree with non-zero balance factors + 2. An insertion which lands into an AVL tree falls into one of three categories: 1) all of its ancestors have zero balance factors; 2) it lies in the shorter of the subtrees

of its first ancestor with a non-zero balance factor; 3) it lies in the taller of the subtrees of its first ancestor with a non-zero balance factor. Note a rotation occurs in the third case and no rotation occurs in the other two.

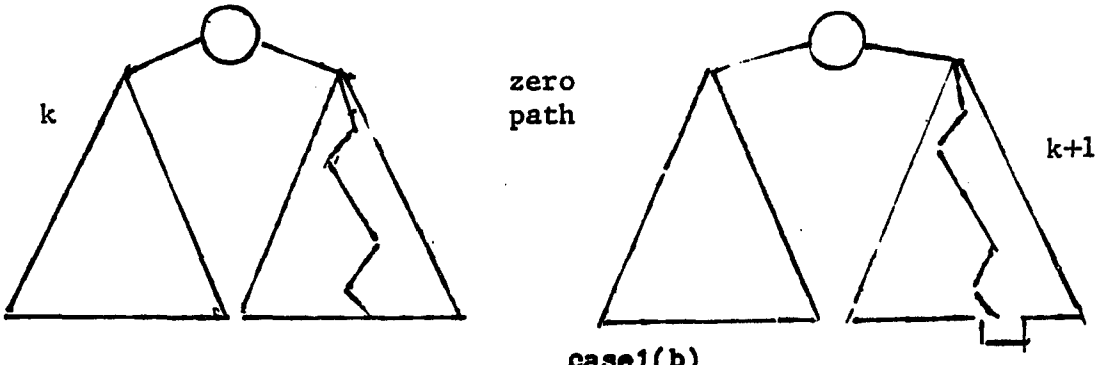
Now consider the number of balance factor adjustments in the the first case and the net change in the number of non-zero balance factors. Note we assume that the balance factor of the newly inserted node is pre-set to zero and so its balance factor need not be considered among those balance factors adjusted.



case 1(a)

Figure 4.III.1

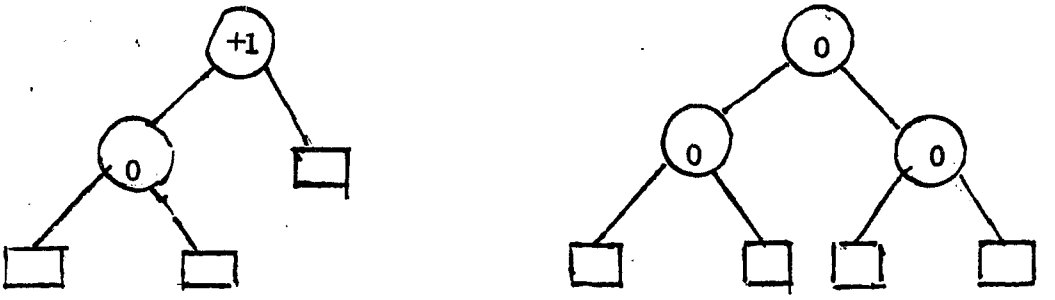
In case 1(a) where the tree consists of a single node the net change in non-zero balance factors is +1 and +1 balance factor adjustments made so the difference is less than +2 consistent with the statement of the theorem. Now consider case 1 for general n.



case1(b)

Figure 4.III.2

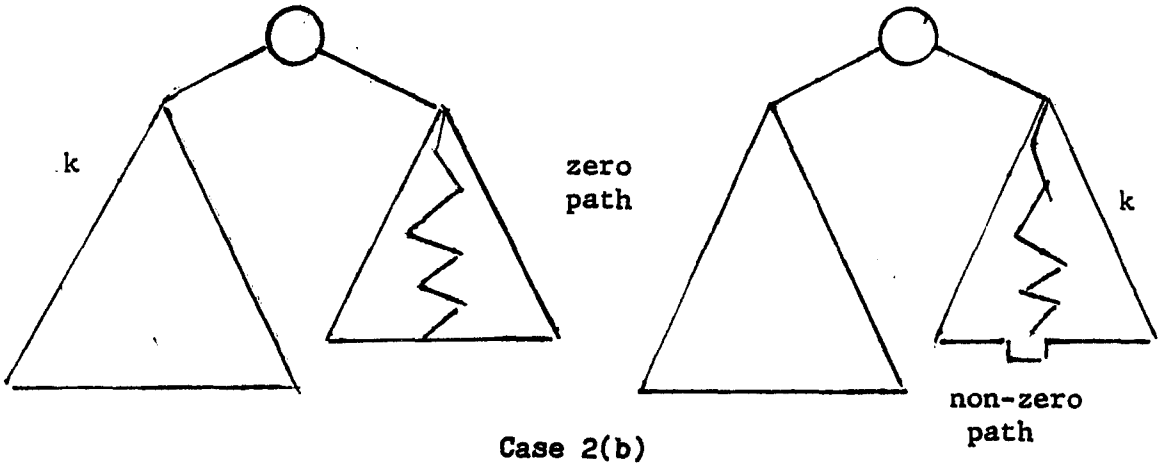
Here the net change in nodes with non-zero balance factors equals the total number of balance factor adjustments. Now consider case 2.



Case 2(a)

Figure 4.III.3

Here again the number of balance factor adjustments equals the net change in the number of non-zero balance factors plus two. Here is the general case 2.



Case 2(b)

Figure 4.III.4

To observe the correctness of the theorem in case 3 the reader is referred to the rotation diagrams on pp.10-12.

