1-1-1979

Optimal 2,3-Trees

Nicholas J. Pippenger
Harvey Mudd College

Raymond E. Miller
IBM

Arnold L. Rosenberg
IBM

Lawrence Snyder
IBM

SIAM J. COMPUT.
Vol. 8, No. 1, February 1979

0097-5397/79/0801-0004 $01.00/0

OPTIMAL 2,3-TREES*

RAYMOND E. MILLER†, NICHOLAS PIPPENGER†, ARNOLD L. ROSENBERG†
AND LAWRENCE SNYDER‡

Abstract. The 2,3-trees that are optimal in the sense of having minimal expected number of nodes visited per access are characterized in terms of their "profiles". The characterization leads directly to a linear-time algorithm for constructing a K-key optimal 2,3-tree for a sorted list of K keys. A number of results are derived that demonstrate how different in structure these optimal 2,3-trees are from their "average" cousins.

Key words. 2,3-trees, B-trees, enumeration

Introduction. Many algorithms use as their principal data structure a "search tree" in which records may be located when present, inserted when absent, and deleted when unwanted in time logarithmic in their number. AVL trees and 2,3-trees (a/k/a 3-2 trees, a/k/a 2-3 trees) are examples of this kind of structure. Both have the property that a number of different representations for the same set of records are permissible within the limits of the definitions of the respective structures. The logarithmic performance is guaranteed regardless of which structure arises, but a natural question is, "What, if any, are the quantitative differences among these different representations?"

This paper addresses that question for 2,3-trees and their generalization, B-trees. We derive a characterization of those 2,3-trees (§ 2) and those B-trees (§ 4) that are optimal in the sense of having minimal expected path length per access. Our characterization directly yields a linear-time algorithm for constructing optimal trees. We round out our study by demonstrating how different in structure these optimal trees are from their "typical" cousins and how rare they are in the forests of 2,3-trees and B-trees, respectively (§ 3).

1. 2,3-Trees and their costs. In this section we prepare the way for our study of optimal 2,3-trees. We assume familiarity with trees and their related notions.

(1.1) A 2,3-*tree* is a rooted, oriented tree each of whose nonleaf nodes has either 2 or 3 successors, and all of whose root-to-leaf paths have the same length. We assume the root is not a leaf.

The use of 2,3-trees as balanced search trees (which use originates in unpublished work by Hopcroft) involves placing keys at the nonleaf nodes of the trees—the leaves are dummy nodes—according to the following discipline. A node with s successors ($s = 2,3$) accomodates $s-1$ keys. All keys in the left (resp., right) subtree rooted at a given node are smaller in magnitude (resp., larger in magnitude) than the key(s) resident in the node; should $s = 3$, the keys in the center subtree are strictly intermediate in magnitude between the keys resident in the node; see Fig. 1. The reader familiar with the literature on 2,3-trees will recognize this description as cleaving to the variant presented by Knuth [2, § 6.2.3] rather than that discussed in [1, §§ 4.4, 4.5].

* Received by the editors May 2, 1977.

† Mathematical Sciences Department, IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598.

‡ Mathematical Sciences Department, IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598. Permanent Address: Department of Computer Science, Yale University, New Haven, Connecticut 06520.

We now delineate those structural features of 2,3-trees that enter into our characterization of optimal trees.

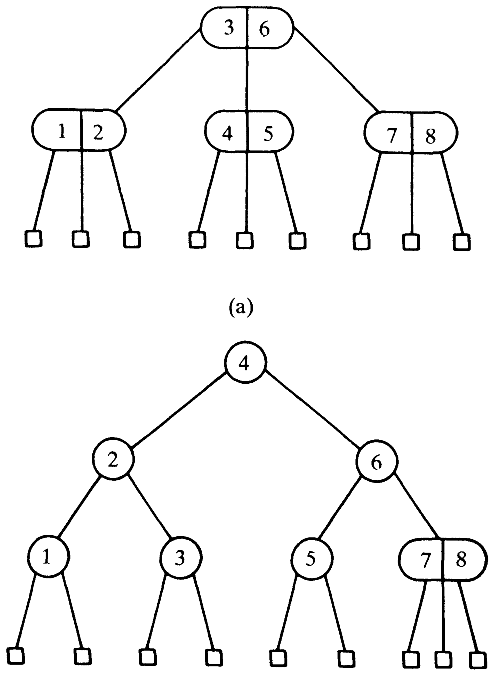


FIG. 1. (a) *A bushy* 2,3-*tree with* 8 *keys.* (b) *A scrawny* 2,3-*tree with* 8 *keys.*

(1.2) The root of a 2,3-tree is said to be at *level* 0; the direct successors of a node at *level* l are said to be at *level* $l+1$. The *depth* of the tree is the (common) level d of its leaves. The *height* of a node is $d-$(its level).

With each level l of a 2,3-tree, we associate three integers:

$\beta_l =$ the number of *b*inary (2-successor) nodes at level l;
$\tau_l =$ the number of *t*ernary (3-successor) nodes at level l;
$\nu_l =$ the *n*umber of nodes at level l.

We combine these integers to yield the following descriptors of the tree.

(1.3) (a) The *profile* of a depth d 2,3-tree is the sequence

$$\Pi = \nu_0, \nu_1, \cdots, \nu_d.$$

(b) The *detailed profile* of the same tree is the sequence

$$\Delta = \langle \beta_0, \tau_0 \rangle \langle \beta_1, \tau_1 \rangle \cdots \langle \beta_d, \tau_d \rangle.$$

The reader can easily verify the following useful relationships among the quantities we have been discussing:

(1.4) (a) $\nu_0 = 1$;
(b) $\nu_d = 1 +$(the number of keys in the tree);
(c) $\nu_l = \beta_l + \tau_l$;
(d) $\nu_{l+1} = 2\beta_l + 3\tau_l$;
(e) (the number of keys at level l) $= \beta_l + 2\tau_l = \nu_{l+1} - \nu_l$.

The 8-key trees of Fig. 1 enjoy the following descriptors.

