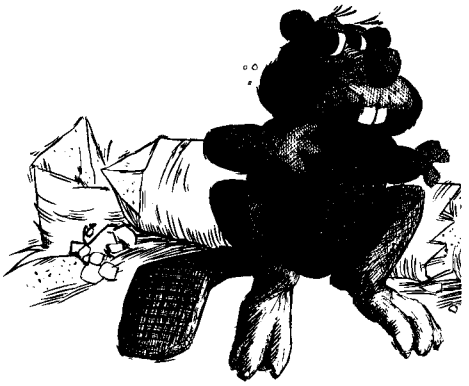
UNIVERSITY OF WATERLOO
COMPUTER SCIENCE DEPARTMENT



Unbalanced AVL Trees and Space Pessimal Brother Trees

Helen Cameron
and
Derick Wood

Data Structuring Group
Research Report CS-89-20

May, 1989

Unbalanced AVL Trees and Space Pessimal Brother Trees *

Helen Cameron† Derick Wood†

Abstract

We characterize a family of AVL trees that have the maximum numbers of unbalanced nodes, nodes whose subtrees differ in height by one, for their heights and weights. We obtain the result from a characterization of brother trees with the largest space costs for their heights and weights.

1 Introduction

In this paper we examine a cost measure specific to AVL and brother trees. In AVL trees, the heights of subtrees rooted at sibling nodes can differ by at most one. (If the two sibling subtrees do not have the same height, then the parent of the two roots is said to be *unbalanced.*) This restriction is designed to keep the search tree from being skewed too far from height $\lceil \log_2(N+1) \rceil$, the height of trees of size N with optimal comparison cost. It is natural to ask, therefore, how many unbalanced nodes an AVL tree of size N can have.

In [OPR+84], the fewest number of unbalanced nodes that an AVL tree of size N can have is shown to be equal to the number of zeros in the binary representation of N. But to know whether any effort should be expended to achieve this minimal number, we need to know how bad things can get. Certainly, if the worst situation is not much worse than the best situation, it would not be worth while achieving and maintaining the best situation. Thus, examining pessimally balanced AVL trees is a natural extension of the study of optimally balanced AVL trees.

Unfortunately, characterizing pessimal comparison cost AVL trees seems to be a hard problem (see [KW89b] and [KW89a]). For each size, however, there is an AVL tree that is comparison-cost optimal and also contains

*This work was supported under a Natural Sciences and Engineering Research Council of Canada Grant No. A-5692 and under a grant from the Information Technology Research Centre.

†Data Structuring Group, Department of Computer Science, University of Waterloo, WATERLOO, Ontario N2L 3G1, CANADA

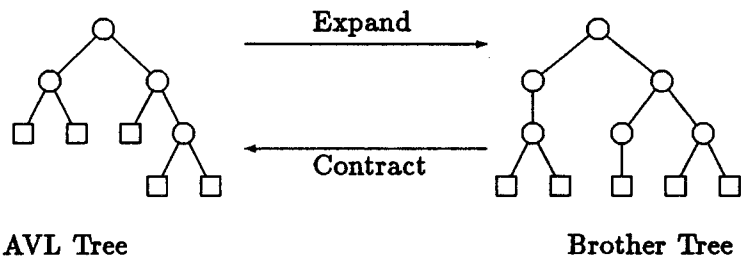


Figure 1: A bijection between AVL and brother trees.

the optimal number of unbalanced nodes, see [OPR+84]. Thus, the number of unbalanced nodes in an AVL tree is a measure of skewness from a comparison-cost optimal AVL tree. The characterization of pessimally balanced AVL trees may lead to new results on pessimal comparison-cost AVL trees.

In [OW80], a bijection is given between the class of AVL trees and the class of brother trees. An AVL tree can be "expanded" into a brother tree by placing a unary node between each unbalanced node and the shorter of its two children. The inverse operation of "contraction" turns a brother tree into an AVL tree by removing all unary nodes and making the child of a unary node the child of the parent of the unary node. The two operations are pictured in Figure 1. Clearly, the number of unbalanced nodes in an AVL tree, T, is exactly equal to the number of unary nodes in the corresponding brother tree, *expand*(T), and the number of internal binary nodes, the *size*, is the same. Thus, we can answer our question about the maximum number of unbalanced nodes that can appear in AVL trees of size N by finding the maximum number of unary nodes that can appear in the corresponding brother trees, the brother trees of size N. This is, in fact, what we do; we analyze the class of brother trees.

The space cost of a brother tree, T, is the number of its internal nodes. So the number of unary nodes in T is its space cost less its size. We wish to characterize the space cost pessimal (SCP) brother trees among all brother trees of size N. Instead, for each possible height h for a brother tree of size N, we first find the maximum space cost for a brother tree of height h and size N. (A brother tree with maximum space cost for its height and size is called a *locally SCP brother tree.*) Then, to find the height of SCP brother trees of size N, we must compare space costs.

In Section 2, we define brother trees, the detailed profile of a brother tree, and the space cost of a brother tree. In Section 3, we characterize a family of locally SCP brother trees that contains a tree with each possible height and weight combination. In Section 4, we translate the characterization of a locally SCP brother tree to the characterization of an AVL tree with the

maximum number of unbalanced nodes for its height and weight. Section 5 concludes with some open problems.

2 Brother trees

If a node has at least one child, it is *internal*; otherwise it is *external.*

Definition 2.1 *A* brother tree *is a tree that satisfies the following conditions:*

- *Each internal node has either one or two children;*
- *Each unary node has a binary sibling;*
- *All root-to-leaf paths have the same length.*

The *weight* of a brother tree, T, is denoted by $wt(T)$ and is defined as the number of external nodes. The *height* of a brother tree, T, is denoted by $ht(T)$ and is defined as the length of the longest root-to-leaf path. For brother trees, this is the length of any root-to-leaf path because all root-to-leaf paths have the same length. The *level* of a node p in brother tree, T, is its distance from the root of T; the root of T is at level 0, its children are at level 1, and so on.

Profiles or detailed profiles, defined below, are used to describe brother trees. The two types of profiles are equivalent, but we will normally use the detailed profile because it provides explicitly the numbers of unary and binary nodes on each level.

Definition 2.2 *The* profile *of a brother tree, T, of height h is the integer sequence $\pi(T) = \nu_0, \ldots, \nu_h$, where ν_i is the number of nodes on level i.*

The detailed profile *of a brother tree, T, of height h and weight w is the sequence of integer pairs $\Delta(T) = \langle \omega_0, \beta_0 \rangle, \ldots, \langle \omega_h, \beta_h \rangle$, where ω_i (β_i) is the number of unary (binary) nodes on level i, for $0 \leq i \leq h$. By convention, external nodes are considered binary nodes; that is, $\omega_h = 0$ and $\beta_h = w$.*

The relationship between the two types of profiles is given by $\nu_i = \beta_i + \omega_i$.

Using the following proposition, we can distinguish between the sequences of integer pairs that are the detailed profiles of brother trees and those that are not.

Proposition 2.1 *Let $\Delta = \langle \omega_0, \beta_0 \rangle, \ldots, \langle \omega_h, \beta_h \rangle$ be a sequence of integer pairs. Then, Δ is the detailed profile of a brother tree if and only if*

- $\beta_0 = 1$,

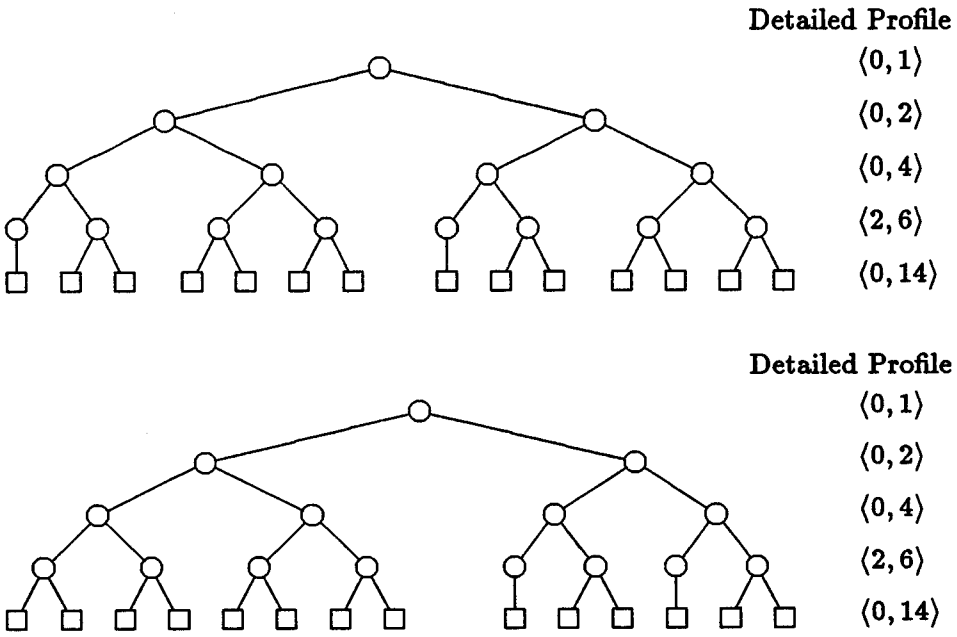


Figure 2: Two distinct brother trees with the same profile.

- $\beta_i \geq 0$ *and* $\omega_i \geq 0$, *for* $0 \leq i \leq h$,
- $\beta_i \geq \omega_{i+1}$, *for* $0 \leq i < h$,
- $\nu_{i+1} = \omega_i + 2\beta_i$, *for* $0 \leq i < h$.

Proof: See [OPR+84]. □

A profile or detailed profile is a description of a set of brother trees, all with the same height, the same weight, and the same numbers of unary and binary nodes on each level. The difference between the trees is the positions of the unary and binary nodes on each level. For example, the two brother trees in Figure 2 have the same profile but are clearly not the same tree. There are, in fact, 24 brother trees with that profile. Most descriptions of brother trees in this paper will not include the positions of nodes on the levels of the trees. In most instances, we could replace the phrase "the brother tree T" with the phrase "the set of brother trees with the same profile as brother tree T", except where the positions of nodes are important.