Let d(U(i)) denote the net change in unbalanced nodes after the i'th insertion and let BC(i) equal the number of balance factor adjustments required for the i'th insertion. Then

$$\sum d(U(i)) + 2 \quad > \sum BC(i)$$

But $\sum d(U(i))$ equals U(T), the number of nodes with non-zero balance factors in the final tree. This proves the theorem. Naturally U(T) is less than n so the total number of balance factor adjustments must be less than 3*n. There is an interesting corollary to THEOREM 4.III.1.

COROLLARY 4.II.1 The expected height of an insertion into any AVL tree with n nodes is O(1) as $n \to \infty$. The height of an insertion denotes the length of the path from the newly inserted node to its first ancestor with a non-zero balance factor or to the root of the tree if the balance factors of all the node's ancestors are identically zero.

Proof: From the discussion above the height of an insertion is less than the number of balance factor adjustments required to make the insertion . Hence the average height of an insertion is less than 3*n/n which is less than 3.

The analogue for B-trees to balance factor adjustments in AVL trees is the number of index comparisons and node splittings required to insert n

elements into a B-tree of order m. Recall the description of the insertion algorithm for B-trees from page 24. If the leaf node L is made full by the insertion of the key J an attempt is made to move the newly inserted key up to its parent node P and to move the adjacent key to the left of J down to the left sibling of L; or if this node if full a symmetric attempt is made to move the right adjacent key of J to the root of the right sibling of L . If these siblings are both full P splits and its middle key M is moved up to the parent node P' of P. Once again, if the insertion of M causes P' to be full an attempt is made to displace keys adjacent to M in P' in corresponding sibling nodes. This splitting and reassignment of nodes may carry to the root in which case it splits creating a new root for the tree. We infer from this discussion each time a key is inserted and causes a chain of splittings the same processing occurs at each step of the chain. Each node access will be accorded a processing unit. At the same time for each splitting there is a corresponding increase of the total depth of the keys of the tree. Suppose an insertion causes k splittings but the final operation is to move a key from a full node successfully to a sibling. Then there is one procesing unit for the leaf access, three accesses and hence three processing units for each node that splits (three k in all). Finally, there is one processing unit for the final insertion. There is a net increase in the total index key depth of the tree by k+2. Then the net processing units are three times more than the net increase of the total depth of tree. When a node is inserted successfully into a leaf there is a net increase of one in the total key depth of the tree and no rebalancing processing. When a node is inserted into a full leaf and a key in the parent node is displaced in a sibling there are two or three processing

units and a net gain of one in total key depth. Then the number of node processing units is less than

$$3*(\text{Total key depth})$$

This leads us to the following result

THEOREM 4.III.2 The total rebalancing processing required to insert n elements into a B-tree of order m is O(n) where the implied constant depends only on the fixed time c required to insert a key into a node, possibly displace other keys in the node, or possibly to split the node, and m the order of the tree.

Proof: We want to show that the total depth of a B-tree with n nodes is O(n). For simplicity assume m is odd and let $r = m/2 + 1$. Assume n has the form $1 + r + r^2 + \ldots + r^k$. A simple proof by induction shows that for such n the total depth of a B-tree is maximized by choosing a tree with one key in the root and exactly r nodes in every other node. Then the total depth of such a tree is

$$\log_r n + (\log_r(n-1)) * r + (\log_r(n-2)) * r^2 + \ldots + r^{\log_r n}$$

This expression is less than

$$\sum_{k=1}^{\infty} \frac{n*k}{r^k}$$

$$< n\left(\sum_{k} \frac{k}{r^k}\right)$$

$$< c*n$$

Results similar to THEOREM 4.III.1 exist [16] but readjustment costs are not related to the number of non-zero balance factors in the resulting tree but rather to the total path length of the tree. Results analogous to THEOREM 4.III.2 also exist in the literature [16].

COMBINATORIAL AND STATISTICAL RESULTS CHAPTER 5

INTRODUCTION

Very often counting aruguments may be applied to recursively defined objects such as height-balanced trees through the use of recursive functions. Note, for instance, that the counting function for the number of height-balanced trees of a given height k is $F(k) = F(k-1)^2 + 2*F(k-1)*F(k-2)$. In this chapter we exploit this relation on a number of asymptotic density limits relating to the properties of height-balanced trees. In particular we establish upper and lower bounds for the following limit problem:

Let T(k) count the number of failure nodes of all height-balanced trees of a given height $k > 0$. A failure node at which an insertion would imbalance the tree will be referred to as a rotational failure node. Now let G(k) enumerate the number of rotational failure nodes in the set enumerated by T(k). We wish to determine the limit G(k)/T(k) as $k \to \infty$. In what follows we establish a lower bound for this limit if it exists.

THEOREM 5.I.1 The ratio $G(k)/T(k) > .70697$ for $k > 9$.

REMARK The ratio we derive cannot automatically be taken as probability estimates of the number of rotations required to insert n elements into a

height-balanced tree. In fact, they are probably quite different. In the node parametrized case (which is Brown's model; see Chapter 2) the probabilities are computed over all failure nodes of all AVL trees generated with multiplicity from all permutations of an ordered set of n elements. In the present case the trees and consequently the failure nodes are not taken with multiplicity. Note that many permutations of the input set may lead to the same tree and in fact empirical evidence suggests (see [13], pp.460-1) that the trees generated in the greatest number are the most balanced. For example, when n = 7 there are 7! possible permutations of the input sequence and 2140 of them lead to the complete binary tree of height 3, none of whose failure nodes are rotational.

