◆◇◆◇◆

NOTES ON FIBONACCI TREES AND THEIR OPTIMALITY*

YASUICHI HORIBE

Shizuoka University, Hamamatsu, 432, Japan
(Submitted February 1982)

INTRODUCTION

Continuing a previous paper [3], some new observations on properties and optimality of Fibonacci trees will be given, beginning with a short review of some parts of [3] in the first section.

1. FIBONACCI TREES

Consider a binary tree (rooted and ordered) with $n - 1$ internal nodes (each having two sons) and n terminal nodes or leaves. A node is at level ℓ if the path from the root to this node has ℓ branches. Assign unit cost 1 to each left branch and cost c ($\geqslant 1$) to each right branch. The cost of a node is defined to be the sum of costs of branches that form the path from the root to this node. Further, we define the *total cost* of a tree as the sum of costs of all terminal nodes. For a given number of terminal nodes, a tree with minimum total cost is called *optimal*. Suppose we have an optimal tree with n terminal nodes. Split in this tree any one terminal node of minimum cost to produce two new terminal nodes. Then the resulting tree with $n + 1$ terminal nodes will be optimal. This growth procedure is due to Varn [6]. (For a simple proof of the validity of this procedure, see [3].)

A beautiful class of binary trees is the class of Fibonacci trees (for an account, see [5]). The *Fibonacci tree of order* k has F_k terminal nodes, where $\{F_k\}$ are the Fibonacci numbers

$$F_0 = 0,\ F_1 = 1,\ F_k = F_{k-1} + F_{k-2},$$

and is defined inductively as follows: If $k = 1$ or 2, the Fibonacci tree

---

*This paper was presented at a meeting on Information Theory, Mathematisches Forschungsinstitut, Oberwolfach, West Germany, April 4-10, 1982.

of order k is simply the root only. If $k \geq 3$, the left subtree of the Fibonacci tree of order k is the Fibonacci tree of order $k - 1$; and the right subtree is the Fibonacci tree of order $k - 2$. The Fibonacci tree of order k will be denoted by T_k for brevity.

Let us say that T_k is *c-optimal*, if it has the minimum total cost of all binary trees having F_k terminal nodes, when cost c is assigned to each right branch, and cost 1 to each left branch.

We have the following properties [3]:

(A) T_k, $k \geq 2$, with cost $c = 2$ has F_{k-1} terminal nodes of cost $k - 2$ and F_{k-2} terminal nodes of cost $k - 1$.

(B) Splitting all terminal nodes of cost $k - 2$ in T_k with $c = 2$ produces T_{k+1}.

(C) T_k is 2-optimal for every k.

By the properties (A) and (B), it may be natural to classify the terminal nodes of T_k into two types, α and β: A terminal node is of type α (α-*node* for short) [respectively, type β (β-*node* for short)], if this node becomes one of the lower [higher] cost nodes when $c = 2$.

See Figure 1. (T_1 and T_2 consist only of a root node. In order that the assignment of types to nodes will satisfy the inductive construction in Lemma 1 below, we take the convention that the node in T_1 is of type β and the node in T_2 is of type α.)