Definition 2.3 *The* space cost *of a brother tree, T, is denoted by* $\mathrm{SCost}(T)$ *and is defined to be the number of internal nodes of the tree, or*

$$\mathrm{SCost}(T) = \sum_{i=0}^{h-1} \nu_i.$$

A space-cost pessimal (SCP) brother tree *is a brother tree with maximum space cost among all brother trees of the same weight. Since all brother trees of the same weight have the same number of internal binary nodes, a SCP brother tree has the maximum number of unary nodes for its weight.*

To characterize SCP brother trees, we will examine the brother trees with maximum space cost for each weight and height combination.

Definition 2.4 *A brother tree, T, with the maximum space cost for its weight and height is* locally SCP.

3 The Characterization

In this section, we characterize a family of locally SCP brother trees, the F^+ trees. In the first subsection, the terms used in the description of F^+ trees are defined, and we show that a description using these terms is equivalent to a detailed profile. Also, the family of F^+ trees is defined. In the second subsection, we prove that one of the requirements for membership in the family of F^+ trees is satisfied by all locally SCP brother trees. In the third subsection, we show that there is a locally SCP F^+ tree with each possible height and weight combination. In the last subsections, we characterize a locally SCP F^+ tree, given a height and a weight.

3.1 Definitions

We will describe a locally SCP brother tree of height h and weight w using the following terms.

Definition 3.1 *A brother tree, T, where $\Delta(T) = \langle \omega_0, \beta_0 \rangle, \ldots, \langle \omega_h, \beta_h \rangle$, has a* complete binary prefix of height p *(a $Bin(p)$ prefix), for some $1 \leq p \leq h$, if $\omega_i = 0$ and $\beta_i = 2^i$, for $0 \leq i < p$.*

The brother tree, T, has a maximum complete binary prefix of height p *($maxBin(T) = p$), for some $1 \leq p \leq h$, if T has a $Bin(p)$ prefix but not a $Bin(p+1)$ prefix.*

Every non-empty brother tree has a $Bin(1)$ prefix, because the root of a non-empty brother tree is binary. Thus, each brother tree, T, satisfies $1 \leq maxBin(T) \leq ht(T)$.

Definition 3.2 *A* surplus node *is a binary node with two binary children.*

All binary nodes on level $ht(T) - 1$ of any brother tree T are surplus nodes because the external nodes on level $ht(T)$ are defined to be binary nodes.

A unary node on level $i+1$ of a brother tree must have a binary parent on level i and can not have a unary sibling. Thus, the number of binary nodes on level i that are not surplus nodes is ω_{i+1}, and level i contains exactly s surplus nodes if and only if $\beta_i = \omega_{i+1} + s$.

Given the height h of a brother tree, T, the height p of the maximum complete binary prefix of T, and the number of surplus nodes on each of the levels $p-1$ to $h-2$, we show that the detailed profile of T is completely specified. Thus, it is sufficient to find these values to completely describe a locally SCP brother tree of a given height and weight.

Lemma 3.1 *Let T be a brother tree with height h, $maxBin(T) = p$, and γ_i surplus nodes on level i, for $p-1 \leq i \leq h-2$. Then, given h, p, and γ_i, for $p-1 \leq i \leq h-2$, the detailed profile of T can be completely specified.*

Proof: Since $maxBin(T) = p$, we have $\beta_j = 2^j$ and $\omega_j = 0$, for $0 \leq j < p$. Since there are γ_{p-1} surplus nodes on level $p-1$, we have $\beta_{p-1} = \omega_p + \gamma_{p-1}$. Also, $\beta_p = \omega_{p-1} + \beta_{p-1} + \gamma_{p-1}$, since each node on level $p-1$ has one binary child on level p, except for surplus nodes on level $p-1$, which have two binary children on level p. Since γ_{p-1}, β_{p-1}, and ω_{p-1} are known, we can compute β_p and ω_p. Similarly, β_i and ω_i can be computed from β_{i-1} and ω_{i-1} and γ_{i-1}, for $p+1 \leq i \leq h-1$. Finally, $\beta_h = \omega_{h-1} + 2 \cdot \beta_{h-1}$ and $\omega_h = 0$, since all external nodes are considered binary nodes. □

We have defined surplus nodes and the maximum complete binary prefix of a brother tree, and we have shown, in Lemma 3.1, how brother trees can be characterized using these terms. We will now describe a class of brother trees, called F^+ trees. Each tree in the class has restrictions placed on the numbers and positions of surplus nodes beneath the maximum complete binary prefix. The name F^+ tree means "Fibonacci plus" tree and was chosen because these trees are Fibonacci-like beneath the binary prefix, with some extra un-Fibonacci-like nodes. (See Section 3.4 for the definition of Fibonacci trees.) Our goal is to prove that there exists a locally SCP brother tree, T, that is also an F^+ tree, for each possible weight and height. We also wish to calculate $maxBin(T)$ and the number and positions of surplus nodes in T beneath the binary prefix of height $maxBin(T)$, given the height and weight of T.

Definition 3.3 *A brother tree, T, is an F^+ tree if*

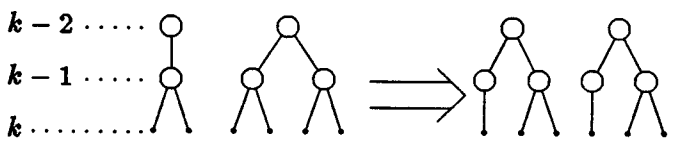


Figure 3: A space-cost increasing transformation.

1. *For* $maxBin(T) \leq i < ht(T)$, *if level* i *contains a surplus node, then level* $i+1$ *does not;*

2. *For* $maxBin(T) \leq i < ht(T) - 1$, *level* i *contains at most one surplus node.*

3.2 Adjacent Levels of Surplus Nodes

In this subsection, we will show that Restriction 1 in the definition of F^+ trees applies to *all* locally SCP brother trees, not just to those locally SCP brother trees that are also F^+ trees.

First, we need the space-cost increasing transformation described in the following lemma. (The transformation is displayed in Figure 3.)

Lemma 3.2 *Let* $\Delta = \langle \omega_0, \beta_0 \rangle, \ldots, \langle \omega_h, \beta_h \rangle$ *be the detailed profile of a locally SCP brother tree,* T. *Then, for each* k, *where* $3 \leq k \leq h$, *at least one of the following must hold:*

$$\omega_{k-2} = 0, \tag{1}$$
$$\beta_{k-2} = \omega_{k-1}, \tag{2}$$
$$\beta_{k-1} = \omega_k. \tag{3}$$

Proof: Let $\Delta = \langle \omega_0, \beta_0 \rangle, \ldots, \langle \omega_h, \beta_h \rangle$ be the detailed profile of a locally SCP brother tree, T. Assume that for some k, where $3 \leq k \leq h$, none of conditions (1), (2), or (3) holds. Consider the sequence

$$\begin{aligned} \Delta' = \; & \langle \omega_0, \beta_0 \rangle, \ldots, \\ & \langle \omega_{k-2} - 1, \beta_{k-2} + 1 \rangle, \langle \omega_{k-1} + 2, \beta_{k-1} - 1 \rangle, \langle \omega_k, \beta_k \rangle, \\ & \ldots, \langle \omega_h, \beta_h \rangle. \end{aligned}$$

Δ' is the detailed profile of some brother tree, T'; see [OPR+84]. Now, T' has the same height and weight as T, and $SCost(T') > SCost(T)$. This contradicts the local space cost pessimality of T. □

Corollary 3.3 *Let $\Delta = \langle \omega_0, \beta_0 \rangle, \ldots, \langle \omega_h, \beta_h \rangle$ be the detailed profile of a locally SCP brother tree, T. If there exists an integer j, where $0 < j < h$, such that $\beta_j > \omega_{j+1}$ and $\beta_{j-1} > \omega_j$, then $\omega_{j'} = 0$, for $0 \leq j' \leq j-1$.*

Proof: See Corollary 3.3 in [OPR+84]. □

Corollary 3.4 *Let T be a locally SCP brother tree and $maxBin(T) = p$. If level i contains a surplus node, then level $i+1$ does not, for $p \leq i < h-1$.*

Proof: Let $maxBin(T) = p$, and let level i contain a surplus node u, for some $p \leq i < h-1$. Since each binary node on level i can have at most one unary child on level $i+1$ (otherwise T is not a brother tree) and surplus node u only has binary children, we must have $\beta_i > \omega_{i+1}$. Assume that level $i+1$ also contains a surplus node; then, by the same argument, $\beta_{i+1} > \omega_{i+2}$. By Corollary 3.3, since $\beta_i > \omega_{i+1}$ and $\beta_{i+1} > \omega_{i+2}$, we must have $\omega_j = 0$, for $0 \leq j \leq i$. Therefore, T has a $Bin(i+1)$ prefix. This is a contradiction, since $p \leq i$. Therefore, level $i+1$ does not contain a surplus node. □

Because all external nodes are defined to be binary, each binary node on level $h-1$ is a surplus node. There must be at least one binary node on level $h-1$, if $h \geq 1$. Thus, by Corollary 3.4, if $h > 2$ and $h > p+1$, there cannot be surplus nodes on level $h-2$.

By Corollary 3.4, every locally SCP brother tree satisfies Restriction 1 in the definition of F^+ trees. In the next subsection, we show that there exists a locally SCP brother tree, for each height and weight, that also satisfies the second restriction.

3.3 A Family of Locally SCP Brother Trees

In this subsection, we transform any locally SCP brother tree, T, of height h and weight w, into another locally SCP brother tree, T', of the same height and weight such that T' is an F^+ tree. We must "spread out" the surplus nodes on levels $maxBin(T)$ to $ht(T)-2$ so that none of these levels contains more than one surplus node. To do this, we need the space-cost preserving transformation described in the following lemma. It transforms a locally SCP brother tree, T, into another locally SCP brother tree, T', of the same height and weight. We will see that this transformation moves surplus nodes, possibly increasing the height of the maximum complete binary prefix.