In addition to this result on insertions we establish the following analogous result for deletions.

THEOREM 5.II.1 The probability that the deletion of an arbitrary node of an arbitrary height-balanced tree of height k will cause an imbalance of the tree is greater than .11417 for k > 9.

REMARK. This result is of particular interest since so little is known about the statistical aspects of deletion algorithms for height-balanced trees.

I PROOF OF THEOREM 5.I.1

Now we establish the lower bound. Let G(h) be the number of failure nodes of all extended height-balanced trees of height h at which the insertion of a new node would imbalance the tree. Let H(h) enumerate failure nodes which initiate zero paths in the set of all extended height-balanced trees of height h. Then we have the following product development.

$$H(k) = \prod_{j=1}^{k-1} 2*F(j)$$

Now we can establish a formula for G(k)

$$G(1) = 0$$

$$G(2) = 4$$

$$G(k) = 2*G(k-1)*(F(k-1) + F(k-2)) + 2*G(k-2)*F(k-1) + 2*H(k-1)*F(k-2)$$

This formula follows from the following observations. If we embed a rotational failure node in a subtree of height k-1 in a tree of height k by taking it as the left subtree, then this failure node will again be rotational in the embedding tree. The factor F(k-1) + F(k-2) in the first expression enumerates the choices for the right subtree under this embedding. A similar analysis holds when the rotational failure node is embedded in the right subtree. The term 2*G(k-2)*F(k-1) counts the number of ways a rotational failure node in a tree of height k-2 may be embedded in a tree of height k. To see the last term note that a failure node will cause an imbalance of the tree if it lies in the taller of the left or right subtrees of the subtree rooted by

the first ancestor of the newly inserted node with a non-zero balance factor. This last term counts the case when the ancestor is the root of the tree itself. The failure nodes which initiate zero paths in the left or right subtrees of height k-1 in trees of height k with a non-zero balance factor in the root will also cause an imbalance.

Let T(k) be the total number of failure nodes of all extended AVL trees of height k. Then

$$T(1) = 2$$

$$T(2) = 10$$

$$T(k) = 2*T(k-1)*(F(k-1) + F(k-2)) + 2*T(k-2)*F(k-1)$$

The derivation of this formula is similar to those above.

Then the probability that an insertion at an arbitrary failure node of an arbitrary AVL tree of height k will imbalance the tree is equal to the ratio G(k)/T(k)

$$G(k) = \frac{G(k-1)}{T(k-1)} * \frac{G(k)*T(k-1)}{T(k)*G(k-1)}$$

Now we may rewrite G(k) as

$$G(k-1)*\left(2*F(k-1) + F(k-2) + \frac{2*G(k-2)*F(k-1)}{G(k-1)} + \frac{H(k-1)*F(k-2)}{G(k-1)}\right)$$

Similarly, we may rewrite T(k) as

$$T(k-1)\left((2*F(k-1) + F(k-2)) + \frac{2*T(k-2)*F(k-1)}{T(k-1)}\right)$$

Then

$$\frac{G(k)}{T(k)} = \frac{G(k-1)}{T(k-1)} * \frac{2*F(k-1)+F(k-2)) + \dfrac{G(k-1)}{2G(k-2)*F(k-1)+F(k-2)*H(k-1)}}{2*(F(k-1) + F(k-2)) + \dfrac{2*T(k-1)*F(k-1)}{T(k-1)}}$$

$$> \frac{G(k-1)}{T(k-1)} * \frac{2*(F(k-1) + F(k-2))}{2*(F(k-1) + F(k-2)) + \dfrac{2*T(k-2)*F(k-1)}{T(k-1)}}$$

$$= \frac{G(k-1)}{T(k-1)} * \frac{1}{1 + \dfrac{2*T(k-2)*F(k-1)}{T(k-1)*2*(F(k-1) + F(k-2))}}$$

$$> \frac{G(k-1)}{T(k-1)} * \frac{1}{1 + \dfrac{T(k-2)}{T(k-1)}}$$

$$> \frac{G(k-1)}{T(k-1)} * \prod_{j=k} \frac{1}{1 + \frac{T(j-2)}{T(j-1)}}$$

It is easy to observe that

$$T(j-2) < 2^{j-2} * F(j-2) \quad \text{and}$$

$$T(j-1) > F(j-1)$$

Hence

$$\frac{G(k)}{T(k)} > \frac{G(k-1)}{T(k-1)} * \prod_{j=k} \frac{1}{1 + \frac{2^{j-2} * F(j-2)}{F(j-1)}}$$

By the relation $F(j) = F(j-1)^2 + 2*F(j-1)*F(j-2)$, each of the factors

$$\frac{1}{1 + \frac{2^{j-2} * F(j-2)}{F(j-1)}} > \frac{1}{1 + \frac{2^{j-2}}{F(j-2)}}$$

Let γ equal $\sqrt[4]{3}$. Then a simple proof by induction shows that $F(k) > \gamma^{2^k}$ for $k > 1$. This implies that our product development is greater than

$$\prod_{j=k} \left(1 + \frac{2^{j-2}}{2^{j-2}}\right)$$

Then for any $q > k$

$$\frac{G(q)}{T(q)} > \frac{G(k-1)}{T(k-1)} * \prod_{j=k} \left(1 + \frac{2^{j-2}}{2^{j-2}}\right)$$

The bound in the result was derived from computing G(9)/T(9). It is obvious that for this value of k the effect of the infinite product on the bound is negligible.