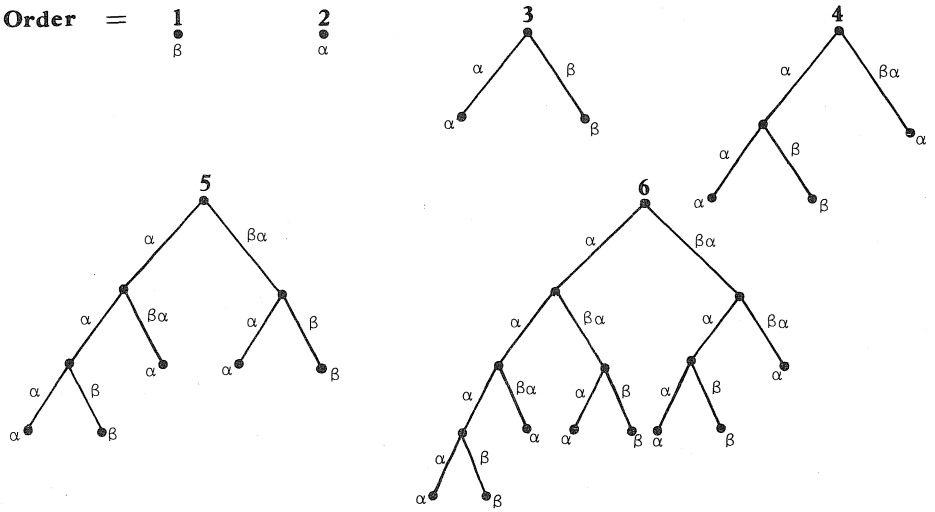

FIGURE 1. Fibonacci Trees (see Section 2 for branch labeling)

Lemma 1

The type determination within each of the left and right subtrees gives the correct type determination for the whole tree.

Proof (induction on order k): Trivially true for T_3. Consider T_k, $k \geqslant 4$, with $c = 2$. The left [right] subtree is T_{k-1} [T_{k-2}], so within this subtree, by (A), the α-nodes have cost $k - 3$ [$k - 4$] and the β-nodes have cost $k - 2$ [$k - 3$]. But in the whole tree, these α-nodes have cost

$$(k - 3) + 1 = k - 2 \quad [(k - 4) + 2 = k - 2],$$

hence, they are still of type α, and these β-nodes have cost

$$(k - 2) + 1 = k - 1 \quad [(k - 3) + 2 = k - 1],$$

hence, they are still of type β. This completes the proof.

Before going to the next section, we show two things. First, let us see that T_k with $c = 2$ has F_{j+1} internal nodes of cost j, $j = 0, 1, \ldots, k - 3$. In fact, T_{j+2} has F_{j+1} nodes of cost j, and they must all be terminal, by (A). Split all these α-nodes, then the resulting tree T_{j+3}, by (B), has F_{j+1} internal nodes of cost j, and so does every Fibonacci tree of order greater than $j + 3$.

Secondly, let us see what happens when we apply the operation "split all α-nodes" $n - 1$ times successively to T_{m+1}. The tree produced is, of course, the Fibonacci tree of order $(m + 1) + (n - 1) = m + n$, by (B). On the other hand, the β-nodes in the original tree of order $m + 1$ will change into α-nodes when the α-nodes in this tree are split to produce the tree of order $m + 2$. Hence, each of the F_m [resp. F_{m-1}] α-nodes [β-nodes] in the original tree of order $m + 1$ will become the root of T_{n+1} [T_n] when the whole process is completed. By counting the terminal nodes, we have obtained a "proof-by-tree" of the well-known relation [4]:

$$F_{m+n} = F_m F_{n+1} + F_{m-1} F_n .$$

2. NUMBER OF TERMINAL NODES AT EACH LEVEL

In this section, we shall show the following:

NOTES ON FIBONACCI TREES AND THEIR OPTIMALITY

Theorem 1

The number of α-nodes at level ℓ of the Fibonacci tree of order $k \geq 2$ is given by $\binom{\ell}{k-2-\ell}$, and the number of β-nodes is given by $\binom{\ell-1}{k-2-\ell}$, $\ell = 0, 1, \ldots, k-2$. [*Remark*: The height (the maximum level) of the Fibonacci tree of order $k \geq 2$ is $k - 2$.]

Before proving this theorem, let us look at the Fibonacci trees more closely with the aid of the following branch labeling. We label (inductively on order k) each branch with one of the three signs, α, $\beta\alpha$, β, as follows: In T_3, the left branch is labeled α, and the right branch is labeled β. Suppose the labeling is already done for T_{k-1} and T_{k-2}. Let these labeled trees be the left and right subtrees of T_k, respectively, and let the left and right branches that are incident to the root of T_k be labeled α and $\beta\alpha$, respectively (see Figure 1). (The branch labeling may have the following "tree-growth" interpretation: Every branching occurs at discrete times $k = 3, 4, \ldots$, and produces two different types of branches α, β. Suppose a branching occurs at time k. The α-branch produced at this time is "ready" for similar branching at time $k + 1$, but the β-branch must "mature" into a $\beta\alpha$-branch at time $k + 1$ to branch at time $k + 2$.) This labeling rule immediately implies that every left branch is labeled α and every right branch not incident to a terminal node of type β is labeled $\beta\alpha$.