	Tree of Fig. 1(a)	Tree of Fig. 1(b)
Π	1,3,9	1,2,4,9
Δ	$\langle 0, 1\rangle\langle 0, 3\rangle\langle 0, 0\rangle$	$\langle 1, 0\rangle\langle 2, 0\rangle\langle 3, 1\rangle\langle 0, 0\rangle$

There are at least two significant measures of the cost of a 2,3-tree, the expected number of key-comparisons per access and the expected number of node-visits per access. The latter measure would likely be the more significant in an environment where a ternary comparator were available or in a paging environment where edge-traversals carried with them the danger of page faults. The former measure would likely be the more significant in an environment where the entire tree resided in main memory and only binary comparators were available. In this paper, we study the latter measure of cost; the last two authors have prepared a paper [3] in which they characterize those 2,3-trees that are optimal with respect to the expected number of key-comparisons.

(1.5) The (node-visit) *cost* of a 2,3-tree T with detailed profile $\Delta = \langle\beta_0, \tau_0\rangle \cdots \langle\beta_d, \tau_d\rangle$ is

$$\text{COST}(T) = \sum_{l=0}^{d-1} (l+1)(\beta_l + 2\tau_l).$$

The cost (1.5) is clearly K times the expected number of nodes per visited access if T contains K keys. Clearly, all trees having the same detailed profile are equally costly. In fact this assertion can be strengthened by removing the qualifier "detailed".

LEMMA 1.1. *If the* 2,3-*tree* T *has profile* $\Pi = \nu_0, \nu_1, \cdots, \nu_d$, *then*

$$\text{COST}(T) = d\nu_d - \sum_{l=0}^{d-1} \nu_l.$$

Hence, trees having the same profile are equally costly.

Proof. If one substitutes equation (1.4e) into the expression (1.5) for COST (T), one finds that

$$\text{COST}(T) = \sum_{l=0}^{d-1} (l+1)(\nu_{l+1} - \nu_l). \tag{1.6}$$

Summing (1.6) by parts yields the result directly. □

Lemma 1.1 affords us one easy technique for deriving costs of 14 and 20, respectively, for the trees of Figure 1(a) and 1(b). In fact, the greater cost of the tree of Figure 1(b) is predicted by the following result which asserts that added depth means added cost.

LEMMA 1.2. *Let* T *and* T' *be* 2,3-*trees, both containing* K *keys, having profiles* $\Pi = \nu_0, \cdots, \nu_d$ *and* $\Pi' = \nu'_0, \cdots, \nu'_e$, *respectively. If* $d < e$, *then* COST $(T) <$ COST (T').

Proof. The positivity of the difference

$$\text{COST}(T') - \text{COST}(T) = e\nu'_e - d\nu_d - \sum_{k=0}^{e-1} \nu'_k + \sum_{l=0}^{d-1} \nu_l$$

is easily established via the following facts: (a) $\nu'_e = \nu_d$ by (1.4b); (b) $e - d \geqq 1$ by hypothesis; (c)

$$\sum_{k=0}^{e-1} \nu'_k < \nu'_e = \nu_d$$

since a 2,3-tree has more leaves than internal nodes; (d)

$$\sum_{l=0}^{d-1} \nu_l \geqq 1$$

since we insist that roots not be leaves. □

Lemma 1.2 points at a necessary condition for cost-optimality of a 2,3-tree, namely, minimum depth. The nonsufficiency of this condition is illustrated by the two 5-key, depth 2 trees of Fig. 2: the tree of Fig. 2(a) has cost 8 while that of Fig. 2(b) has cost 9. Thus our characterization of optimal trees must await further conditions, which will be developed in the next section.

For the remainder of the paper, we shall adopt the following abbreviations whose motivation will become clear in § 2:

(1.7) A K-key 2,3-tree is *bushy* if its cost (1.5) is minimum among K-key 2,3-trees. The tree is *scrawny* if its cost is maximum among these trees.

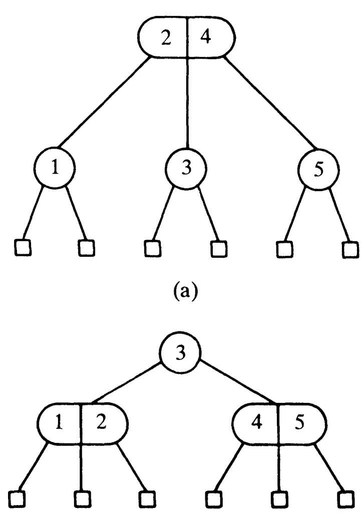


FIG. 2. (a) *A bushy* 2,3-*tree with* 5 *keys*. (b) *A scrawny* 2,3-*tree with* 5 *keys*.

2. Bushy trees. In this section we develop the two components of our main result. We begin with our characterization of bushy trees; and we follow with the linear-time algorithm that derives from the characterization. We close the section with a discussion of an interesting sidelight of our development.

2.1. The characterization theorem. We lead up to our theorem with two lemmas that expose facets of the structure of bushy trees that are needed in the theorem. These structural properties are of some interest in their own rights.

(2.1) Let the 2,3-tree T have profile $\Pi = \nu_0, \nu_1, \cdots, \nu_d$. The *k-prefix* of T $(1 \leqq k \leqq d)$, denoted $T^{(k)}$, is the 2,3-tree obtained by replacing all of T's level k nodes by leaves. $T^{(k)}$ thus has profile $\Pi^{(k)} = \nu_0, \nu_1, \cdots, \nu_k$.

LEMMA 2.1. *Every prefix of a bushy tree is bushy.*

Proof. Say for contradiction that the prefix $T^{(k)}$ of the bushy tree T is not bushy. Let T' be a bushy tree with the same number of keys—hence, the same number of leaves—as $T^{(k)}$. Let T^* be the tree obtained by appending to each leaf of T' the subtree rooted at the corresponding leaf of $T^{(k)}$ in T. The construction of T^* should be obvious from Fig. 3.