II PROOF OF THEOREM 5.II.1

The proof follows from two lemmas: the first which establishes that a positive density of the leaves of AVL trees of height k will imbalance the tree if deleted; and the second which shows that the leaves of AVL trees of height have a positive density in the set of all nodes of height balanced trees of height k. We will refer to a node as deletion rotational if its deletion would require a rebalancing of the tree.

LEMMA 5.II.1 The density of rotational leaves in the set of all leaves of AVL trees is positive and effectively computable from the analysis described below.

Proof: Let R(k) be the number of leaves of all height-balanced tree of height k at which a deletion would imblance the tree. Let B(k) enumerate leaves of all height-balanced trees of height k at which a deletion would reduce the height of the tree from k to k-1. Finally, let T(k) enumerate all leave of height-balanced trees of height k. Again, let F(k) enumerate distinct AVL trees of height k. Then we have the following formula.

$$B(1) = 1$$

$$B(2) = 4$$

$$B(k) = \prod_{j=1}^{k-2} 2*F(j)$$

To observe the derivation of this formula note that $B(k) = 2*B(k-1)*F(k-2)$, $k > 3$, since a node whose deletion will shorten a subtree of the root of a height-balanced tree of height k will shorten the entire tree by its deletion only if the height of the subtree opposite it is of height k-2.

$$R(1) = 0$$

$$R(2) = 0$$

$$R(k) = 2*R(k-1)*(F(k-1) + F(k-2)) + 2*R(k-2)*F(k-1) + 2*B(k-2)*F(k-1)$$

This formula follows from the following observations. For the term $2*R(k-1)*(F(k-1) + F(k-2))$ note that a leaf whose deletion will imbalance a subtree of height k-1 in a tree of height k will imbalance the entire tree as well. The factor $2*(F(k-1) + F(k-2))$ enumerates the ways such a leaf in a subtree of height k-1 may be embedded in a height-balanced tree of height k. A similar analysis applied toward a leaf whose deletion will imbalance a subtree of

height k-2 in a height-balanced tree of height k acccounts for the term 2*R(k-2)*F(k-1). To see the last term note that an imbalance will occur at the root of a height-balanced tree of height k only if the leaf deleted shortens a subtree of the root of height k-2. Again we have the following formula for T(k)

T(1) = 2

T(2) = 10

T(k) = 2*T(k-1)*(F(k-1) + F(k-2)) + 2*T(k-2)*F(k-1)

which first appeared in the proof of the lower bound for THEOREM 5.I.1 Then the probability that the deletion of a leaf will imbalance a tree of height k is R(k)/T(k) and

$$\frac{R(k)}{T(k)} > \frac{R(k-1)}{T(k-1)} \quad \frac{R(k)*T(k-1)}{T(k)*R(k-1)}$$

Now we rewrite R(k) as

$$R(k-1)*2((F(k-1) + F(k-2)) + \frac{2*R(k-2)*F(k-1)}{R(k-1)}$$

$$+ \frac{2*B(k-2)*F(k-1)}{R(k-1)} \quad)$$

and we may rewrite T(k) as

$$T(k-1)*(2*(F(k-1)+F(k-2)) + \frac{2*T(k-2)*F(k-1)}{T(k-1)})$$

Then

$$\frac{R(k)}{T(k)} = \frac{R(k-1)}{T(k-1)} * \frac{2*(F(k-1)+F(k-2)) + \frac{2*R(k-2)*F(k-2)}{R(k-1)} + \frac{F(k-1)*B(k-2)}{R(k-1)}}{2*(F(k-1)+F(k-2)) + \frac{2*T(k-2)*F(k-1)}{T(k-1)}}$$

$$> \frac{R(k-1)}{T(k-1)} * \frac{1}{1 + \frac{2*T(k-2)*F(k-1)}{T(k-1)*2*(F(k-1)+F(k-2))}}$$

$$> \frac{R(k-1)}{T(k-1)} * \frac{1}{1 + \frac{T(k-2)}{T(k-1)}}$$

Then for any $q > k$ we have the ratio

$$\frac{R(q)}{T(q)} > \frac{R(k-1)}{T(k-1)} \prod_{j=k}^{\infty} \frac{1}{1 + \frac{T(j-2)}{T(j-1)}}$$

Again we observe that $T(j-2) < 2^{j-2}*F(j-2)$ and $T(j-1) > F(j-1)$

Hence

$$\frac{R(q)}{T(q)} > \frac{R(k-1)}{R(k-1)} * \frac{1}{1 + \frac{2^{j-2}*F(j-2)}{F(j-1)}}$$

By the relation $F(j) = F(j-1)^2 + 2*F(j-1)*F(j-2)$ each of the factors

$$\frac{1}{1 + \frac{2^{j-2}F(j-2)}{F(j-1)}} > \frac{1}{1 + \frac{2^{j-2}}{F(j-1)}}$$

Note that in deletion rotation analysis extended height-balanced trees are not considered. Hence $F(1) = 3$ and we may argue as in the proof of THEOREM 5.1 that for $\gamma = \sqrt{3}$ and for $k > 1$, $F(k) > \gamma^{2^k}$. This implies that the product development is greater than

$$\prod_{j=k}^{\infty} \frac{1}{1 + \frac{2^{j-2}}{\gamma^{2^{j-2}}}}$$

Again this product converges to a strictly positive limit.