Lemma 3.5 *Let $\Delta = \langle \omega_0, \beta_0 \rangle, \ldots, \langle \omega_h, \beta_h \rangle$ be the detailed profile of a brother tree, T, with $maxBin(T) = p$. If $\omega_{k-1} \leq \beta_{k-2} - 2$, for some k, where $p+2 \leq k < h$, then define the detailed profile, $\Delta' = \langle \omega'_0, \beta'_0 \rangle, \ldots, \langle \omega'_h, \beta'_h \rangle$, by*

- $\omega'_i = \omega_i$ *and* $\beta'_i = \beta_i$, *for* $0 \leq i \leq k-3$ *and* $k+1 \leq i \leq h$,
- $\omega'_{k-2} = \omega_{k-2} - 1$ *and* $\beta'_{k-2} = \beta_{k-2} + 1$,
- $\omega'_{k-1} = \omega_{k-1} + 3$ *and* $\beta'_{k-1} = \beta_{k-1} - 2$, *and*
- $\omega'_k = \omega_k - 2$ *and* $\beta'_k = \beta_k + 1$.

Δ' is also the detailed profile of a brother tree, T', with the same weight, height, and space cost as T. In particular, if T is locally SCP, then T' is locally SCP.

Proof: Note that neither the height nor the weight nor the space cost is changed by the transformation given in the statement of the lemma. Thus, if T is locally SCP and Δ' is the profile of a brother tree, T', then T' has the same weight and height as T and is also locally SCP.

By Proposition 2.1, to prove that Δ' is the profile of a brother tree, it is necessary to prove that

1. $\beta'_0 = 1$,
2. $\beta'_i \geq 0$ and $\omega'_i \geq 0$, for $0 \leq i \leq h$,
3. $\beta'_i \geq \omega'_{i+1}$, for $0 \leq i < h$, and
4. $\nu'_{i+1} = \omega'_i + 2 \cdot \beta'_i$, for $0 \leq i < h$.

Δ is the profile of a brother tree so these conditions hold for Δ. Δ' differs from Δ only on levels $k-2$, $k-1$, and k, so most of these conditions already hold for Δ' also.

We have $\beta_i \geq 0$ and $\omega_i \geq 0$, for $0 \leq i \leq h$. Thus, trivially, $\beta'_{k-2} \geq 0$, $\omega'_{k-1} \geq 0$, and $\beta'_k \geq 0$. Since $\beta_{k-2} \geq \omega_{k-1} + 2$, there are at least two surplus nodes on level $k-2$. Since there are surplus nodes on level $k-2 \geq p$ and Δ is the profile of a locally SCP brother tree, by Corollary 3.4, there are no surplus nodes on level $k-1$; therefore, $\omega_k = \beta_{k-1}$. Also, $\beta_{k-1} \geq 4$, since there are at least two surplus nodes on level $k-2$. Thus, $\omega'_k = \omega_k - 2 = \beta_{k-1} - 2 \geq 4 - 2 > 0$ and $\beta'_{k-1} = \beta_{k-1} - 2 \geq 4 - 2 > 0$.

If $k - 2 > p$, then, by Corollary 3.4, since there are surplus nodes on level $k-2$, there are no surplus nodes on level $k-3$. Thus, $\beta_{k-3} = \omega_{k-2}$. Since $k - 3 \geq p \geq 1$, we have $\beta_{k-3} \geq 1$. Thus, $\omega'_{k-2} = \omega_{k-2} - 2 = \beta_{k-3} - 1 \geq 1 - 1 = 0$, if $k - 2 > p$. Otherwise $k - 2 = p$ and $\omega_{k-2} \geq 1$, because $maxBin(T) = p$. Thus, $\omega'_{k-2} = \omega_{k-2} - 1 \geq 0$. Hence, condition 2 holds for all levels.

Since $\beta_{k-2} \geq \omega_{k-1} + 2$, we have $\beta'_{k-2} = \beta_{k-2} + 1 \geq \omega_{k-1} + 3 = \omega'_{k-1}$, and condition 3 holds for Δ'.

Finally, condition 4, for $i = k - 3$,

$$\begin{aligned}
\omega'_{k-3} + 2 \cdot \beta'_{k-3} &= \omega_{k-3} + 2 \cdot \beta_{k-3} \\
&= \omega_{k-2} + \beta_{k-2} \\
&= (\omega_{k-2} - 1) + (\beta_{k-2} + 1) \\
&= \omega'_{k-2} + \beta'_{k-2} \\
&= \nu'_{k-2}.
\end{aligned}$$

For $i = k - 2$,

$$\begin{aligned}
\omega'_{k-2} + 2 \cdot \beta'_{k-2} &= (\omega_{k-2} - 1) + 2 \cdot (\beta_{k-2} + 1) \\
&= (\omega_{k-2} + 2 \cdot \beta_{k-2}) + 1 \\
&= (\omega_{k-1} + \beta_{k-1}) + 1 \\
&= (\omega_{k-1} + 3) + (\beta_{k-1} - 2) \\
&= \omega'_{k-1} + \beta'_{k-1} \\
&= \nu'_{k-1}.
\end{aligned}$$

For $i = k - 1$,

$$\begin{aligned}
\omega'_{k-1} + 2 \cdot \beta'_{k-1} &= (\omega_{k-1} + 3) + 2 \cdot (\beta_{k-1} - 2) \\
&= (\omega_{k-1} + 2 \cdot \beta_{k-1}) - 1 \\
&= (\omega_k + \beta_k) - 1 \\
&= (\omega_k - 2) + (\beta_k + 1) \\
&= \omega'_k + \beta'_k \\
&= \nu'_k.
\end{aligned}$$

For $i = k$,

$$\begin{aligned}
\omega'_k + 2 \cdot \beta'_k &= (\omega_k - 2) + 2 \cdot (\beta_k + 1) \\
&= \omega_k + 2 \cdot \beta_k \\
&= \omega_{k+1} + \beta_{k+1} \\
&= \omega'_{k+1} + \beta'_{k+1} \\
&= \nu'_{k+1}.
\end{aligned}$$

Thus, condition 4 holds for Δ', and Δ' is the detailed profile of a locally SCP brother tree of the same weight and height as T. □

The space-cost preserving transformation in the above Lemma can be viewed as a transformation on surplus nodes; see Figure 4. The original tree, T, has at least two surplus nodes on level $k - 2$, where $p \leq k - 2 <$

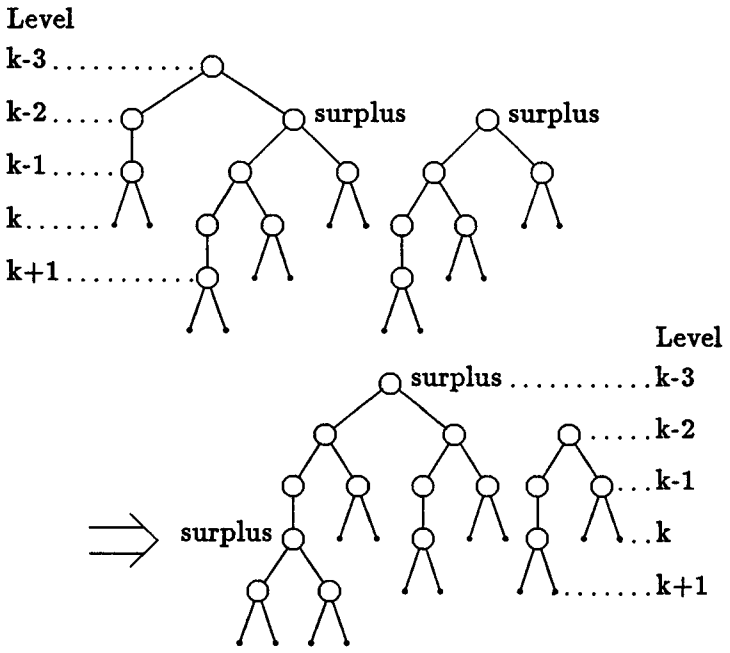


Figure 4: A space-cost preserving transformation.

$h-2$. The resulting tree, T', has one new surplus node on level $k-3$, one new surplus node on level k, and two fewer surplus nodes on level $k-2$. Note that if $k-2=p$ and T has $2^{p-1}-1$ surplus nodes on level $p-1$, then $maxBin(T')=p+1$. (T' does not have a $Bin(p+2)$ prefix because $\omega'_{p+1}=\omega_{p+1}+3$; that is, level $p+1$ in T' is not completely binary.)

The space-cost preserving transformation in Lemma 3.5 allows us to draw the following two conclusions about the numbers and positions of surplus nodes beneath the maximum complete binary prefix in any locally SCP brother tree (not just locally SCP F^+ trees).