Now, T^* clearly contains the same number of keys as does T. However, it is a straightforward matter to verify (using Lemma 1.1) that

$$\text{COST}(T^*) \leqq \text{COST}(T) - \text{COST}(T^{(k)}) + \text{COST}(T') < \text{COST}(T),$$

which contradicts the alleged bushiness of T. □

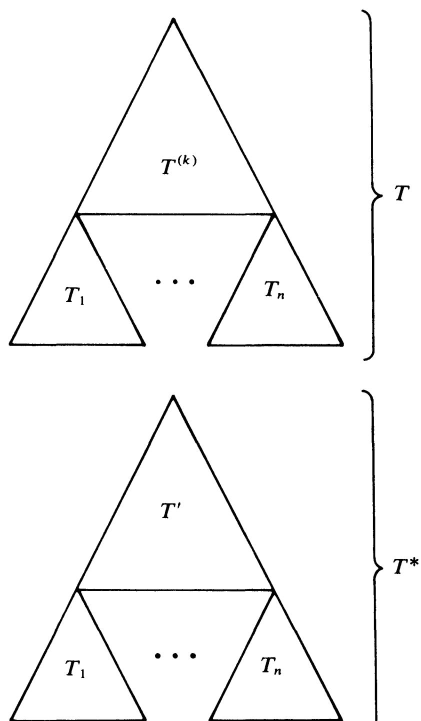


FIG. 3. *The construction of T^* from T and T' in the proof of Lemma 2.1.*

PROPOSITION 2.2. *There exist bushy trees with nonbushy subtrees. Thus, Lemma 2.1 cannot be strengthened by replacing "prefix" by "subtree".*

Proof. Immediate upon comparing the trees of Fig. 4. □

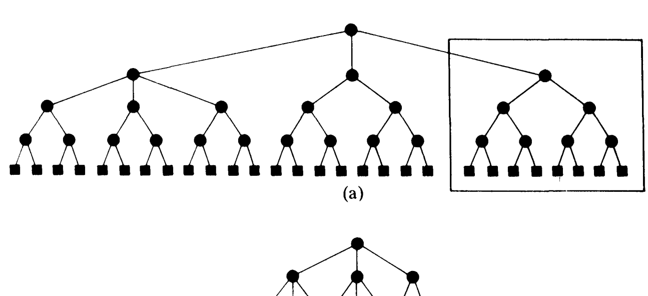


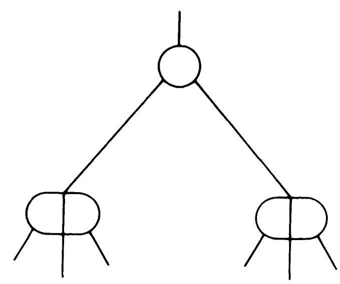
FIG. 4. *Bushy trees with* (a) 27 *and* (b) 7 *keys for the proof of Propositions* 2.2.

The next lemma peers a bit deeper into the structure of bushy trees.

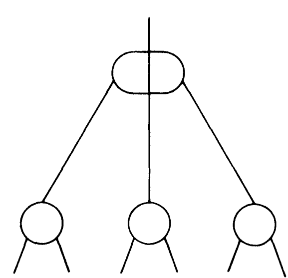
LEMMA 2.3. *If a 2,3-tree has a binary node at level i ($\beta_i > 0$) and two (or more) ternary nodes at level $j > i$ ($\tau_j > 1$), then it is not bushy.*

Proof. Suppose, for contradiction, that the bushy tree T has a binary node at level i and two ternary nodes at level $j > i$. We may assume that $j = i+1$, for if not, we must have a binary node at level $i+1$ (since $\nu_{i+1} \geqq 2$), and we may shift attention from the original binary node to this one. Continuing in this way, we must find a level l with $\beta_l > 0$ and $\tau_{l+1} > 1$.

Since trees with the same profile are equal in cost (Lemma 1.1), we may assume further that the ternary nodes are direct successors of the binary node. Thus we have the configuration



in T (or in a tree that shares T's profile). It is easy to verify, however, that the cost (1.5) can be reduced by replacing this configuration with the configuration



This contradicts T's alleged bushiness. □

Although the necessary condition of Lemma 2.3 is not sufficient—cf. the 7-key trees of Fig. 4—it combines with the depth condition of Lemma 1.2 to yield the sought characterization. The conjoined conditions are best presented in the following numerological setting.

(2.2) The profile $\Pi = \nu_0, \cdots, \nu_d$ of a K-key 2,3-tree is *dense* if
(a) $d = \lceil \log_3 (K+1) \rceil$;
(b) $\nu_l = \min (3^l, \lfloor \nu_{l+1}/2 \rfloor)$ for $1 \leqq l \leqq d-1$.
Note that $\nu_0 = 1$ and $\nu_d = K+1$ automatically.

THEOREM 2.4. *A 2,3-tree is bushy iff it has a dense profile.*

Proof. It will suffice to show that a bushy tree has a dense profile, for once this is done the following argument gives the converse. Let T have a dense profile and let T' be bushy. Then T' has a dense profile. Since there is only one dense profile, T' has the same profile as T. Since the profile determines the cost (Lemma 1.1), T' has the same cost as T. Thus T is also bushy.

Suppose T is a bushy tree with K keys. We seek to show that its profile satisfies (2.2a) and (2.2b). The first of these is easy, for $d = \lceil \log_3 (K+1) \rceil$ is clearly the minimum possible depth of a 2,3-tree with K keys, and by Lemma 1.2 a deeper tree would have a greater cost.

For the rest, we proceed by induction. The result is trivial for 1 or 2 keys; let us assume that it holds for all trees with fewer than K keys and prove it for those with K keys.

It will suffice to prove (2.2b) for $l = d-1$, for once this is done, we may consider the prefix of T of depth $d-1$. By Lemma 2.1, this is bushy. By inductive hypothesis, it has a dense profile. This gives (2.2b) for the remaining values of l.

It remains to prove that

$$\nu_{d-1} = \min(3^{d-1}, \lfloor \nu_d/2 \rfloor).$$

Clearly,

$$\nu_{d-1} \leqq 3^{d-1},$$

for a 2,3-tree cannot have more than 3^{d-1} nodes at level $d-1$. Furthermore,

$$\nu_{d-1} \leqq \lfloor \nu_d/2 \rfloor,$$

for every node at level $d-1$ has at least 2 successors. We must show that one of these bounds is attained. Suppose, on the contrary, that

$$\nu_{d-1} < 3^{d-1}$$

and

$$\nu_{d-1} < \lfloor \nu_d/2 \rfloor.$$

From the first of these it follows that there is a binary node at or above level $d-2$, and from the second it follows that there are at least two ternary nodes at level $d-1$. Thus, by Lemma 2.3, T is not bushy, a contradiction. □

It follows immediately from the Theorem that a bushy tree has at most two "active" or unsaturated levels.

PROPOSITION 2.5. *If* $\Pi = \nu_0, \cdots, \nu_d$ *is a dense profile, then* $\nu_i = 3^i$ *for all* $i < d-2$.