Now, by *F-sequence* (called *PM* sequence in [2]), we mean a sequence of α and β with no two β's adjacent. It is easy to see, by induction on order k, that paths (by which we always mean paths from the root to terminal nodes) in T_k correspond, in one-to-one manner, to F-sequences of length $k - 2$ obtained by concatenating branch labels along paths, and that all possible F-sequences of length $k - 2$ appear in T_k; hence, there are F_k F-sequences of length $k - 2$ in all. For example, if we enumerate all paths in T_6 (see Figure 1) "from left to right," we have eight ($=F_6$) F-sequences of length 4: $\alpha\alpha\alpha\alpha$, $\alpha\alpha\alpha\beta$, $\alpha\alpha\beta\alpha$, $\alpha\beta\alpha\alpha$, $\alpha\beta\alpha\beta$, $\beta\alpha\alpha\alpha$, $\beta\alpha\alpha\beta$, $\beta\alpha\beta\alpha$.

Proof of Theorem 1: It is also easy to show, using Lemma 1 and by induction on order k, that any path leading to an α-node [resp. a β-node] corresponds to an F-sequence ending with α [β]. Therefore, the number of

α-nodes at level ℓ of T_k is the number of F-sequences of length $k-2$ ending with α and composed of ℓ α's and $k-2-\ell$ β's. The number of such F-sequences is the number of ways to choose $k-2-\ell$ positions to receive a β from the ℓ starred positions in the alternating sequence $*\alpha*\alpha \ldots *\alpha$. This is $\binom{\ell}{k-2-\ell}$. Similarly, the number of β nodes at level ℓ of T_k is the number of F-sequences of length $k-2$ ending with β and composed of $\ell-1$ α's and $k-1-\ell$ β's. The number of such F-sequences is the number of ways to choose $k-2-\ell$ positions to receive a β from the $\ell-1$ starred positions in the (almost) alternating sequence $*\alpha*\alpha \ldots *\alpha\beta$. This is $\binom{\ell-1}{k-2-\ell}$. This completes the proof.

Note that, since

$$\binom{\ell-1}{k-2-\ell} = \binom{\ell-1}{k-3-(\ell-1)},$$

the number of β-nodes at level $\ell \geq 1$ of the Fibonacci tree of order $k \geq 3$ equals the number of α-nodes at level $\ell - 1$ of the Fibonacci tree of order $k-1$.

Now, let us look at a relation between the numbers of the terminal nodes of each type and some sequences of binomial coefficients appearing in the Pascal triangle. Draw diagonals in the Pascal triangle as shown in Figure 2. It is well known ([2], [4]) that, if we add up the numbers between the parallel lines, the sums are precisely the Fibonacci numbers.

F_1	1							
F_2	1	1						
F_3	1	2	1					
F_4	1	3	3	1				
F_5	1	4	6	4	1			
F_6	1	5	10	10	5	1		
F_7	1	6	15	20	15	6	1	
F_8	1	7	21	35	35	21	7	1
F_9	1	8	28	56	70	56	28	8	1
				. . . . .				

FIGURE 2. Pascal Triangle

We observe that the sequences totalling F_{k-2} and F_{k-1} in the triangle

$$F_{k-2}: \binom{k-3}{0}, \binom{k-3}{1}, \binom{k-3}{2}, \ldots, \binom{\ell-1}{k-2-\ell}, \ldots$$

$$F_{k-1}: \binom{k-2}{0}, \binom{k-2}{1}, \binom{k-2}{2}, \ldots, \binom{\ell}{k-2-\ell}, \ldots$$

$$\text{level} = k-2, \quad k-3, \quad k-4, \quad \ldots, \quad \ell \quad , \ldots$$

display the numbers of the β-nodes and the α-nodes, respectively, at decreasing levels of T_k. For example, we find in Figure 2 that T_{10} has 15 α-nodes and 10 β-nodes at level 6. In [1], the total number of terminal nodes at level ℓ of T_k is also given (with a slightly different interpretation) but not in the form of the sum of two meaningful numbers:

$$\binom{\ell}{k-2-\ell} + \binom{\ell-1}{k-2-\ell}.$$

3. c-OPTIMALITY OF FIBONACCI TREES

Property (C) above states that T_k is 2-optimal for every k. In this section we prove the following.