Corollary 3.6 *Let $\Delta=\langle\omega_0,\beta_0\rangle,\ldots,\langle\omega_h,\beta_h\rangle$ be the detailed profile of a locally SCP brother tree, T, with $maxBin(T)=p$ and at least two surplus nodes on level $k-2$, where $p+2\leq k<h$. Let T' be the locally SCP brother tree obtained from T by applying the space-cost preserving transformation of Lemma 3.5. Then, neither T nor T' have a surplus node on level $k-1$.*

Proof: Since T has at least two surplus nodes on level $k-2$, where $p\leq k-2<h$, by Corollary 3.4, T cannot have surplus nodes on level $k-1$. Thus, $\omega_k=\beta_{k-1}$. In $\Delta(T')$, $\beta'_{k-1}=\beta_{k-1}-2=\omega_k-2=\omega'_k$. Therefore, T' does not have any surplus nodes on level $k-1$ either. □

Corollary 3.7 *Let $\Delta = \langle \omega_0, \beta_0 \rangle, \ldots, \langle \omega_h, \beta_h \rangle$ be the detailed profile of a locally SCP brother tree, T, with at least two surplus nodes on level $k-2$, where $maxBin(T) + 2 < k < h$. Then, T has exactly two surplus nodes on level $k-2$.*

Proof: Let T' be the locally SCP brother tree constructed from T using the space-cost preserving transformation in Lemma 3.5. Assume T has more than two surplus nodes on level $k-2 > maxBin(T)$. Then, T' has one surplus node on level $k-3 \geq maxBin(T)$ and at least one surplus node on level $k-2$. But Corollary 3.4 forbids this; therefore, T has at most two surplus nodes on level $k-2$. □

We now want to show how the space-cost preserving transformation of Lemma 3.5 can be used to create a locally SCP tree, T', with at most one surplus node on each of the levels $maxBin(T')$ to $ht(T') - 2$. To do this, we first show how the transformation can be used to push surplus nodes down the tree so that the levels beneath the binary prefix that contain more than one surplus node get closer and closer to the external nodes.

Definition 3.4 *Let T be a locally SCP brother tree. Define $g(T)$ to be the smallest value i such that $maxBin(T) \leq i < ht(T) - 1$ and level i contains at least two surplus nodes. If no such i exists, then let $g(T) = ht(T)$.*

The function $g(T)$ indicates where we must first apply the space-cost preserving transformation to turn a locally SCP brother tree T into a locally SCP F^+ tree.

Lemma 3.8 *Let T be a locally SCP brother tree such that $g(T) = j < ht(T)$. Then, there exists a locally SCP brother tree, T', with the same height and weight as T, such that $g(T') > g(T)$.*

Proof: Let $ht(T) = h$, and let $maxBin(T) = p$. Then, $p \leq j < h - 1$, by the definition of $g(T)$. Now, level $h-2$ does not contain surplus nodes unless $p > h-2$, so $j \neq h-2$. Therefore, $p \leq j < h-2$.

- $p < j$.

  Then, by Corollary 3.7, T has exactly two surplus nodes on level j. By Corollary 3.4, T does not have any surplus nodes on levels $j-1$ and $j+1$.

  Applying the space-cost preserving transformation of Lemma 3.5 with $k - 2 = j$, we obtain a locally SCP brother tree, T', with the same height, weight, and space cost as tree T. T' has exactly one surplus node on level $j-1$, no surplus nodes on levels j and $j+1$, and one new surplus node on level $j+2$. Moreover, T' has a $maxBin(p)$ prefix. Thus, $g(T') \geq j + 2 > g(T)$.

- $p = j$, tree T has $i \geq 2$ surplus nodes on level p, and T has $2^{p-1} - l$ surplus nodes on level $p-1$, where $l > \lfloor \frac{i}{2} \rfloor$.

  Applying the space-cost preserving transformation of Lemma 3.5 $\lfloor \frac{i}{2} \rfloor$ times with $k - 2 = p$, we obtain a locally SCP brother tree, T', with the same height and weight as T. T' has $\lfloor \frac{i}{2} \rfloor$ new surplus nodes on level $p-1$, at most one surplus node on level p, no surplus nodes on level $p+1$, and $\lfloor \frac{i}{2} \rfloor$ new surplus nodes on level $p+2$. Since T' has $2^{p-1} - l + \lfloor \frac{i}{2} \rfloor < 2^{p-1}$ surplus nodes on level $p-1$, level p is not completely binary; thus, $maxBin(T') = p$. Therefore, $g(T') \geq p + 2 > g(T)$.

- $p = j$, tree T has $i \geq 2$ surplus nodes on level p, and T has $2^{p-1} - l$ surplus nodes on level $p-1$, where $l \leq \lfloor \frac{i}{2} \rfloor$.

  Applying the space-cost preserving transformation of Lemma 3.5 l times with $k-2 = p$, we obtain a locally SCP brother tree, T', with the same height and weight as T. Furthermore, $maxBin(T') = p+1$, and T' has $i-2l$ remaining surplus nodes on level p, no surplus nodes on level $p+1$, and l new surplus nodes on level $p+2$. Thus, $g(T') \geq p+2 > g(T)$.

In all cases, there exists a locally SCP brother tree, T', with the same height and weight as T, such that $g(T') > g(T)$. □

Finally, we are in a position to prove that if there exists a brother tree of height h and weight w, then there exists a locally SCP F^+ tree of height h and weight w.

Theorem 3.9 *For each height h and weight w, there exists a locally SCP brother tree, T, of height h and weight w, such that level i contains at most one surplus node, for $maxBin(T) \leq i < h-1$. That is, T is a locally SCP F^+ tree.*

Proof: Clearly, any locally SCP brother tree, T, of height h, weight w, and $maxBin(T) = h-1$ or $maxBin(T) = h$ satisfies the statement of the theorem. Similarly, any locally SCP brother tree, T, with $maxBin(T) = h-2$ cannot have any surplus nodes on level $h-2$. This follows because every binary node on level $h-1$ is a surplus node and, by Corollary 3.4, if level $h-2$ contains a surplus node, then level $h-1$ does not.

Let T_1 be a locally SCP brother tree of height h and weight w, and let $maxBin(T_1) = p_1$, for some $1 \leq p_1 \leq h-3$.

If $g(T_1) = h$, then there is no i, where $p_1 \leq i < h-1$, such that level i contains more than one surplus node. Thus, T_1 satisfies the theorem.

If $g(T_1) = j$, for some $p_1 \leq j < h-1$, then, by Lemma 3.8, there exists a locally SCP brother tree, T_2, of height h and weight w such that $g(T_2) >$

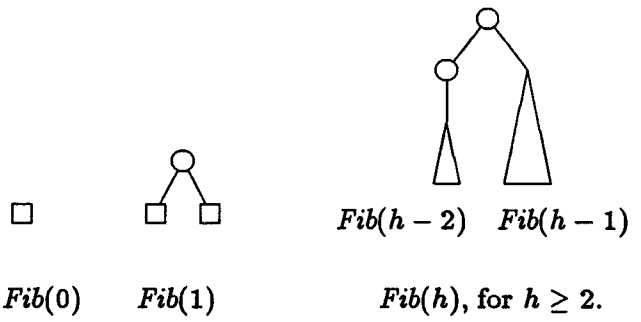


Figure 5: The definition of $Fib(h)$.

$g(T_1)$. If $g(T_2) = h$, then T_2 satisfies the theorem. Otherwise, we reapply Lemma 3.8 to T_2. This yields a sequence T_1, T_2, T_3, and so on, of locally SCP brother trees of weight w and height h, such that $g(T_i) < g(T_{i+1})$. This sequence is finite, so we must eventually find T_n such that $g(T_n) = h$; that is, T_n satisfies the conditions of the theorem. □

Thus, there is a family of locally SCP brother trees, at least one for each height h and weight w, such that each tree, T, in the family has at most one surplus node on each of the levels $maxBin(T)$ to $ht(T) - 2$.

3.4 The Height of the Largest Binary Prefix

We will now show that, given a height and weight, there is only one possible value for $maxBin(T)$, when T is a locally SCP F^+ tree. To do this, we need the notion of *Fibonacci trees*, as defined in Figure 5. The *Fibonacci numbers* are defined by $f_0 = 0$, $f_1 = 1$, and $f_{i+2} = f_{i+1} + f_i$, for $i \geq 0$.

Lemma 3.10 *The detailed profile of a Fibonacci tree of height h is*

$$\langle f_0, f_1 \rangle, \langle f_1, f_2 \rangle, \ldots, \langle f_{h-1}, f_h \rangle, \langle 0, f_{h+2} \rangle.$$

Thus, a Fibonacci tree of height h has weight f_{h+2} and has $f_{h+1} - 1$ unary nodes.

Proof: By induction on h, the height of the Fibonacci tree. See Lemma 3.1 in [KW87]. □

The importance of $Fib(h)$ in this section is that $Fib(h)$ contains no surplus nodes on levels 0 to $h-2$, as the following corollary shows. If the subtree rooted at a node in some brother tree, T, on level $maxBin(T) - 1$ does not contain any surplus nodes on any apart from level $ht(T) - 1$, then it must

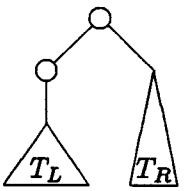


Figure 6: The root of T is not a surplus node.

be a Fibonacci brother tree. We use Fibonacci trees to help us construct F^+ trees with surplus nodes only in specified positions.

Corollary 3.11 *A brother tree, T, of height h contains no surplus nodes on level i, $0 \leq i \leq h-2$, if and only if T is a Fibonacci tree of height h.*

Proof: **If:** Let T be a Fibonacci tree. Then, by Lemma 3.10, $\beta_i = \omega_{i+1}$, for $0 \leq i < h-1$. Thus, T contains no surplus nodes on levels 0 to $h-2$.

Only if: Let T be a brother tree of height h that contains no surplus nodes on levels 0 to $h-2$. Clearly, $Fib(0)$ and $Fib(1)$, the only trees of heights 0 and 1, respectively, fall in this category and are Fibonacci trees. If $h > 1$, then, since the root of T is not a surplus node, we must have $\omega_0 = 0 = f_0$, $\beta_0 = 1 = f_1$, $\omega_1 = 1 = f_1$, and $\beta_1 = 1 = f_2$, see Figure 6. Thus, T_L and T_R are trees of heights $h-2$ and $h-1$ that have no surplus nodes on levels 0 to $h-4$ and $h-3$, respectively. By the induction hypothesis, T_L is a Fibonacci tree of height $h-2$ and T_R is a Fibonacci tree of height $h-1$. Thus, T is a Fibonacci tree of height h. □

Let $\mathcal{T}_{h,p}$, where $h > 0$ and $1 \leq p \leq h$, be the set of all brother trees, T, of height h with $maxBin(T) = p$ such that:

1. For $p \leq i < h$, if level i contains a surplus node, then level $i+1$ doesn't; and

2. For $p \leq i < h-1$, level i contains at most one surplus node.

In other words, the set $\mathcal{T}_{h,p}$ contains all F^+ trees of height h with a maximum complete binary prefix of height p. Observe that each $T \in \mathcal{T}_{h,p}$ can have as few as zero and as many as $2^{p-1}-1$ surplus nodes on level $p-1$.