Proof. Since $d = \lceil \log_3 \nu_d \rceil$ by (2.2a), we know that $\nu_d > 3^{d-1}$. Hence, $\nu_d/8 > 3^{d-1}/8 = 3^d/24 > 3^d/27 = 3^{d-3}$, so that $\nu_{d-3} = 3^{d-3}$ by (2.2b), since $\lfloor\lfloor\lfloor \nu_d/2\rfloor/2\rfloor/2\rfloor > \nu_d/8 - 1$. Moreover, since

$$3^k \leqq \lfloor 3^{k+1}/2 \rfloor \quad \text{for all } k,$$

we are assured that $\nu_i = 3^i$ for all $i \leqq d-3$, as was claimed. □

2.2. An algorithm for constructing bushy trees. Our characterization of bushiness in terms of dense profiles yields directly an algorithm for constructing a bushy tree for a given set of keys. If the input set of keys is already sorted, then the algorithm is linear in the size of the set; otherwise, the algorithm operates in time $O(K \log K)$. (No better timing could be expected since the set is sorted once it resides in the tree.)

We interlace our description of the algorithm with an example.

THE ALGORITHM. Our algorithm can best be described in four phases.

Phase 1. Given the cardinality K of the set of keys to be stored, use the prescription (2.2) to construct the profile of a bushy K-tree.

(2.3) $$\text{If } K = 14, \text{ then } \Pi = 1, 3, 7, 15.$$

Phase 2. Using the equations

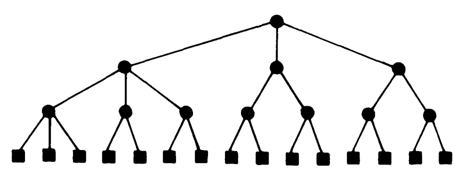
(2.4) $$\beta_i = 3\nu_i - \nu_{i+1}, \qquad \tau_i = \nu_{i+1} - 2\nu_i,$$

construct the detailed profile of the tree from its profile.

(2.5) Given the profile Π of (2.3), we have $\Delta = \langle 0, 1\rangle\langle 2, 1\rangle\langle 6, 1\rangle\langle 0, 0\rangle$.

Phase 3. Construct the "skeleton" of the tree from its detailed profile; that is, decide how to place the binary and ternary nodes at those levels that have both. Clearly, this decision will not affect the cost of the resulting tree, but the layout may affect the efficiency of subsequent transactions with the key set. Other things being equal, a decision to *left-bias* the tree by forcing all ternary nodes as far to the left as possible is as good as any other.

(2.6) The left-biased tree with the detailed profile Δ of (2.5) has the following appearance:



Phase 4. Traverse the tree of Phase 3 in FILLORDER, dropping off the keys in ascending order as one goes.

(2.7) To traverse a tree in FILLORDER, follow the ensuing recursive prescription.
1. Visit the left subtree in FILLORDER.
2. Visit the root, and deposit a key.
3. Visit the center subtree in FILLORDER. } for ternary roots only.
4. Visit the root, and deposit a key. } for ternary roots only.
5. Visit the right subtree in FILLORDER.

(2.8) We finally complete the example of (2.3), (2.5), (2.6). We use the key set $\{1, \cdots, 14\}$ to illustrate in Figure 5 the FILLORDER of the tree of (2.6).

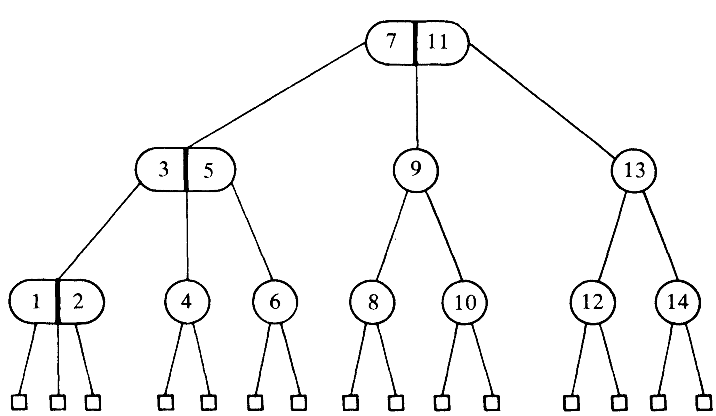


FIG. 5. *The* FILLORDER *of the tree* (2.5).

Timing of the Algorithm. We assume that our algorithm is to be executed on a uniform-cost RAM [1, § 1]. Accordingly, we assess time $O(\log K)$ for the $\lceil \log_3 (K+1)\rceil$ operations performed in Phase 1 and for the $2\lceil \log_3 (K+1)\rceil$ linear-form evaluations in Phase 2. Phases 3 and 4, which likely would be done simultaneously in an efficient implementation, can be seen to take time $O(K)$ to perform if the list of keys is sorted, and time $O(K \log K)$ otherwise. (Note that FILLORDER traversal of a tree is almost identical to depth-first traversal.)

Discussion. The obvious algorithm for constructing a bushy 2,3-tree would construct the tree top-down, making it as ternary as possible, with some backtracking at the high-numbered levels to ensure a "flat bottom." Our use of profiles and detailed profiles in our algorithm obviates this backtracking, thus enhancing the efficiency of the construction. A logical competitor for any direct-construction procedure would be one that constructs a 2,3-tree by successively inserting, in ascending order, say, the keys one is given, according to the insertion algorithm for 2,3-trees [2, § 6.2.3]. The reader can easily reproduce the induction that demonstrates that trees produced in this way are often very far from bushy. Specifically, whenever, $K = 2^n - 1$, the tree so produced is a purely binary tree!

2.3. Characterizing scrawny trees. There is a striking and appealing duality between our characterization of optimal 2,3-trees on the one hand and the analogous characterization for pessimal or *scrawny* 2,3-trees.

METATHEOREM. *In order to reproduce the results of* § 2.1 *for scrawny trees, perform the following transliteration throughout.*

For		*Read*
2	even in bases of	3
3	logs and exponentials	2
min		max
floor $\lfloor x \rfloor$		ceiling $\lceil x \rceil$
ceiling $\lceil x \rceil$		floor $\lfloor x \rfloor$
τ		β
β		τ

Details are left to the reader.

3. Typical 2,3-trees. We have found the optimal (bushy) and pessimal (scrawny) 2,3-trees; let us have a look at typical, run-of-the-forest 2,3-trees. We shall find that almost all n-leaf 2,3-trees share some remarkable statistical properties involving the golden ratio,

$$\phi = \frac{1+\sqrt{5}}{2} = 1.618 \cdots .$$

These properties, which will allow us to predict the cost of a typical 2,3-tree, will be obtained as by-products of an argument for enumerating 2,3-trees.