LEMMA 5.II.2 There exists a positive effectively computable constant such that the density of leaves of height-balanced tree of height k in the set of all nodes of height-balanced trees of height k is greater than as k

Proof: As in the proof of LEMMA 5.IV.1 we have the following recursive formula for the number of leaves of height-balanced trees of height k.

$$T(0) = 1$$

$$T(1) = 4$$

$$T(k) = 2T(k-2)*(F(k-1)\ F(k-2)) + 2*T(k-2)*F(k-1)$$

and a recursive formula for the number of nodes of height-balanced trees of height k

$$N(0) = 1$$

$$N(1) = 7$$

$$N(k) = 2*(N(k-1)*F(k-1) + F(k-2)) + 2*N(k-2)*F(k-1) + F(k-1)^2 + 2*F(k-1)*F(k-2)$$

To account for the term $2*N(k-1)*(F(k-1) + F(k-2))$ note that any node in a height-balanced tree of height k-1 is magnified by the F(k-1) choices for the subtree opposite it when it is embedded in a tree of height k with a balance factor of zero in the root and it is magnified by F(k-2) when it is embedded in a tree of height k with a non-zero balance factor in the root. As usual, the factor 2 accounts for the symmetry. A similar analysis for the case where a node in a tree of height k-2 is embedded as a subtree in a tree of height k yields the term $2*N(k-2)*F(k-2)$. Note finally, that the root is counted $F(k-1)^2$ for trees with a balance factor of zero in the root and $2*F(k-1)*F(k-2)$ for trees with a non-zero balance factor in the root. Then the ratio

$$\frac{T(k)}{N(k)} = \frac{T(k-1) + \dfrac{2*F(k-1)*T(k-2)}{2*(F(k-1)+F(k-2))*F(k-1)}}{N(k-1) + \dfrac{2*F(k-1)*N(k-2)+F(k-1)\{2\}+2*N(k-1)*F(k-2)}{2*(F(k-1)+F(k-2))*N(K-1)}}$$

$$> \frac{T(k-1)}{N(k-1) + \dfrac{N(k-1)+F(k-1)}{N(k-1)}}$$

$$> \frac{T(k-1)}{N(k-1)} * \frac{1}{1 + \dfrac{N(k-2)+F(k-1)}{N(k-1)^2}}$$

$$\frac{T(k-1)}{N(k-1)} * \frac{1}{1 + \dfrac{2*N(k-1)}{N(k-1)^2}}$$

Then for any q > k we have the ratio

$$\frac{T(q)}{N(q)} > \frac{T(k-1)}{N(k-1)} * \prod_{v=k+1} \frac{1}{1 + \dfrac{2}{N(v-1)^2}}$$

Again let $\gamma = \sqrt{2/3}$. Then it will be no suprise if we substitute $\gamma^{2^{k-1}}$ for N(k-1) in our product development to obtain the inequality

$$\frac{T(q)}{N(q)} > \frac{T(k-1)}{N(k-1)} \prod_{j=k}^{\infty} \left(1 + \frac{1}{2^{j-2}}\right)$$

and certainly our product development converges to a strictly positive limit.

III INSERTION ALGORITHMS FOR K-BALANCED TREES

In this section we extend the lower bound results of section 5.I to the k-balanced trees originally introduced by C.C. Foster [10]. K-balanced trees are defined the same way as height-balanced trees except that the height difference of subtrees is allowed to vary by an integer k. Here is an example of a 2-balanced tree.

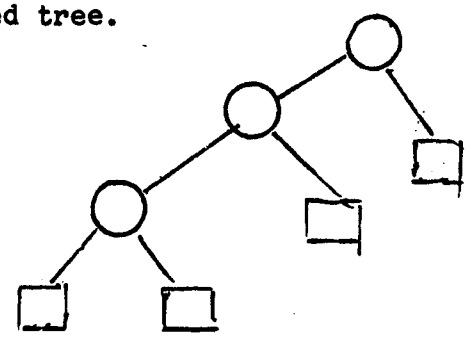


Figure 5.III.1

By way of illustration we prove the lower bound first for 2-balanced trees and then extend the result to general k.

THEOREM 5.III.1 There exists a positive effectively computable constant α_2 such that the probability $q_2(h)$ that the insertion of a new element into an arbitrary 2-balanced tree of height h will imbalance the tree satisfies $q_2(h) > \alpha_2$

THEOREM 5.III.2 There exists a positive effectively computable constant k such that the probability $_k(h)$ that an insertion of a new element into an arbitrary k-balanced tree of height h will imbalance the tree satisfies $_k(h) > _k$. for h > than fixed j.

Proof of THEOREM 5.III.1: let $G_2(h)$ enumerate failure nodes of all extended 2-balanced trees of height h at which the insertion of a new element will imbalance the tree. Let $H_2(h)$ enumerate failure nodes of all extended 2-balanced trees of height h at which the insertion of a new element will increase the height of the tree. Finally, let $F_2(h)$ enumerate extended 2-balanced trees of height h. Then

$F_2(1) = 1$

$F_2(2) = 3$

$F_2(3) = 33$

$F_2(k) = F_2(k-1)^2 + 2*F_2(k-1)*F_2(k-2) + 2*F_2(k-1)*F_2(k-3)$

To account for this formula node that any extended 2-balanced tree must have as left and right subtrees extended 2-balanced trees of heights k-1, k-2, or k-3. The factor 2 accounts for the symmetry. Now we can give the formula for $H_2(k)$