Theorem 2

When $1 \leqslant c < 2$, the Fibonacci tree of order $k \geqslant 3$ is c-optimal if and only if

$$k \leqslant 2\left\lfloor \frac{1}{2-c} \right\rfloor + 3.$$

When $c > 2$, the Fibonacci tree of order $k \geqslant 3$ is c-optimal if and only if

$$k \leqslant 2\left\lfloor \frac{1}{c-2} \right\rfloor + 4.$$

($\lfloor x \rfloor$ is the largest integer $\leqslant x$.)

To prove the theorem, we first note the following: T_k, $k \geqslant 5$, has the shape shown in Figure 3 and Figure 4, and $k-2$ ($k \geqslant 3$) is the maximum level of T_k, where both α- and β-nodes exist, because from Theorem 1 the maximum level of T_k must be $\leqslant k-2$ and $\ell = k-2$ gives

$$\binom{\ell}{k-2-\ell} = \binom{\ell-1}{k-2-\ell} = 1 \text{ if } k \geq 3.$$

The minimum level where a terminal α-node [resp. β-node] exists is given by

$$\left\lfloor \frac{k-1}{2} \right\rfloor \left[\left\lfloor \frac{k}{2} \right\rfloor \right],$$

the smallest integer ℓ satisfying $k - 2 - \ell \leq \ell$ [$k - 2 - \ell \leq \ell - 1$], from Theorem 1 (see Figures 3 and 4).

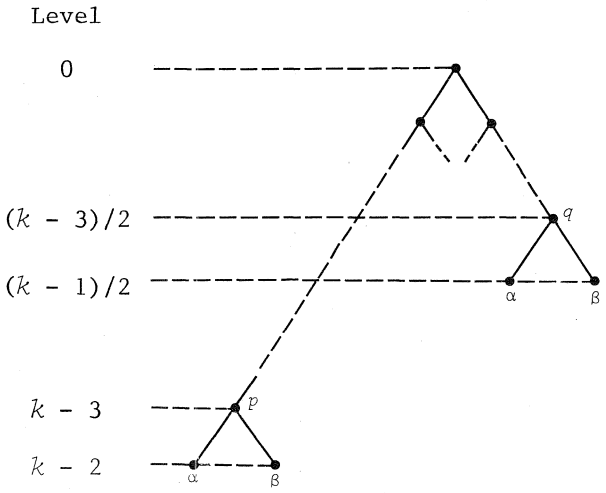

FIGURE 3. Fibonacci Tree of Odd Order $k \geq 5$

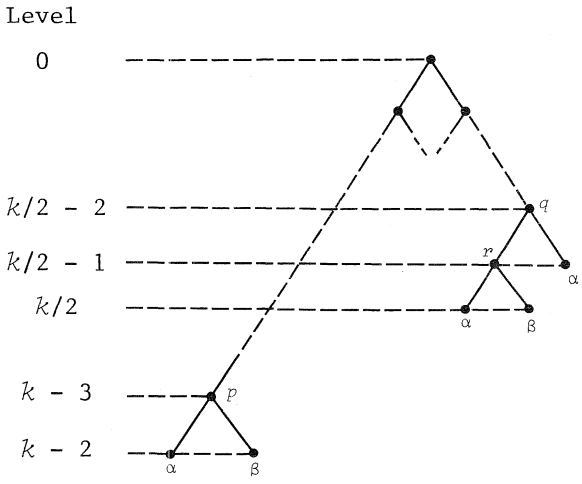

FIGURE 4. Fibonacci Tree of Even Order $k \geq 6$

Proof of the "only if" part of Theorem 2

Trivial for $k = 3, 4$.