Let T_n denote the number of n-leaf 2,3-trees. We shall show that

$$T_n = \frac{1}{n} \phi^n U(1), \tag{3.1}$$

where $U(1)$ denotes a factor of the form $\exp O(1)$. More generally, since we shall be dealing with "error factors" more often than with "error terms", we shall let $U(f(n))$ denote a factor of the form $\exp O(f(n))$. In ordinary language, (3.1) determines T_n to within constant factors.

The major steps in our derivation of (3.1) will be as follows. First we shall obtain a recurrence having the sequence T_n as its unique fixed point (Lemma 3.1). Then we shall show that any sequence that is an "approximately fixed" point of the recurrence must be "approximately equal" to the exact fixed point (Lemma 3.2). Finally we shall show that

$$\frac{1}{n}\phi^n$$

is an approximately fixed point of the recurrence (Lemmas 3.3, 3.4, 3.5). In the derivation, the notions "approximately fixed" and "approximately equal" will be given precise meanings by means of error factors.

This method of proceeding leaves unanswered the question of how the solution was found in the first place. Experience with the enumeration of unlabeled trees in general, and consideration of 2,3-trees in particular, suggests that T_n grows exponentially, say as e^{An}. An attempt to prove this, along the lines indicated above, reveals that A must be $\ln \phi$ and that a correction of the form n^B is necessary. Another pass through the proof reveals that $B=-1$. Though with hindsight we might find a more convincing motivation for the solution (by considering, for example, the singularities of the generating function for T_n), this method of iteration is extremely robust and after the first pass one can usually work out the successive corrections with very little wasted motion. An example of a more formidable problem which was solved in the same way will be given later.

LEMMA 3.1. *T_n satisfies the recurrence*

$$T_n = \sum_{2\beta+3\tau=n} \binom{\beta+\tau}{\tau} T_{\beta+\tau}. \tag{3.2}$$

Proof. Given an n-leaf 2,3-tree, consider the nodes at height one, that is, the nodes whose successors are leaves. If β and τ denote the number of binary and ternary nodes, respectively, at height one, then $2\beta+3\tau=n$.

An n-leaf 2,3-tree can be constructed by the following three-step procedure. First, choose β and τ satisfying $2\beta+3\tau=n$. Second, choose the structure of the tree below height one. This amounts to choosing τ of the $\beta+\tau$ nodes at height one to be ternary nodes, leaving the remaining β to be binary nodes. This can be done in

$$\binom{\beta+\tau}{\tau}$$

ways. Third, choose the structure of the tree above height one. This amounts to choosing a $(\beta+\tau)$-leaf 2,3-tree, and can be done in $T_{\beta+\tau}$ ways. Since each n-leaf 2,3-tree can be constructed in exactly one way by this procedure, we arrive at (3.2). □

The recurrence (3.2), together with the initial conditions $T_2=1$ and $T_3=1$, completely determines T_n.

LEMMA 3.2. *If S_n is a positive sequence satisfying*

$$S_n = U(f(n)) \sum_{2\beta+3\tau=n} \binom{\beta+\tau}{\tau} S_{\beta+\tau}, \tag{3.3}$$

where $f(n)$ is an eventually decreasing positive function such that

$$\sum_{0\leqq t\leqq\infty} f(2^t) \tag{3.4}$$

converges, then

$$T_n = S_n U(1). \tag{3.5}$$

Proof. Let A_N denote the maximum of S_n/T_n for $1 \leqq n \leqq N$. If N is large enough that $f(n)$ is decreasing beyond N, and if $N \leqq n \leqq 2N$, then

$$\begin{aligned} S_n &= U(f(n)) \sum_{2\beta+3\tau=n} \binom{\beta+\tau}{\tau} S_{\beta+\tau} \\ &\leqq U(f(n)) \sum_{2\beta+3\tau=n} \binom{\beta+\tau}{\tau} A_N T_{\beta+\tau} \\ &= U(f(n)) A_N T_n \\ &\leqq U(f(N)) A_N T_n, \end{aligned}$$

since $n \leqq 2N$ implies $\beta+\tau \leqq N$, and $f(n) \leqq f(N)$. Thus

$$A_{2N} \leqq A_N U(f(N)),$$

and by induction

$$\begin{aligned} A_{2^t N} &\leqq A_N \prod_{0 \leqq s < t} U(f(2^s N)) \\ &= A_N U\Big(\sum_{0 \leqq s < t} f(2^s N)\Big). \end{aligned}$$

Letting $t \to \infty$ with N fixed, we have

$$A_{2^t N} \leqq U(1),$$

since A_N is positive and (3.4) converges. Thus

$$S_n / T_n \leqq U(1);$$

a similar argument shows that

$$S_n / T_n \geqq U(1),$$

so (3.5) is proved. □

It remains for us to show that

$$\frac{1}{n} \phi^n$$

is an approximately fixed point of our recurrence. Specifically, we shall show that

$$\sum_{2\beta+3\tau=n} \binom{\beta+\tau}{\tau} \frac{1}{\beta+\tau} \phi^{\beta+\tau} = \frac{1}{n} \phi^n U\Big(\frac{\log^{3/2} n}{n^{1/2}}\Big). \tag{3.6}$$

This will be done in three steps as follows. First we shall estimate the summand, separating our estimate into algebraically varying factors (which are $U(\log n)$) and exponentially varying factors (which are $U(n)$). We shall then focus our attention on the exponentially varying factors and see that they impart to the summand a peaking reminiscent of the central limit theorem: the greatest contribution to the sum comes from those terms in which β and τ are in certain fixed ratios to $\beta+\tau$ and hence to n. Finally we shall use this central peaking to estimate the sum.

Successive values of β and τ differ by 3 and 2, respectively; it will be convenient to have an index whose successive values differ by 1. Thus we introduce the index m satisfying

$$\tau = 2m, \qquad \beta = \frac{n}{2} - 3m, \qquad \beta + \tau = \frac{n}{2} - m.$$

This index assumes integral values if n is even and half-integral values if n is odd.