$H_2(0) = 1$

$H_2(1) = 2$

$H_2(2) = 8$

$H_2(k) = 2*H_2(k-1)*(F_2(k-1) + F_2(k-2))$

$$H_2(2) = 8$$

$$H_2(k) = 2*H_2(k-1)*(F_2(k-1) + F_2(k-2))$$

or equivalently

$$H_2(k) = \prod_{j=3}^{k-1} 2*(F_2(j-1) + F_2(j-2))$$

To see this formula node that if an insertion at a failure node in a 2-balanced tree of height k-1 will increase the height of tree, then the same property will be true for the failure node if the tree is embedded as the left or right subtree of a 2-balanced tree of height k with any 2-balanced tree of height k-1 or k-2 chosen as the subtree oppposite it. Note also that a single failure node is a k-balanced tree of height zero. Now we can give the formula for $G_2(k)$.

$$G_2(1) = 0$$

$$G_2(2) = 0$$

$$G_2(3) = 0$$

$$G_2(k) = 2*G_2(k-1)*(F_2(k-1) + F_2(k-2) + F_2(k-3)) + 2*(G_2(k-2) + G_2(k-3))*F_2(k-1) + 2*H_2(k-1)*F_2(k-3)$$

This formula follows from the following observations. If we embed in a tree of height k a failure node in a tree of height k-1 which would cause an imbalance, then an insertion at this failure node would also cause an

imbalance in the larger tree. The factor $2*F_2(k-1)+ F_2(k-2))$ enumerates the possible choices for the subtree opposite the tree containing our node under the embedding. The term $2*F(k-1)*(G_2(k-2) + G_2(k-3))$ counts the number of ways a failure node at which an insertion would cause an imbalance in a tree of height k-2 or k-3 may be embedded in a 2-balanced tree of height k. To see the least term note that a failure node will result in an imbalance after insertion if 1) it lies in the the taller, by a height difference of two, of the subtrees rooted by its first ancestor with a balance factor of two in absolute value, and 2) an insertion at this failure node increases the height of the subtree. The last term $2*H_2(k-1)*F_2(k-3)$ counts the case where the ancestor with the balance factor of +2 or -2 is the root of the tree itself.

Let $T_2(k)$ be the total number of failure nodes of all extended 2-balanced trees of height k . Then

$$T_2(1) = 2$$

$$T_2(2) = 10$$

$$T_2(3) = 240$$

$$T_2(k) = 2*T_2(k-1)*(F_2(k-1) + F_2(k-2) + F_2(k-3)) + 2*(T_2(k-2) + T_2(k-3))*F_2(k-1)$$

The derivation of this formula is similar to those above. Then the probability that an insertion at an arbitrary failure node of an arbitrary 2-balanced tree of height k will imbalance the tree is equal to the ratio

$$\frac{G_2(2)}{T_2(k)} = \frac{G_2(k-1)}{T_2(k-1)} * \frac{G_2(k)*T_2(k-1)}{G_2(k-1)*T_2(k)}$$

Now we may rewrite $G_2(k)$ as

$$\frac{G_2(k-1)*(2*F_2(k-1) + F_2(k-2) + F_2(k-3)) + 2*(G_2(k-2)+G_2(k-3))*F_2(k-1)+2*H_2(k-1)*F_2(k-3)}{G_2(k-1)}$$

$$= G_2(k-1)*(\text{factor1})$$

Similarly, we may rewrite $T_2(k)$ as

$$T_2(k-1)*\left(2*(F_2(k-1) + F_2(k-2) + F_2(k-3)) + \frac{2*(T_2(k-2) + T_2(k-3))*F_2(k-1)}{T_2(k-1)}\right)$$

$$= T_2(k-1)*(\text{factor2})$$

Then

$$\frac{G_2(k)}{T_2(k)} * \frac{G_2(k-1)}{T_2(k-1)} * \frac{(\text{factor1})}{(\text{factor 2})}$$

$$> \frac{G_2(k-1)}{T_2(k-1)} * \frac{2*(F_2(k-1) + F_2(k-2) + F_2(k-3))}{2*(F_2(k-1) + F_2(k-2) + F_2(k-3)) + \frac{2*(T_2(k-2) + T_2(k-3))*F_2(k-1)}{T_2(k-1)}}$$

$$> \frac{G_2(k-1)}{T_2(k-1)} * \frac{1}{1 + \frac{2*(T_2(k-2) + T_2(k-3))*F_2(k-1)}{T_2(k-1)*(F_2(k-1) + F_2(k-2) + F_2(k-3))}}$$

$$> \frac{G_2(k-1)}{T_2(k-1)} * \frac{1}{1 + \frac{T_2(k-2) + T_2(k-3)}{T_2(k-1)}}$$

$$> \frac{G_2(k-1)}{T_2(k-1)} * \frac{1}{1 + \frac{2*T_2(k-2)}{T_2(k-2)}}$$

Now $T_2(k-1) < 2^{k-2}*F_2(k-2)$ and $T_2(k-1) > F_2(k-1)$. Hence our ratio is greater than

$$\frac{G_2(k-1)}{T_2(k-1)} * \frac{1}{1 + \frac{2^{k-2}*F_2(k-2)}{F_2(k-1)}}$$

$$> \frac{G_2(k-1)}{T_2(k-1)} * \frac{1}{1 + \frac{2^{k-1}}{F_2(k-2)}}$$

Let equal ^{4} 3. Then a simple proof by inducion shows that $F_2(k) > 2^k$

$$\prod_{j=k}^{\infty} \frac{1}{1 + \frac{1}{2^{j-1}}}$$

Then again for any $q > k$

$$\frac{G_2(q)}{T_2(q)} > \frac{G_2(k-1)}{T_2(k-1)} \prod_{j=k}^{\infty} \frac{1}{1 + \frac{2^{j-1})}{2^{j-2}}}$$

It is easy to see that this infinite product converges. Whence the result

Proof of THEOREM 5.III.2: Let $G_k(j)$ count the number of failure nodes of all extended k-balanced trees of height k at which an insertion would imbalance the tree. Let $H_k(j)$ count the number of failure nodes of all extended k-balanced trees of height j at which an insetion would increase the height of the tree. Finally, let $F_k(j)$ enumerate distinct k-balanced trees of height j. Then

$F_k(0) = 1$

$F_k(1) = F_2(1)$

.