We will show that the weight of a tree in $\mathcal{T}_{h,p}$ is strictly less than the weight of each tree in $\mathcal{T}_{h,p+1}$. Thus, given a height h and weight w, there is only one possible value of $maxBin(T)$ for a locally SCP brother tree, T, of height h and weight w, when T is also an F^+ tree. To prove this inequality between weights of trees in $\mathcal{T}_{h,p}$ and weights of trees $\mathcal{T}_{h,p+1}$, we show that a

maximum weight tree in $\mathcal{T}_{h,p}$ has weight exactly one less than the weight of a minimum weight tree in $\mathcal{T}_{h,p+1}$.

First, we characterize the minimum weight trees in $\mathcal{T}_{h,p}$.

Lemma 3.12 *Given a brother tree, T, of minimum weight in $\mathcal{T}_{h,p}$, every node on level $p-1$ in T is the root of a* Fib$(h-p+1)$ *subtree.*

Proof: We examine the following two cases:

- $p < h$.

  Let T be a brother tree of height h and $maxBin(T) = p$ that has minimum weight among all brother trees, T', of height h and $maxBin(T') = p$. If some node on level $p-1$ of T is not the root of a $Fib(h-p+1)$ subtree, then we can replace it with a $Fib(h-p+1)$ subtree. Since $Fib(h-p+1)$ is the brother tree that has minimum weight among all brother trees of height $h-p+1$, this does not increase the weight of T. If the resulting tree has the same weight, then the replaced subtree must be profile equivalent to $Fib(h-p+1)$. Otherwise, the resulting tree has smaller weight, and this contradicts the minimality of the weight of T. Thus, each node on level $p-1$ of T is the root of a $Fib(h-p+1)$ subtree. Since Fibonacci brother trees contain no surplus nodes, T has no surplus nodes on levels $p-1$ to $h-2$. Thus, $T \in \mathcal{T}_{h,p}$ and T is a brother tree of minimum weight in $\mathcal{T}_{h,p}$.

- $p = h$.

  There is only one tree T of height h with $maxBin(T) = h$. This is the only tree in $\mathcal{T}_{h,h}$. The tree is completely binary. Thus, each node on level $p-1 = h-1$ of this tree is a binary node and, therefore, each node on level $p-1$ is the root of a $Fib(1) = Fib(h-p+1)$ subtree.

In each case, each node on level $p-1$ is the root of a $Fib(h-p+1)$ subtree. □

Figure 7 displays a tree of minimum weight in $\mathcal{T}_{h,p}$.

Corollary 3.13 *A minimum weight tree, T, in $\mathcal{T}_{h,p}$ has weight $wt(T) = 2^{p-1} \cdot f_{h-p+3}$.*

Proof: By Lemma 3.12, each node on level $p-1$ of a minimum weight tree in $\mathcal{T}_{h,p}$ is the root of a $Fib(h-p+1)$ subtree. Since there are 2^{p-1} nodes on level $p-1$, the weight of such a tree is $2^{p-1} \cdot wt(Fib(h-p+1)) = 2^{p-1} \cdot f_{h-p+3}$. □

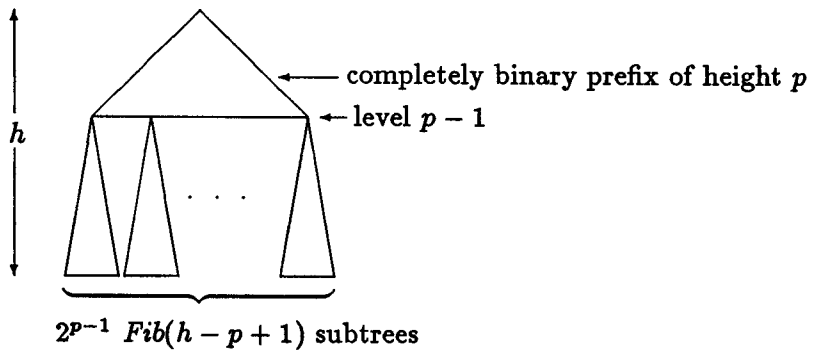


Figure 7: A tree of minimum weight in $\mathcal{T}_{h,p}$.

Maximum weight trees in $\mathcal{T}_{h,p}$ are more difficult to characterize because they contain surplus nodes beneath the maximum complete binary prefix.

First, we consider the positions of surplus nodes on the levels that can contain at most one surplus node per level; that is, levels p to $h-2$. For the moment, we restrict the discussion to subsets of trees in $\mathcal{T}_{h,p}$ that have the same number of surplus nodes on level $p-1$.

Definition 3.5 *Let $\mathcal{T}_{h,p,s}$ be the subset of trees in $\mathcal{T}_{h,p}$ that have s surplus nodes on level $p-1$.*

Lemma 3.14 *Let $p < h-2$. A brother tree, T, in $\mathcal{T}_{h,p,s}$ of maximum weight has exactly one surplus node on each of the levels $p, p+2, \ldots, p+2q$, where $h+4 \leq p+2q < h-2$, and no surplus nodes on the levels $p+1, p+3, \ldots, p+2q+1$.*

Proof: Let T be a maximum weight brother tree in $\mathcal{T}_{h,p,s}$. Assume T has one surplus node on each of the levels $p, p+2, \ldots, p+2i$, where $i < q$, but T has no surplus node on level $p+2(i+1)$. Since T is in $\mathcal{T}_{h,p,s}$ and T has a surplus node on level $p+2i$, T has no surplus node on level $p+2i+1$.

Case 1: T has no surplus node on level $p+2(i+1)+1$.

Then, add a surplus node to level $p+2(i+1)$ by choosing any binary node on level $p+2(i+1)$ and performing the transformation shown in Figure 8. The resulting tree, T', is in $\mathcal{T}_{h,p,s}$, and $wt(T') = wt(T) + wt(Fib(h-(p+2(i+1))-3)) = wt(T) + f_{h-(p+2(i+1))-1}$. This contradicts the maximality of the weight of T in $\mathcal{T}_{h,p,s}$, since $h-(p+2(i+1))-1 > 0$.

Case 2: T has a surplus node on level $p+2(i+1)+1$ and $p+2(i+1) = h-3$.

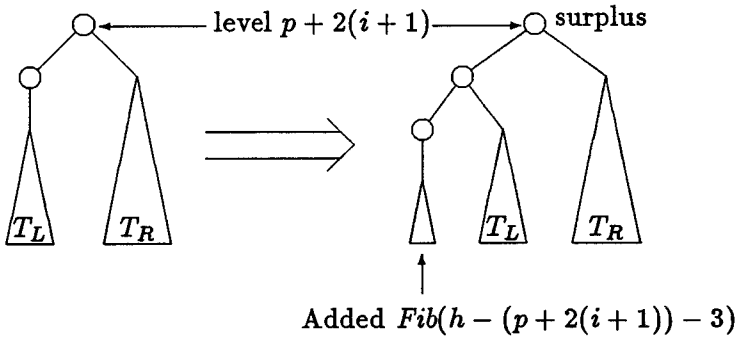


Figure 8: Create a new surplus node on level $p + 2(i + 1)$.

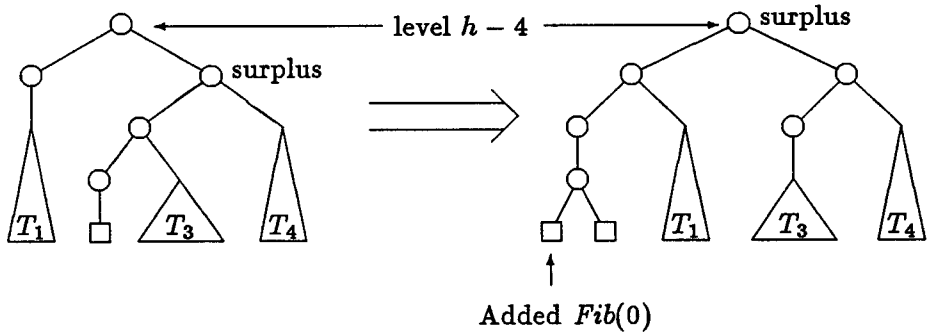


Figure 9: Shift a surplus node from level $h - 3$ to level $h - 4$.

This cannot occur because $p + 2(i + 1) + 1 = h - 2$ and level $h - 2$ cannot contain surplus nodes because every binary node on level $h - 1$ is a surplus node. By the definition of $\mathcal{T}_{h,p,s}$, if level $h - 2$ contains a surplus node, then level $h - 1$ cannot.

Case 3: T has a surplus node on level $p+2(i+1)+1$ and $p+2(i+1) = h-4$.

We cannot simply add a surplus node to level $h - 4$. Instead, we choose any binary node on level $h - 4$ and perform the transformation show in Figure 9. The resulting tree, T', has no surplus node on level $h - 3$, a new surplus node on level $h - 4$, and no other new surplus nodes on levels p to $h - 2$. T' is also in $\mathcal{T}_{h,p,s}$. In addition, the weight of T' is $wt(T') = wt(T) + 1$, which contradicts the maximality of the weight of T.

Case 4: T has a surplus node on level $p+2(i+1)+1$ and $p+2(i+1) < h-4$.