LEMMA 3.3.

$$\binom{\beta+\tau}{\tau}\frac{1}{\beta+\tau}\phi^{\beta+\tau} = U\left(\frac{1}{\beta}+\frac{1}{\tau}\right)\left(\frac{1}{2\pi\beta\tau(\beta+\tau)}\right)^{1/2} \exp nE\left(\frac{m}{n}\right), \tag{3.7}$$

where

$$E(\mu) = F(G(\mu)), \qquad F(\lambda) = \frac{H(\lambda)+\ln\phi}{2+\lambda},$$

$$G(\mu) = \frac{4\mu}{1-2\mu}, \qquad H(\lambda) = -\lambda \ln \lambda - (1-\lambda)\ln(1-\lambda),$$

and ln *denotes the natural logarithm.*

Proof. For the binomial coefficient, the estimate

$$\binom{\beta+\tau}{\tau} = U\left(\frac{1}{\beta}+\frac{1}{\tau}\right)\left(\frac{\beta+\tau}{2\pi\beta\tau}\right)^{1/2} \exp(\beta+\tau)H\left(\frac{\tau}{\beta+\tau}\right)$$

is an immediate consequence of Stirling's formula. Define λ such that

$$\tau = \lambda(\beta+\tau), \qquad \beta = (1-\lambda)(\beta+\tau), \qquad n = (2+\lambda)(\beta+\tau).$$

Then

$$\binom{\beta+\tau}{\tau}\frac{1}{\beta+\tau}\phi^{\beta+\tau} = U\left(\frac{1}{\beta}+\frac{1}{\tau}\right)\left(\frac{1}{2\pi\beta\tau(\beta+\tau)}\right)^{1/2} \exp nF\left(\frac{\tau}{\beta+\tau}\right).$$

Define μ such that

$$m = \mu n.$$

Then λ and μ are related by

$$\lambda = \frac{4\mu}{1-2\mu}, \qquad \mu = \frac{\lambda}{2(2+\lambda)}.$$

Thus

$$\binom{\beta+\tau}{\tau}\frac{1}{\beta+\tau}\phi^{\beta+\tau} = U\left(\frac{1}{\beta}+\frac{1}{\tau}\right)\left(\frac{1}{2\pi\beta\tau(\beta+\tau)}\right)^{1/2} \exp nE\left(\frac{m}{n}\right),$$

as was to be shown. □

LEMMA 3.4. *The function $F(\lambda)$ assumes its unique maximum (for $0 \leqq \lambda \leqq 1$) at*

$$\Lambda = \phi^{-2}.$$

At this point

$$F(\Lambda) = \ln\phi, \qquad F'(\Lambda) = 0, \qquad F''(\Lambda) = \frac{-1}{(2+\Lambda)\Lambda(1-\Lambda)},$$

where the primes indicate differentiation. Accordingly, $E(\mu)$ assumes its maximum at

$$M=\frac{\Lambda}{2(2+\Lambda)},$$

and at this point

$$E(M)=\ln\phi,\qquad E'(M)=0,\qquad E''(m)=-\frac{(2+\Lambda)^3}{\Lambda(1-\Lambda)}.$$

Proof. We shall let $H(0)=H(1)=0$; this makes $H(\lambda)$, and therefore also $F(\lambda)$, continuous on the closed interval $0\leqq\lambda\leqq 1$. These functions are in fact analytic in the open interval $0<\lambda<1$, and thus $F(\lambda)$ can assume its maximum only where its first derivative vanishes or at an endpoint. We compute the first derivatives

$$H'(\lambda)=\ln(1-\lambda)/\lambda,$$

$$F'(\lambda)=-\frac{H(\lambda)+\ln\phi}{(2+\lambda)^2}+\frac{\ln(1-\lambda)/\lambda}{2+\lambda}.$$

Equating $F'(\lambda)$ with 0 leads to the equation

$$(1-\lambda)^3=\lambda^2\phi.$$

In the interval $0<\lambda<1$ the left side decreases while the right side increases, so there can be at most one solution. This occurs at

$$\Lambda=\phi^{-2},$$

by virtue of the equation

$$1-\phi^{-2}=\phi^{-1}.$$

This gives

$$F(\Lambda)=\ln\phi,$$

which is obviously larger than $F(\lambda)$ at either of the endpoints. We compute the second derivatives

$$H''(\lambda)=\frac{-1}{\lambda(1-\lambda)},\qquad F''(\lambda)=2\frac{H(\lambda)+\ln\phi}{(2+\lambda)^3}-2\frac{\ln(1-\lambda)/\lambda}{(2+\lambda)^2}-\frac{1}{(2+\lambda)\lambda(1-\lambda)}.$$

Since the first two terms are a multiple of $F'(\lambda)$, they vanish at Λ, leaving

$$F''(\Lambda)=\frac{-1}{(2+\Lambda)\Lambda(1-\Lambda)}.$$

All of this can be carried over to $E(\mu)$, $E'(\mu)$, and $E''(\mu)$ through the derivatives

$$G'(\mu)=\frac{4}{(1-2\mu)^2}=(2+\lambda)^2,\qquad G''(\mu)=\frac{16}{(1-2\mu)^3}=2(2+\lambda)^3$$

and the chain rule. □

LEMMA 3.5.

$$\sum_m U\left(\frac{1}{\beta}+\frac{1}{\tau}\right)\left(\frac{1}{2\pi\beta\tau(\beta+\tau)}\right)^{1/2}\exp nE\left(\frac{m}{n}\right)=U\left(\frac{\log^{3/2}n}{n^{1/2}}\right)\frac{1}{n}\phi^n. \tag{3.8}$$

Proof. The major steps of the derivation are as follows. The central peaking of the summand will be exploited, allowing the tails of the summation to be neglected. The decaudated sum can be simplified since the algebraically varying factors behave like constants in the remaining range of summation. The resulting sum will be estimated with an integral, to which the tails previously removed will be restored. The recaudated integral can be evaluated by standard methods.