.

.

$F_k(k-1) = F_{k-1}(k-1)$

$$F_k(j) = F_k(j-1)^2 + 2*\left(\sum_{q=2}^{k+1} F_k(j-q)\right)*F_k(j-1)$$

$j > k$

Now a formula for $H_k(j)$ can be generated

$H_k(0) = 1$

$H_k(1) = 2$

$H_k(2) = H_2(2)$

.

.

.

$H_k(k-1) = H_{k-1}(k-1)$

$$H_k(j) = 2*H_k(j-1)*\left(\sum_{q=1}^{k+1} F_k(j-q)\right)$$

Then $G_k(j)$ has the formula

$G_k(0) = 0$

$G_k(1) = 0$

.

.

.

$$G_k = 2*G_k(j-1)*\left(\sum_{q=2}^{k+1} F_k(j-q)\right) + 2*F_k(j-1)*$$

$$\left(\sum_{q=2}^{k+1} G_k(j-q)\right) + 2*H_k(j-1)*F_k(j-k-1))$$

Finally we have the formula $T_k(j)$ for the total number of failure nodes of all extended k-balanced trees of height j

$T_k(0) = 1$

$T_k(1) = 2$

$T_k(2) = T_2(2)$

.

.

.

$T_k(k-1) = T_{k-1}(k-1)$

$$T_k(j) = 2*T_k(j-1)*\left(\sum_{q=1}^{k+1} F_k(j-q)\right)+2*F_k(j-1)*\left(\sum_{q=2}^{k+1} T_k(j-q)\right)$$

Then the probability that an insertion at an arbitrary failure node of an arbitrary k-balanced tree will imbalance the tree is the ratio

$$\frac{G_k(j)}{T_k(j)} = \frac{G_k(j-1)}{T_k(j-1)} * \frac{G_k(j)*T_k(j-1)}{T_k(j)*G_k(j-1)}$$

Now rewrite $G_k(j)$ as

$$G_k(j-1)*\left(2*\left(\sum_{q=1}^{k+1} F_k(j-q)\right) + \frac{2*\left(\sum_{q=1}^{k+1} G_k(j-q)\right)*F_k(j-1)}{G_k(j-1)} + \frac{H_k(j-1)*F_k(j-k-1)}{G_k(j-1)}\right)$$

$$= G_k(j-1)*(\text{factor1})$$

Similarly we may rewrite $T_k(j)$ as

$$T_k(j-1)*2*\left(\sum_{q=1}^{k-1} F_k(j-q)\right) + \frac{2*F_k(j-1)*\left(\sum_{q=2}^{k} T_k(j-q)\right)}{T_k(j-1)}$$

$$= T_k(j-1)*(\text{factor2})$$

Then

$$\frac{G_k(j)}{G_k(j)} = \frac{G_k(j-1)}{T_k(j-1)} * \frac{\text{factor1}}{\text{factor2}}$$

$$> \frac{G_k(j-1) * 2*\left(\sum_{q=1}^{k+1} F(j-q)\right)}{T_k(j-1) * 2*\left(\sum_{q=1}^{k+1} F_k(j-q) + \frac{\sum_{q=2}^{k+1} T_k(j-q))*F_k(j-1)}{T_k(j-1)}\right)}$$

$$> \frac{G_k(j-1)}{T_k(j-1)} * \frac{1}{1 + \frac{2*\left(\sum_{q=2}^{k+1} T_k(j-q))*F_k(j-1)\right)}{\sum_{q=2}^{k+1} F_k(j-q))*T_k(j-1)}}$$

$$> \frac{G_k(j-1)}{T_k(j-1)} * \frac{1}{1 + \frac{2*\left(\sum_{q=2}^{k+1} T_k(j-q)\right)}{T_k(j-1)}}$$

$$> \frac{G_k(j-1)}{T_k(j-1)} * \frac{1}{1 + \frac{2*(k-1)*T_k(j-2)}{T_k(j-1)}}$$

We observe $T_k(j-1) > F_k(j-1)$ and $T_k(j-2) < 2^{j-2}*F_k(j-2)$

Then

$$\frac{G_k(j)}{T_k(j)} > \frac{G_k(j-1)}{T_k(j-1)} * \frac{1}{1 + \frac{(k-1)*2^{j-1}*F_k(j-2)}{F_k(j-1)}}$$

Again let $\gamma = \sqrt[4]{3}$. As in previous proofs it can be shown $F_k(r) > \gamma^{2^r}$ for $r > 2$. Then for any $p > j$

$$\frac{G_k(p)}{T_k(p)} > \frac{G_k(j-1)}{T_k(j-1)} * \prod_{r=j} \frac{1}{1 + \frac{(k-1)*2^{*r-1}}{2^{r-2}}}$$

We observe that the infinite product development converges to a strictly positive limit which is so close to unity as to have a negligible impact on the lower bound for even small values of j.