In this case, we cannot simply add a surplus node to level $p+2(i + 1)$. Instead, we perform the transformation shown in Figure 10. The re-

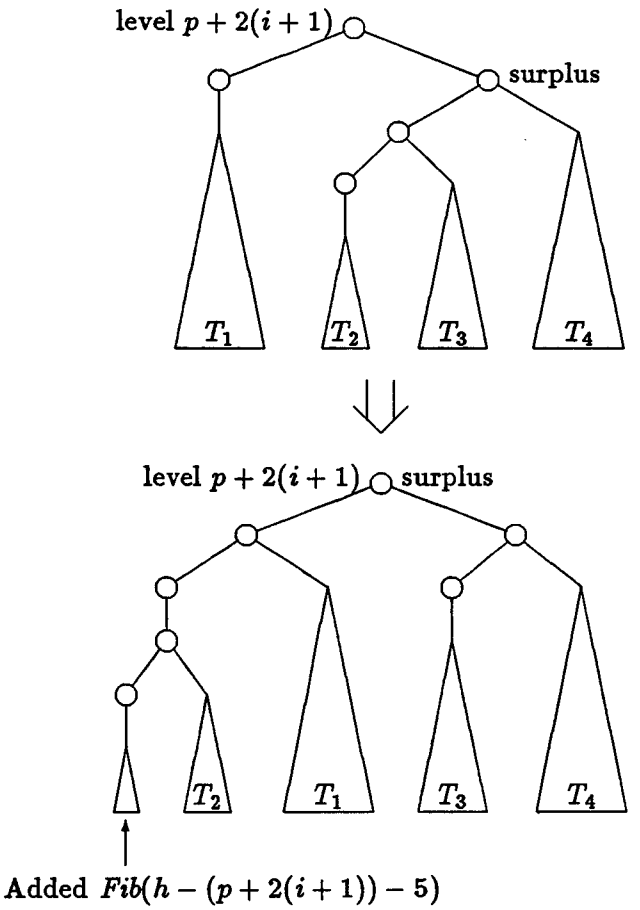


Figure 10: Shift a surplus node up to level $p + 2(i + 1)$.

sulting tree, T', has a surplus node on level $p + 2(i + 1)$, no surplus node on level $p + 2(i + 1) + 1$, and is in $\mathcal{T}_{h,p,s}$. Also,

$$\begin{aligned} wt(T') &= wt(T) + wt(Fib(h - (p + 2(i + 1)) - 5)) \\ &= wt(T) + f_{h-(p+2(i+1))-3}. \end{aligned}$$

Since $h - (p + 2(i + 1)) - 3 > 0$, this contradicts the maximality of the weight of T.

In each case, we see that T must have a surplus node on level $p + 2(i + 1)$, if T has maximum weight in $\mathcal{T}_{h,p,s}$. By induction, T must have a surplus node on each of the levels $p, p + 2, \ldots, p + 2q$, where $h - 4 \leq p + 2q < h - 2$.

The same argument applies if T does not have a surplus node on level p: if there is no surplus node on level $p + 1$, then add a surplus node to level p; otherwise, perform the transformation in Figure 10 (or Figure 9, if $p = h - 4$). In either case, we obtain a tree with weight greater than T; a contradiction. Thus, T must have a surplus node on level p.

Since T is in $\mathcal{T}_{h,p,s}$, whenever a level contains a surplus node, the level below it does not. Since T contains one surplus node on each of the levels $p, p + 2, \ldots, p + 2q$, where $h - 4 \leq p + 2q < h - 2$, it follows that T does not contain surplus nodes on levels $p + 1, p + 3, \ldots, p + 2q + 1$. Thus, T has exactly one surplus node on each of the levels $p, p + 2, \ldots, p + 2q$, and no surplus nodes on levels $p + 1, p + 3, \ldots, p + 2q + 1$. □

Now we can characterize maximum weight trees in $\mathcal{T}_{h,p}$, for $p < h - 2$.

Lemma 3.15 *Let $p < h - 2$. A brother tree, T, in $\mathcal{T}_{h,p}$ of maximum weight must have $2^{p-1} - 1$ surplus nodes on level $p - 1$ and one surplus node on each of the levels $p, p + 2, \ldots, p + 2q$, where $h - 4 \leq p + 2q < h - 2$, and no surplus nodes on level $p + 1, p + 3, \ldots, p + 2q + 1$.*

Proof: Let T be a brother tree of maximum weight in $\mathcal{T}_{h,p}$. The tree T cannot have 2^{p-1} surplus nodes on level $p - 1$, since level p would be completely binary and this would contradict the assumption that $maxBin(T) = p$. Thus, T can have at most $2^{p-1} - 1$ surplus nodes on level $p - 1$. We prove that T has exactly this number of surplus nodes on level $p - 1$ by contradiction.

Assume T does not have $2^{p-1} - 1$ surplus nodes on level $p - 1$. Then, there must be at least two binary nodes on level $p - 1$ that are not surplus nodes. Choose one of these and perform the transformation of Lemma 3.14 shown in Figure 8. Note that the chosen node on level $p - 1$ becomes a surplus node and no other surplus nodes are created, so the resulting tree, T', is in $\mathcal{T}_{h,p}$. Note also that $wt(T') = wt(T) + wt(Fib(h - p - 2)) = wt(T) + f_{h-p}$. Thus, T does not have maximum weight in $\mathcal{T}_{h,p}$; a contradiction.

Thus, $T \in \mathcal{T}_{h,p,s}$, where $s = 2^{p-1} - 1$. Since $\mathcal{T}_{h,p,s} \subseteq \mathcal{T}_{h,p}$ and T has maximum weight in $\mathcal{T}_{h,p}$, this implies T must have maximum weight in $\mathcal{T}_{h,p,s}$. By Lemma 3.14, T has one surplus node on each of the levels $p, p+2, \ldots, p+2q$, where $h-4 \leq p+2q < h-2$. □

The following lemma allows us to calculate the weight of the maximum weight trees in $\mathcal{T}_{h,p}$.

Lemma 3.16 *Let T be a brother tree of height h with $maxBin(T) = p < h$. If T has s surplus nodes on level $p-1$, exactly one surplus node on each of the levels $h-k_1-1, \ldots, h-k_r-1$, for some $r \geq 0$, where $p \leq h-k_1-1 < \ldots < h-k_r-1 < h-2$, and no other level among the levels $p, \ldots, h-2$ contains a surplus node, then the weight of T is*

$$2^{p-1} \cdot f_{h-p+3} + s \cdot f_{h-p} + \sum_{i=1}^{r} f_{k_i}.$$

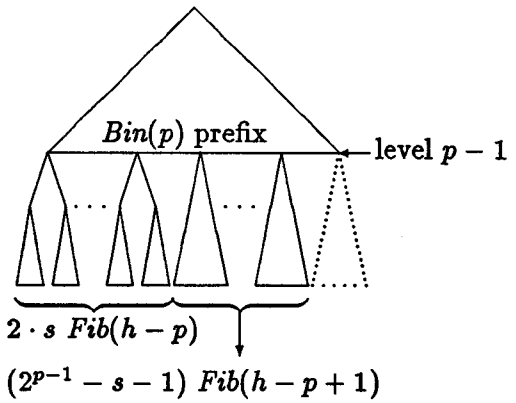
Proof: Consider the brother tree, T', in Figure 11. It has height h, $maxBin(T') = p$, s surplus nodes on level $p-1$, exactly one surplus node on each of the levels $h-k_1-1, \ldots, h-k_r-1$, and no other level among the levels $p, \ldots, h-2$ contains a surplus node. By Lemma 3.1, the detailed profiles of T and T' can be completely specified by the given information, and $\Delta(T) = \Delta(T')$. Thus, $wt(T) = wt(T')$.

We can calculate the weight of T' by adding the weights of the 2^{p-1} subtrees rooted at level $p-1$. Each the s surplus nodes on level $p-1$ contributes $2 \cdot wt(Fib(h-p))$ to the weight of T'. Each of the other nodes, except for one, contributes $wt(Fib(h-p+1))$ to the weight. The weight of the final subtree rooted at level $p-1$ can be calculated by summing the weights of the subtrees hanging from the binary "spine" of the final subtree. These subtrees are Fibonacci trees, starting at height 0 at the bottom and increasing to height $h-p-1$ at the top, with "glitches" caused by the surplus nodes on the spine. The surplus node on level $h-k_i-1$ causes a $Fib(k_i)$ subtree to occur where a $Fib(k_i-1)$ subtree is expected. But $Fib(k_i)$ is constructed from $Fib(k_i-1)$ and $Fib(k_i-2)$, so the surplus node on level $h-k_i-1$ adds an extra $Fib(k_i-2)$ into the calculation. Thus, the weight of the entire tree T' is

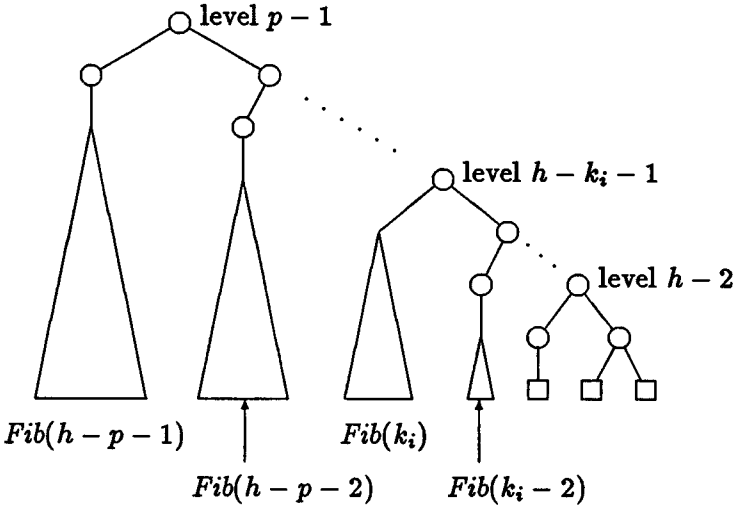
$$\begin{aligned} wt(T') &= s \cdot 2 \cdot wt(Fib(h-p)) + (2^{p-1} - s - 1) \cdot wt(Fib(h-p+1)) \\ &\quad + (2 + \sum_{i=0}^{h-p-1} wt(Fib(i)) + \sum_{i=1}^{r} wt(Fib(k_i-2))). \end{aligned}$$

The weight of $Fib(m)$ is f_{m+2}, so the weight of T' is

$$wt(T') = s \cdot 2 \cdot f_{h-p+2} + (2^{p-1} - s - 1) \cdot f_{h-p+3} + (2 + \sum_{i=2}^{h-p+1} f_i + \sum_{i=1}^{r} f_{k_i}).$$



The form of brother tree T', except for the rightmost subtree rooted at level $p-1$.