Our sum is

$$\sum_m W_m,$$

where

$$W_m = U\Big(\frac{1}{\beta}+\frac{1}{\tau}\Big)\Big(\frac{1}{2\pi\beta\tau(\beta+\tau)}\Big)^{1/2} \exp nE\Big(\frac{m}{n}\Big).$$

Since $E(\mu)$ is analytic at M, it can be expanded in a Taylor series about M. The result is

$$E(\mu) = \ln\phi - (\mu - M)^2/\delta^2 + O((\mu - M)^3),$$

where

$$\delta = \Big(\frac{2\Lambda(1-\Lambda)}{(2+\Lambda)^3}\Big)^{1/2}.$$

Thus W_m can be rewritten as

$$W_m = U\Big(\frac{1}{\beta}+\frac{1}{\tau}\Big)\Big(\frac{1}{2\pi\beta\tau(\beta+\tau)}\Big)^{1/2} U((m-Mn)^3/n^2)\phi^n V_m,$$

where

$$V_m = \exp -(m-Mn)^2/\delta^2 n.$$

We shall break our sum into three parts,

$$\sum_m W_m = \sum_{m<a} W_m + \sum_{a\leq m\leq b} W_m + \sum_{b<m} W_m,$$

where

$$a = Mn - \Big(\frac{6\Lambda(1-\Lambda)n\ln n}{(2+\Lambda)^3}\Big)^{1/2},$$

$$b = Mn + \Big(\frac{6\Lambda(1-\Lambda)n\ln n}{(2+\Lambda)^3}\Big)^{1/2}.$$

For any term in the sum over $m < a$,

$$V_m = O(V_a) = O\Big(\frac{1}{n^3}\Big),$$

and the other factors in W_m are $O(\phi^n)$. Since there are $O(n)$ terms,

$$\sum_{m<a} W_m = O\Big(\frac{1}{n^2}\phi^n\Big).$$

A similar argument shows that

$$\sum_{b<m} W_m = O\Big(\frac{1}{n^2}\phi^n\Big),$$

so

$$\sum_m W_m = \sum_{a \leqq m \leqq b} W_m + O\left(\frac{1}{n^2}\phi^n\right). \tag{3.9}$$

For any term in the sum over $a \leqq m \leqq b$,

$$m = MnU\left(\frac{\log^{1/2} n}{n^{1/2}}\right),$$

from which it follows that

$$\tau = \frac{\Lambda n}{2+\Lambda} U\left(\frac{\log^{1/2} n}{n^{1/2}}\right), \qquad \beta = \frac{(1-\Lambda)n}{2+\Lambda} U\left(\frac{\log^{1/2} n}{n^{1/2}}\right), \qquad \beta + \tau = \frac{n}{2+\Lambda} U\left(\frac{\log^{1/2} n}{n^{1/2}}\right),$$

and further that

$$W_m = U\left(\frac{\log^{3/2} n}{n^{1/2}}\right)\left(\frac{(2+\Lambda)^3}{2\pi n^3 \Lambda(1-\Lambda)}\right)^{1/2} \phi^n V_m.$$

Thus

$$\sum_{a \leqq m \leqq b} W_m = U\left(\frac{\log^{3/2} n}{n^{1/2}}\right)\left(\frac{(2+\Lambda)^3}{2\pi n^3 \Lambda(1-\Lambda)}\right)^{1/2} \phi^n \sum_{a \leqq m \leqq b} V_m. \tag{3.10}$$

Now,

$$\sum_{a \leqq m \leqq b} V_m = \int_a^b V_x \, dx + O(1), \tag{3.11}$$

since the total variation of the integrand is $O(1)$. We shall express our integral as the sum of three integrals:

$$\int_a^b V_x \, dx = -\int_{-\infty}^a V_x \, dx + \int_{-\infty}^{+\infty} V_x \, dx - \int_b^{+\infty} V_x \, dx.$$

Integration by parts gives

$$\int_{-\infty}^a V_x \, dx = O\left(\frac{1}{a} V_a\right) = O\left(\frac{1}{n^4}\right).$$

Similar considerations show that

$$\int_b^{+\infty} V_x \, dx = O\left(\frac{1}{n^4}\right),$$

so

$$\int_a^b V_x \, dx = \int_{-\infty}^{+\infty} V_x \, dx + O\left(\frac{1}{n^4}\right). \tag{3.12}$$

Using the transformation

$$x = Mn + \delta n^{1/2} y$$

and the well-known integral

$$\int_{-\infty}^{+\infty} \exp -y^2 \, dy = \pi^{1/2},$$

we obtain

$$\int_{-\infty}^{+\infty} V_x\,dx = \left(\frac{2\pi n\Lambda(1-\Lambda)}{(2+\Lambda)^3}\right)^{1/2}.$$

Working backwards through (3.12), (3.11), (3.10), and (3.9), we arrive at (3.8). □

At last we have

THEOREM 3.6.

$$T_n = \frac{1}{n}\phi^n U(1).$$

Proof. Lemmas 3.3, 3.4, and 3.5, taken together, prove formula (3.6), which, taken together with Lemmas 3.1 and 3.2, proves the theorem. □

The methods we have used to prove this theorem can be used to obtain a fairly complete picture of what a typical n-leaf 2,3-tree looks like. The argument that allowed us to neglect the tails of the sum in Lemma 3.5 shows that, with probability approaching 1 as $n \to \infty$,

$$\beta+\tau = \frac{n}{2+\Lambda}\,U\!\left(\frac{\log^{1/2} n}{n^{1/2}}\right), \qquad \beta = \frac{(1-\Lambda)n}{2+\Lambda}\,U\!\left(\frac{\log^{1/2} n}{n^{1/2}}\right), \qquad \tau = \frac{\Lambda n}{2+\Lambda}\,U\!\left(\frac{\log^{1/2} n}{n^{1/2}}\right).$$

Thus the number of nodes at height one is less than the number of leaves by the factor $2+\Lambda = 2+\phi^{-2} = 2.381\cdots$, and these nodes are partitioned into binary and ternary nodes in the golden ratio $1-\Lambda = \phi^{-1} = 0.618\cdots$, $\Lambda = \phi^{-2} = 0.381\cdots$. The same ratios manifest themselves at greater heights, with the result that, with probability approaching 1 as $n \to \infty$, a 2,3-tree has height $\log_{2+\Lambda} n + O(1)$. This implies that it also has cost $n \log_{2+\Lambda} n + O(n)$. Typical 2,3-trees thus assume a position intermediate between their bushy and scrawny forest-mates:

	bushy	typical	scrawny
cost	$n \log_3 n + O(n)$	$n \log_{2+\Lambda} n + O(n)$	$n \log_2 n + O(n)$

$(\Lambda = 0.381\cdots)$

It should be observed that 2,3-trees that are "typical" in the static sense in which we have used the word (with all n-leaf trees considered equally) have nothing to do with those that are "typical" in the dynamic sense of being grown by the standard insertion algorithm (with all $n!$ orders of insertion considered equally). This is easily seen by comparing the average proportions of binary and ternary nodes derived earlier for the static sense with the corresponding average proportion found by Yao [4] for the dynamic sense:

	static	dynamic
binary nodes: $\beta/(\beta+\tau)$	$1-\Lambda = \phi^{-1}$	$\frac{2}{3}$
ternary nodes: $\tau/(\beta+\tau)$	$\Lambda = \phi^{-2}$	$\frac{1}{3}$