PROBLEMS FOR FUTURE RESEARCH CHAPTER 6

What follows is a list of problems which have arisen in the course of the research for this thesis. Some can certainly solved by extensions of the methods deployed in the proofs of this thesis, but others would probably require new techniques for solutions. In any event these problems suggest the rich combinatorial structure underlying height-balanced trees.

1) One of the most attractive directions for future research lies in the connection (established in this thesis) between fringe analysis and urn models for aftereffect. It would be of particular interest if the methods of this thesis could be applied to estimate the function F(N,n). Again, a comprehensive study of the literature for this class of urn models may yield new ideas for fringe analysis.

2) In section 4.III it was established that the maximal number of rotations of height 2 which could occur in the generation of a height-balanced tree with n nodes by insertions is n/2 - 1. In general, we conjecture that the maximal number of rotations of height k which can occur in the generation of an AVL tree with n nodes through insertions is $n/(2^{k-1}) - 1$. Remark that exactly

$n/2^{k-1} - 1$ rotations of height k occur in the generation of the height-balanced tree corresponding to the identity permutation (see THEOREM 4.I.2).

3) When an insertion imbalances a height-balanced tree there are two general types of rotations which would be required to rebalance the tree: single (LL or RR) or double (LR or RL) rotations. One natural extension of the research of Chapter 5 would be to deploy the infinite product development methods to obtain bounds for the probability that an insertion of an arbitrary failure node of an arbitrary AVL tree of height h will result in an imbalance that requires either a single or a doulble rotation.

4) In another direction random binary search trees have become a source for new research in mathematical analysis. In [5], Rheingold and Bagchi use enumerating functions for random binary search trees to identity a function f, closely related to the structure of the trees, on the domain D = (0, 1/2] such that for any $\alpha \in D$, f is discontinuous at α if α is irrational, and f is continuous at α if α is irrational.

REFERENCES

[1] Adelson-Vel'skii, G.M., and Landis, E.M., "An algorithm for the organization of information," Dokl. Acad. Nauk SSSR, 146(1962), pp. 1259-63; English translation

[2] Aho, Hopcroft, Ullman, THE DESIGN AND ANALYSIS OF COMPUTER ALGORITHMS, Addison-Wesley, Reading, Mass., 1974.

[3] Arbib,M., Kfoury, A.J., Moll R., A BASIS FOR THEORETICAL COMPUTER SCIENCE, Springer-Verlag, New York, 1981.

[4] Baer, J.L, "Weight-balanced Trees," AFIPS Conference Proceedings, 44, (1975, National Computer Conference Proceedings) pp. 467-472.

[5] Bagchi, A., and Reingold, E., "A Naturally Occurring Function Continuous Only at Irrationals," American Math. Monthly, June-July, 1982.

[6] Bayer, R., and McCreight, E., "Organization and Maintenance of Large Ordered Indices," Acta Informatica (1972), pp.173-189.

[7] Brown, M.R., "A Partial Analysis of Random Height-balanced Trees," Siam J. Computing, Vol. 8. No. 1, Feb. 1979.

[8] Brown, M.R., and Tarjan, R.E., "A Fast Merging Algorithm," J. of the Asso. for Computing Machinery, Vol 26, No. 2, April 1979.

[9] Feller, William, INTRODUCTION TO PROBABILITY THEORY AND ITS APPLICATIONS, John Wiley, 1957, p.119.

[10] Foster, C.C., "A Generalization of AVL Trees," CACM 16(1973),pp.87-90.

[11] Horowitz and Sahni, FUNDAMENTALS OF DATA STRUCTURES, Computer Science Press, Rockville, Md.,1976.

[12] Karlton, P.L., Fuller, S.H., Scroggs, R.E., Kaehler, E.B., "Performance of Height-balanced Trees," CACM, Vol. 19, 1976, pp.23-28.

[13] Knuth,D.E., THE ART OF COMPUTER PROGRAMMING, vol. 2, Addison-Wesley, Reading, Mass., 1973.

[14] Luchio, F. and Pagli, L., "On the Height of Height-balanced Trees," IEEE Transactions on Computers, Vol. 25,1976, pp.87-90.

[15] Melhorn, K., "A Partial Analysis of Height-balanced Trees under Random Insertions and Deletion," Siam J. of Computing, Vol. 11, No. 4, Nov. 1982, pp. 748-760.

[16] Melhorn, K., EFFEZIENTE ALGORITHMEN STUDIERENBUCHER, Teubner-Verlag, Leipzig, 1977.

[17] Nievergelt, J., and Reingold, E., "Binary Search Trees of Bounded Balance," Siam J. of Computing, Vol. 2, No. 1, March 1973.

[18] Nievergelt, J., and Wong, C.C., "On Binary Search Trees," Proceedings IFIP Conference, 1971, North-Holland, Amsterdam, 1972.

[19] Parzen, Emanual, MODERN PROBABILITY THEORY AND ITS APPLICATIONS, John Wiley, 1960, p.208.

[20] Tarjan, R.E., "Complexity of Combinatorial Algorithms," Siam Review, Vol. 20, No. 3, July, 1978, pp. 457-491.

[21] Teorey, T.J., and Fry, J.P., DESIGN OF DATA BASE STRUCTURES, Prentice-Hall, Englewood Cliffs, N.J.,1982.

[22] Yau, A., "On Random 2-3 Trees," Acta. Information, 9(1978)