Case $1 \leqslant c < 2$, odd $k \geqslant 5$: See Figure 3. Change T_k into a non-Fibonacci tree having F_k terminal nodes by deleting the two sons of the node p and by splitting the left son of the node q. Let us compute the change in the total cost by this transformation. Deletion of the old vertices saves $(k - 3) + (1 + c) = s$. The new vertices add cost

$$1 + c\left(\frac{k - 3}{2}\right) + (1 + c) = t.$$

The net change in cost is

$$t - s = 1 + (c - 2)\left(\frac{k - 3}{2}\right).$$

If T is c-optimal, we must have $t - s \geqslant 0$, so

$$\frac{k - 3}{2} \leqslant \frac{1}{2 - c} \quad \text{or} \quad k \leqslant \frac{2}{2 - c} + 3.$$

Case $1 \leqslant c < 2$, even $k \geqslant 6$: See Figure 4. Change T_k into a non-Fibonacci tree having F_k terminal nodes by deleting the two sons of the node p and by splitting the right son of the node q. Again, if t is the added cost of the new vertices and s the savings from deleting old vertices, we have $s = (k - 3) + (1 + c)$, $t = 1 + c(k/2)$, so

$$t - s = 1 + (c - 2)\left(\frac{k - 2}{2}\right).$$

If T_k is c-optimal, we must have $t - s \geqslant 0$, so

$$\frac{k - 2}{2} \leqslant \frac{1}{2 - c} \quad \text{or} \quad k \leqslant \frac{2}{2 - c} + 2.$$

The conditions $k \leqslant \frac{2}{2 - c} + 3$ for k odd and $k \leqslant \frac{2}{2 - c} + 2$ for k even can be combined to get

$$k \leqslant 2\left\lfloor \frac{1}{2 - c} \right\rfloor + 3.$$

Case $c > 2$, odd $k \geqslant 5$: See Figure 3. Change T_k into a non-Fibonacci tree having F_k terminal nodes by deleting the two sons of the node q and

by splitting the left son of the node p. Here

$$s = 1 + c\left(\frac{k-1}{2}\right),\ t = 1 + (k - 2 + c),\ t - s = (2 - c)\left(\frac{k-3}{2}\right) + 1.$$

Fibonacci c-optimality requires $t - s \geq 0$, so

$$\frac{k-3}{2} \leq \frac{1}{c-2} \quad \text{or} \quad k \leq \frac{2}{c-2} + 3.$$

Case $c > 2$, even $k \geq 6$: See Figure 4. Change T_k into a non-Fibonacci tree having F_k terminal nodes by deleting the two sons of the node r and by splitting the left son of the node p. Here

$$s = 1 + c\left(\frac{k}{2} - 2\right) + (1 + c),\ t = 1 + (k - 2) + c,$$

$$t - s = (2 - c)\left(\frac{k-4}{2}\right) + 1.$$

Fibonacci c-optimality requires $t - s \geq 0$, so

$$\frac{k-4}{2} \leq \frac{1}{c-2} \quad \text{or} \quad k \leq \frac{2}{c-2} + 4.$$

The conditions $k \leq \frac{2}{c-2} + 3$ for k odd and $k \leq \frac{2}{c-2} + 4$ for k even can be combined to get

$$k \leq 2\left\lfloor \frac{1}{c-2} \right\rfloor + 4.$$

Our proof of the "if" part of the theorem will be based on the next lemma.

Lemma 2

Denote by $\alpha(k, \ell, c)$ and $\beta(k, \ell, c)$ the costs of the α-nodes and the β-nodes at level ℓ of the Fibonacci tree of order $k \geq 3$ with cost c for right branches. Then we have:

$$\alpha(k, \ell, c) = (2 - c)\ell + (c - 1)(k - 2),$$

$$\beta(k, \ell, c) = (2 - c)\ell + (c - 1)(k - 1).$$

Proof: Obviously, $\alpha(k, \ell, 1) = \beta(k, \ell, 1) = \ell$. By (A), we have

$$\alpha(k, \ell, 2) = k - 2,\ \beta(k, \ell, 2) = k - 1.$$

Since $(2 - c)(1, 1) + (c - 1)(1, 2) = (1, c)$, i.e., the cost assignment $(1, c)$ to (left branch, right branch) may be written as this linear combination of two cost assignments $(1, 1)$ and $(1, 2)$, the proof is finished.