The rightmost subtree rooted at level $p-1$. It contains the surplus nodes on level $h-k_1-1,\ldots,h-k_r-1$.

Figure 11: A brother tree T' with $ht(T')=h$, $maxBin(T')=p$, s surplus nodes on level $p-1$, and one surplus node on each of the levels $h-k_1-1,\ldots,h-k_r-1$.

Since $f_0 = 0$, $f_1 = 1$, and $2 \cdot f_{h-p+2} = f_{h-p+2} + f_{h-p+1} + f_{h-p} = f_{h-p+3} + f_{h-p}$, we have

$$wt(T') = s \cdot (f_{h-p+3} + f_{h-p}) + (2^{p-1} - s - 1) \cdot f_{h-p+3} + (1 + \sum_{i=0}^{h-p+1} f_i + \sum_{i=1}^{r} f_{k_i}).$$

Finally, since $\sum_{i=0}^{h-p+1} f_i = f_{h-p+3} - 1$, we have

$$wt(T') = 2^{p-1} \cdot f_{h-p+3} + s \cdot f_{h-p} + \sum_{i=1}^{r} f_{k_i},$$

as required. □

Corollary 3.17 *For $p < h$, the weight of a maximum weight tree in $\mathcal{T}_{h,p}$ is $2^p \cdot f_{h-p+2} - 1$. For $p = h$, the set $\mathcal{T}_{h,p}$ contains exactly one tree of weight 2^h.*

Proof: We examine the following four possibilities separately: $p < h - 2$, $p = h - 2$, $p = h - 1$, and $p = h$.

Case $p < h - 2$.

Lemma 3.16 allows us to calculate the weight of the maximum weight tree, T, in $\mathcal{T}_{h,p}$ described in Lemma 3.15, namely,

$$\begin{aligned} wt(T) &= 2^{p-1} \cdot f_{h-p+3} + (2^{p-1} - 1) \cdot f_{h-p} + \sum_{i=0}^{q} f_{h-(p+2i)-1} \\ &= 2^{p-1} \cdot f_{h-p+3} + (2^{p-1} - 1) \cdot f_{h-p} \\ &\quad + \begin{cases} \sum_{j=1}^{q+1} f_{2j} & \text{if } p + 2q = h - 3 \\ \sum_{j=1}^{q+1} f_{2j+1} & \text{if } p + 2q = h - 4. \end{cases} \end{aligned}$$

Since $\sum_{i=1}^{n} f_{2i} = f_{2n+1} - 1$ and $\sum_{i=1}^{n} f_{2i+1} = f_{2(n+1)} - 1$, the weight becomes

$$\begin{aligned} wt(T) &= 2^{p-1} \cdot f_{h-p+3} + (2^{p-1} - 1) \cdot f_{h-p} + (f_{h-p} - 1) \\ &= 2^{p-1} \cdot (f_{h-p+3} + f_{h-p}) - 1. \end{aligned}$$

Finally, since $f_{h-p+3} + f_{h-p} = f_{h-p+2} + f_{h-p+1} + f_{h-p} = 2 \cdot f_{h-p+2}$, we have

$$wt(T) = 2^p \cdot f_{h-p+2} - 1.$$

Case $p = h - 2$.

By the argument used in the proof of Lemma 3.15, a maximum weight tree must have $2^{p-1} - 1$ surplus nodes on level $p - 1 = h - 3$. Such a tree cannot have any surplus nodes on level $p = h - 2$ because adjacent levels beneath the maximum binary prefix cannot both contain surplus nodes and

because every binary node on level $h-1$ is a surplus node. Thus, the maximum weight of a tree in $\mathcal{T}_{h,h-2}$ is $(2^{h-3}-1)\cdot 2\cdot wt(Fib(2))+wt(Fib(3)) = 2^{h-2}\cdot f_4 - 1 = 2^{p-1}\cdot f_{h-p+2} - 1$.

Case $p = h-1$.

As before, a tree of maximum weight in $\mathcal{T}_{h,h-1}$ has $2^{p-1}-1$ surplus nodes on level $p-1 = h-2$. By Lemma 3.1, the value of $maxBin(T)$ and the number of surplus nodes on level $h-2$ completely describe the detailed profile of a maximum weight tree. It has weight $(2^{h-2}-1)\cdot 2\cdot wt(Fib(1))+wt(Fib(2)) = 2^{h-1}\cdot f_3 - 1 = 2^p\cdot f_{h-p+2} - 1$.

Case $p = h$.

A tree in $\mathcal{T}_{h,h}$ has a maximum complete binary prefix of height h. Thus, a tree in $\mathcal{T}_{h,h}$ is completely binary. But there is only one completely binary tree of height h, so $\mathcal{T}_{h,h}$ contains exactly one tree. The weight of the completely binary tree of height h is 2^h. □

Having calculated the maximum and minimum weights of trees in $\mathcal{T}_{h,p}$, we can now calculate the height of the maximum complete binary prefix of a locally SCP F^+ tree, given its height and weight.

Theorem 3.18 *Let T be a locally SCP F^+ tree of height h and weight w. If $w < 2^h$, then $maxBin(T) = p$, where*

$$2^{p-1}\cdot f_{h-p+3} \le w \le 2^p\cdot f_{h-p+2} - 1.$$

If $w = 2^h$, then $maxBin(T) = h$.

Proof: We are given that T has height h. Let $maxBin(T) = p$. Since T is an F^+ tree, for $p \le i < h$, if level i contains a surplus node, then level $i+1$ doesn't, and, for $p \le i < h-1$, level i contains at most one surplus node. Thus, $T \in \mathcal{T}_{h,p}$.

By Corollary 3.17, the maximum weight among all trees in $\mathcal{T}_{h,p}$, for $p < h$, is $2^p\cdot f_{h-p+2} - 1$. This is one less than $2^p\cdot f_{h-p+2}$, the minimum weight among all trees in $\mathcal{T}_{h,p+1}$ (by Corollary 3.13). Also, the maximum weight among all trees in $\mathcal{T}_{h,h-1}$ is $2^{h-1}\cdot f_3 - 1 = 2^h - 1$, which is one less than the weight of the only tree in $\mathcal{T}_{h,h}$. Thus, $wt(T_{h,p}) < wt(T_{h,p+1})$, for any $T_{h,p} \in \mathcal{T}_{h,p}$ and any $T_{h,p+1} \in \mathcal{T}_{h,p+1}$, where $p < h$.

By the above argument, we see that $wt(T_{h,p}) > wt(T_{h,i})$, for any $T_{h,p} \in \mathcal{T}_{h,p}$ and any $T_{h,i} \in \mathcal{T}_{h,i}$, for any $1 \le i < p$. Also, $wt(T_{h,p}) < wt(T_{h,j})$, for any $T_{h,p} \in \mathcal{T}_{h,p}$ and any $T_{h,j} \in \mathcal{T}_{h,j}$, for any $p < j \le h$. Thus, the set $\mathcal{T}_{h,p}$ contains all F^+ trees, T, of height h with weight in the range $2^{p-1}\cdot f_{h-p+3} \le wt(T) \le 2^p\cdot f_{h-p+2} - 1$, for $p < h$.

Therefore, to find the height p of the maximum complete binary prefix of a locally SCP F^+ tree, T, of height h and weight $w < 2^h$, we need only

find the integer p such that $2^{p-1} \cdot f_{h-p+3} \leq w \leq 2^p \cdot f_{h-p+2} - 1$. If $w = 2^h$, then the tree is completely binary; that is, $maxBin(T) = h$. □

3.5 Surplus Nodes Beneath the Largest Binary Prefix

In the last section, we computed the height of the maximum complete binary prefix of a locally SCP F^+ tree, given a weight and a height. Now, we examine the numbers and positions of surplus nodes on the last level of the binary prefix (level $p-1$) and below it. First, we consider the level $p-1$.

Theorem 3.19 *Let T be a locally SCP F^+ tree of height h and weight w, and let $maxBin(T) = p < h$. Then, level $p-1$ contains exactly s surplus nodes, where $s \geq 0$, and*

$$\frac{(w - 2^{p-1} \cdot f_{h-p+3}) - f_{h-p}}{f_{h-p}} < s \leq \frac{(w - 2^{p-1} \cdot f_{h-p+3})}{f_{h-p}}.$$

Proof: Let s be the integer such that

$$\frac{(w - 2^{p-1} \cdot f_{h-p+3}) - f_{h-p}}{f_{h-p}} < s \leq \frac{(w - 2^{p-1} \cdot f_{h-p+3})}{f_{h-p}}.$$

By Theorem 3.18, since T is a locally SCP F^+ tree of height h and T has a maximum complete binary prefix of height p, where $p < h$, it follows that

$$2^{p-1} \cdot f_{h-p+3} \leq w \leq 2^p \cdot f_{h-p+2} - 1.$$

Thus, $w - 2^{p-1} \cdot f_{h-p+3} \geq 0$, and, therefore, $s \geq 0$.

If T contains $m > s$ surplus nodes on level $p-1$, then the weight of T is at least $2^{p-1} \cdot f_{h-p+3} + m \cdot f_{h-p}$ which is at least $2^{p-1} \cdot f_{h-p+3} + (s+1) \cdot f_{h-p}$. But,

$$\frac{(w - 2^{p-1} \cdot f_{h-p+3}) - f_{h-p}}{f_{h-p}} < s \leq \frac{(w - 2^{p-1} \cdot f_{h-p+3})}{f_{h-p}};$$

that is,

$$2^{p-1} \cdot f_{h-p+3} + s \cdot f_{h-p} \leq w < 2^{p-1} \cdot f_{h-p+3} + (s+1) \cdot f_{h-p}.$$

So, T contains at most s surplus nodes on level $p-1$.