The methods of this section can be applied to the number $T_n^{(3)}$ of bushy n-leaf 2,3-trees or to the number $T_n^{(2)}$ of scrawny trees. These numbers do not behave as smoothly with n as T_n does: for n a power of 3, $T_n^{(3)}=1$ and for n a power of 2, $T_n^{(2)}=1$; for other values of n, $T_n^{(3)}$ and $T_n^{(2)}$ may be large. But one can show that

$$T_n^{(3)}=O(n^{-1/2}\psi^n),\qquad T_n^{(2)}=O(n^{-1/2}\psi^n),$$

where $\psi=1.324\cdots$ is the real root of the equation $\psi^3=\psi+1$. These upper bounds are the best possible, in the sense that they become false if $O(\cdots)$ is replaced by $o(\cdots)$. Since $\psi<\phi$, bushy or scrawny trees constitute an exponentially small fraction of all 2,3-trees.

The methods of this section can also be applied to the number P_n of profiles of n-leaf 2,3-trees. The recurrence

$$P_n=\sum_{2\beta+3\tau=n} P_{\beta+\tau}$$

is obtained by analogy with Lemma 3.1. The solution of this recurrence is the same in outline as that of (3.2), but much more elaborate in detail. The result is

$$P_n=U(1)n^{(1/2)\log_2 n-\log_2\log_2 n+\log_2 e-1/2}(\log_2 n)^{(1/2)\log_2\log_2 n},$$

which is perhaps not what one would have first conjectured.

4. B-trees. All of the results in §§ 1–3 generalize from 2,3-trees to their more practical relatives B-trees [2, § 6.2.3]. Although these generalized results are often harder to prove than their 2,3-relatives, the added difficulty is technical rather than conceptual in nature. Accordingly, we shall discuss the generalizations in only a cursory fashion, pointing out the slight differences in formulation as we go.

(4.1) A *B-tree of order m* ($\geqq 3$) is a rooted, oriented tree whose root has $2\leqq s\leqq m$ successors, whose nonroot interior nodes have $\lceil m/2\rceil\leqq s\leqq m$ successors each, and all of whose root-to-leaf paths have the same length.

(4.2) The *detailed profile* of an order m B-tree T is a sequence of $(m-1)$-tuples

$$\Delta=\langle\sigma_0^2,\sigma_0^3,\cdots,\sigma_0^m\rangle\cdots\langle\sigma_d^2,\sigma_d^3,\cdots,\sigma_d^m\rangle$$

where σ_l^s is the number of s-successor nodes at level l of T.

A 2,3-tree is an order 3 B-tree; the quantities earlier denoted β_l and τ_l are now denoted σ_l^2 and σ_l^3, respectively. Obviously, if $l>1$, all $\sigma_l^s=0$ for $s<\lceil m/2\rceil$.

(4.3) The *cost* of the B-tree T with detailed profile Δ as in (4.2) is

$$\text{COST}(T)=\sum_{l=0}^{d-1}(l+1)\left(\sum_{k=2}^{m}k\sigma_l^k\right).$$

Section 1. The results of § 1 and their proofs translate verbatim to our new setting.

Section 2. Lemma 2.1 and its proof translate verbatim. Lemma 2.3 requires some translation, as follows.

LEMMA 2.3′. *If an order m B-tree has an "unsaturated" node at level i* (*i.e.*, $\sigma_i^s>0$ *for some* $s<m$), *and if it has* (*at least*) $\lceil m/2\rceil-1$ "*available*" *keys at level* $j>i$

$$\left(i.e.,\quad\sum_{\lceil m/2\rceil<k\leqq m}(k-\lceil m/2\rceil+1)\sigma_j^k\geqq\lceil m/2\rceil-1\right),$$

then it is not bushy.

The proof of Lemma 2.3′, as well as that of Theorem 2.4 carry over in a transparent way to B-trees, once one has translated definition (2.2) by replacing 2 and 3 by $\lceil m/2\rceil$ and m, respectively.

The linear-time algorithm for constructing bushy trees requires only two emendations of any substance in order to accommodate general B-trees. First, in Phase 2 of the algorithm, one replaces the equations (2.4) by the equations

$$\nu_i = \sum_s \sigma_i^s, \qquad \nu_{i+1} = \sum_s s\sigma_i^s; \tag{4.4}$$

hence, the detailed profile of the tree is no longer uniquely specified by the profile. However, one can still produce a detailed profile for the tree from the equations (4.4) in time $O(\log K)$, as the reader can easily verify. The second required change is to the definition (2.7) of FILLORDER; the needed change is obvious.

Finally, the duality between the optimal and pessimal B-tree is almost as striking as that between the corresponding 2,3-trees. The major distinction results from the fact that the "2" in 2,3-trees plays the dual role of $\lceil m/2 \rceil$ and the minimal degree of the root. Thus the scrawny order m B-tree profile satisfies the equations

$$d = \lfloor \log_{\lceil m/2 \rceil} (\nu_d/2) \rfloor + 1,$$
$$\nu_l = \max\left(\lceil m/2 \rceil^l, \lceil \nu_{l+1}/m \rceil\right),$$
$$\nu_0 = 1, \qquad \nu_1 = 2.$$

The proof of the characterization theorem, however, mirrors that of the characterization of bushy trees, as is the case with 2,3-trees.

Section 3. The B-tree generalization of § 3 can be done, with much more labor but no more insight. The major observable change is that the golden ratio ϕ is replaced by a less familiar algebraic number whose degree depends on the order of the B-trees studied.

REFERENCES

[1] A. V. AHO, J. E. HOPCROFT AND J. D. ULLMAN, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, MA, 1974.

[2] D. E. KNUTH, *The Art of Computer Programming III: Sorting and Searching*, Addison-Wesley, Reading, MA, 1973.

[3] A. L. ROSENBERG AND L. SNYDER, *Minimal-Comparison* 2,3-*Trees*, this Journal, 7 (1978), pp. 465–480.

[4] A. C.-C. YAO, *Random* 3-2 *Trees*, Acta Inform, 9 (1978), pp. 159–170.