Proof of the "if" part of Theorem 2

Case $1 \leqslant c < 2$: Put

$$k^* = 2\left\lfloor \frac{1}{2 - c} \right\rfloor + 2.$$

We show that, for every $k \leqslant k^*$,

(1) $\alpha(k, k - 2, c) \leqslant \beta\left(k, \left\lfloor \frac{k}{2} \right\rfloor, c\right)$,

(2) $\alpha(k, k - 2, c) \leqslant \beta\left(k, \left\lfloor \frac{k - 1}{2} \right\rfloor, c\right) + 1$.

To show (1) [(2) and (3) and (4) below can be verified similarly), consider the difference:

$$D = \beta\left(k, \left\lfloor \frac{k}{2} \right\rfloor, c\right) - \alpha(k, k - 2, c).$$

If k is even, we have, using Lemma 2 and $k \leqslant k^*$,

$$D = (2 - c)\left(\frac{k}{2}\right) + (c - 1)(k - 1) - (k - 2)$$

$$= -(2 - c)\left(\frac{k - 2}{2}\right) + 1 \geqslant -(2 - c)\left\lfloor \frac{1}{2 - c} \right\rfloor + 1 \geqslant 0.$$

If k is odd, we have, using Lemma 2 and $k \leqslant k^* - 1$ (note that k^* is even),

$$D = (2 - c)\left(\frac{k - 1}{2}\right) + (c - 1)(k - 1) - (k - 2)$$

$$= -(2 - c)\left(\frac{k - 1}{2}\right) + 1 \geqslant -(2 - c)\left\lfloor \frac{1}{2 - c} \right\rfloor + 1 \geqslant 0.$$

Now, let us remember the remarks given just before the proof of the "only if" part. By Lemma 2, $\alpha(k, \ell, c)$ and $\beta(k, \ell, c)$ increase linearly in ℓ, so (1) implies that all α-nodes in T_k, $k \leqslant k^*$, are the cheapest of all terminal nodes. The inequality (2) implies that, if the cheapest α-node—its cost is $\alpha\left(k, \left\lfloor \frac{k-1}{2} \right\rfloor, c\right)$—is split, the cost $\alpha\left(k, \left\lfloor \frac{k-1}{2} \right\rfloor, c\right) + 1$

of its left son will never be less than the highest cost $\alpha(k, k-2, c)$ of all α-nodes. This means that the successive applications (F_{k-1} times) of Varn's procedure mentioned in the first section will result in splitting all α-nodes of T_k. Hence, if this tree of order k is c-optimal, the resulting tree, which is T_{k+1} by (B), is also c-optimal. Since T_3 is c-optimal and $k^* \geq 3$, we conclude, inductively, that T_k is c-optimal for every $k \leq k^* + 1$.

Case $c > 2$: Put $k^* = 2\left\lfloor \frac{1}{c-2} \right\rfloor + 3$. We have, for every $k \leq k^*$,

$$(3)\quad \alpha\left(k, \left\lfloor \frac{k-1}{2} \right\rfloor, c\right) \leq \beta(k, k-2, c),$$

$$(4)\quad \alpha\left(k, \left\lfloor \frac{k-1}{2} \right\rfloor, c\right) \leq \alpha(k, k-2, c) + 1.$$

The remainder of the proof is similar to Case $1 \leq c < 2$. Note in this case that $\alpha(k, \ell, c)$ and $\beta(k, \ell, c)$ decrease linearly in ℓ by Lemma 2. ■

REFERENCES

1. M. Agu & Y. Yokoi. "On the Evolution Equations of Tree Structures." (Submitted.)
2. G. Berman & K. D. Fryer. *Introduction to Combinatorics*. New York: Academic Press, 1972.
3. Y. Horibe. "An Entropy View of Fibonacci Trees." *The Fibonacci Quarterly* 20, no. 2 (1982):168-78.
4. D. Knuth. *Fundamental Algorithms*. New York: Addison-Wesley, 1968.
5. P. S. Stevens. *Patterns in Nature*. Boston: Atlantic Monthly Press/ Little, Brown and Company, 1974.
6. B. Varn. "Optimal Variable Length Code (Arbitrary Symbol Cost and Equal Code Word Probability)." *Information and Control* 19 (1971): 289-301.

◆◇◆◇◆