Suppose T contains $m' < s$ surplus nodes on level $p-1$; then T is in $\mathcal{T}_{h,p,m'}$.

If $p < h-2$, then, by Lemma 3.14, a maximum weight tree in $\mathcal{T}_{h,p,m'}$ has exactly one surplus node on each of the levels $p, p+2, \ldots, p+2q$, where $h-4 \leq p+2q < h-2$. By Lemma 3.16, a tree, T', with $maxBin(T') = p$, height h, m' surplus nodes on level $maxBin(T')$, and one surplus node on

each of the levels $p, p+2, \ldots, p+2q$, where $h-4 \leq p+2q < h-2$, has weight equal to

$$\begin{aligned} wt(T') &= 2^{p-1} \cdot f_{h-p+3} + m' \cdot f_{h-p} + \sum_{i=0}^{q} f_{h-(p+2i)-1} \\ &= 2^{p-1} \cdot f_{h-p+3} + m' \cdot f_{h-p} + \begin{cases} \sum_{j=1}^{q+1} f_{2j} & \text{if } p+2q = h-3 \\ \sum_{j=1}^{q+1} f_{2j+1} & \text{if } p+2q = h-4 \end{cases} \\ &= 2^{p-1} \cdot f_{h-p+3} + m' \cdot f_{h-p} + (f_{h-p} - 1) \\ &< 2^{p-1} \cdot f_{h-p+3} + s \cdot f_{h-p}. \end{aligned}$$

If $p = h-2$, then the detailed profile of T is completely specified by $maxBin(T) = h-2$ and T has m' surplus nodes on level $h-3$. (T cannot have surplus nodes on level $h-2$.) The weight of T is

$$\begin{aligned} wt(T) &= (2^{h-3} - m') \cdot wt(Fib(3)) + m' \cdot 2 \cdot wt(Fib(2)) \\ &= 2^{p-1} \cdot f_{h-p+3} + m' \cdot f_{h-p} \\ &< 2^{p-1} \cdot f_{h-p+3} + s \cdot f_{h-p}. \end{aligned}$$

If $p = h-1$ and there are m' surplus nodes on level $p-1 = h-2$, then the detailed profile is completely specified and the corresponding weight is

$$\begin{aligned} wt(T) &= (2^{h-2} - m') \cdot wt(Fib(2)) + m' \cdot 2 \cdot wt(Fib(1)) \\ &= 2^{p-1} \cdot f_{h-p+3} + m' \cdot f_{h-p} \\ &< 2^{p-1} \cdot f_{h-p+3} + s \cdot f_{h-p}. \end{aligned}$$

But we assumed that $wt(T) \geq 2^{p-1} \cdot f_{h-p+3} + s \cdot f_{h-p}$. If T has fewer than s surplus nodes on level $p-1$, we obtain a contradiction in each case. Therefore, T must have at least s surplus nodes on level $p-1$. □

To specify which levels among the levels $maxBin(T)$ to $ht(T)-2$ contain a surplus node, we make use of the Fibonacci numbering system.

Theorem 3.20 ([Lek52]) *Every positive integer n has a unique representation $n = f_{k_1} + f_{k_2} + \ldots + f_{k_r}$, where $k_i \geq k_{i+1} + 2$, for $1 \leq i < r$, and $k_r \geq 2$.*

The condition $k_i \geq k_{i+1}+2$ corresponds to the requirement that adjacent levels beneath the binary prefix of a locally SCP brother tree cannot both contain surplus nodes. Each term f_{k_i} in the sum corresponds to a level $h - k_i - 1$ that contains a surplus node.

Theorem 3.21 *Let T be a locally SCP F^+ tree of height h and weight w, with $maxBin(T) = p < h - 2$ and having s surplus nodes on level $p - 1$. Let $\sum_{i=1}^{r} f_{k_i}$, for some $r \geq 0$, be the unique Fibonacci representation of $w - 2^{p-1} \cdot f_{h-p+3} - s \cdot f_{h-p}$. Then, T has exactly one surplus node on each of the levels $h - k_1 - 1, h - k_2 - 1, \ldots, h - k_r - 1$, and no other level among the levels $p, p + 1, \ldots, h - 2$ contains a surplus node.*

Proof: Since $\sum_{i=1}^{r} f_{k_i}$, for some $r \geq 0$, is the unique Fibonacci representation of $w - 2^{p-1} \cdot f_{h-p+3} - s \cdot f_{h-p}$, it follows that $k_i \geq k_{i+1} + 2$, for $1 \leq i < r$, and $k_r \geq 2$.

Suppose the brother tree T has exactly one surplus node on each of the levels $h - b_1 - 1, h - b_2 - 1, \ldots, h - b_q - 1$, for some $q \geq 0$, where $p \leq h - b_1 - 1$, and $h - b_q - 1 < h - 2$, and T contains no further surplus nodes on levels $p, p + 1, \ldots, h - 3$. Since adjacent levels cannot both contain surplus nodes, it follows that $h - b_i - 1 \leq (h - b_{i+1} - 1) - 2$. Since $maxBin(T) = p$, and T has s surplus nodes on level $p - 1$, by Lemma 3.16, T has weight

$$w = 2^{p-1} \cdot f_{h-p+3} + s \cdot f_{h-p} + \sum_{i=1}^{q} f_{b_i}.$$

Therefore,

$$\sum_{i=1}^{q} f_{b_i} = w - 2^{p-1} \cdot f_{h-p+3} - s \cdot f_{h-p}.$$

Since $h - b_i - 1 \leq (h - b_{i+1} - 1) - 2$ and $h - b_q - 1 < h - 2$, we have $b_i \geq b_{i+1} + 2$ and $b_q \geq 2$. Thus, $\sum_{i=1}^{q} f_{b_i}$ is the unique Fibonacci representation of $w - 2^{p-1} \cdot f_{h-p+3} - s \cdot f_{h-p}$. But $\sum_{i=1}^{r} f_{k_i}$ is the unique Fibonacci representation of $w - 2^{p-1} \cdot f_{h-p+3} - s \cdot f_{h-p}$, so, $r = q$ and $k_i = b_i$, for $1 \leq i \leq r$. □

3.6 Tying It All Together

Now we have all the pieces necessary to completely describe a locally SCP F^+ tree, given a height and weight.

Theorem 3.22 *Let $f_{h+2} \leq w \leq 2^h$. Then, there exists a locally SCP F^+ tree, T, of height h and weight w that is completely described by the following characteristics.*

1. *Its binary prefix has height p, where p is the largest integer such that $2^{p-1} \cdot f_{h-p+3} \leq w$; that is, $maxBin(T) = p$.*

2. *If $maxBin(T) < h$, then surplus nodes are distributed on the levels $p - 1, p, \ldots, h - 2$ as follows:*

(a) T has s surplus nodes on level $p-1$, where

$$\frac{w - 2^{p-1} \cdot f_{h-p+3} - f_{h-p}}{f_{h-p}} < s \leq \frac{w - 2^{p-1} \cdot f_{h-p+3}}{f_{h-p}}.$$

(b) T has one surplus node on each of the levels $h - k_i - 1$, where $\sum_{i=1}^{r} f_{k_i}$ is the unique Fibonacci representation of

$$w - 2^{p-1} \cdot f_{h-p+3} - s \cdot f_{h-p}.$$

Furthermore, all locally SCP F^+ trees of height h and weight w have this description.

Proof: Since $f_{h+2} \leq w \leq 2^h$, we know that there exists a brother tree of height h and weight w. By Theorem 3.9, there exists a locally SCP F^+ tree, T, of height h and weight w.

By Theorem 3.18, if $w = 2^h = 2^{h-1} \cdot f_{h-h+3}$, then $maxBin(T) = h$, which completely describes the tree (it is completely binary). Thus, there is exactly one locally SCP F^+ tree, if $w = 2^h$. Otherwise, $2^{p-1} \cdot f_{h-p+3} \leq w \leq 2^p \cdot f_{h-p+2} - 1$, where $maxBin(T) = p$. Thus, $maxBin(T)$ must be the largest integer p such that $2^{p-1} \cdot f_{h-p+3} \leq w$.

By Theorem 3.19, if $maxBin(T) < h$, then level $p-1$ must contain exactly s surplus nodes where

$$\frac{(w - 2^{p-1} \cdot f_{h-p+3}) - f_{h-p}}{f_{h-p}} < s \leq \frac{(w - 2^{p-1} \cdot f_{h-p+3})}{f_{h-p}},$$

and, by Theorem 3.21, there must be exactly one surplus node on each of the levels $h - k_i - 1$, where $\sum_{i=1}^{r} f_{k_i}$ is the unique Fibonacci representation of $w - 2^{p-1} \cdot f_{h-p+3} - s \cdot f_{h-p}$.

We have determined the only choices for $maxBin(T)$, and the number of surplus nodes on each of the levels $maxBin(T) - 1, \ldots, h-2$. Thus, any locally SCP F^+ tree of height h and weight w have the same detailed profile, which can be found using Lemma 3.1. □

Now that we have characterized a locally SCP brother tree of height h and weight w, we can calculate the number of unary nodes of a locally SCP brother tree. The space cost is simply the sum of the number of internal binary nodes ($w-1$) and the number of unary nodes. Clearly, if a brother tree is completely binary, that is, $w = 2^h$, then the tree contains no unary nodes. Otherwise, the calculation of the number of unary nodes is similar to the calculation, in Lemma 3.16, of the weight of a brother tree given its description.

Corollary 3.23 *Let T be a brother tree of height h with $maxBin(T) = p < h$. If T has s surplus nodes on level $p-1$, exactly one surplus node on each of the levels $h-k_1-1,\ldots,h-k_r-1$, for some $r \geq 0$, where $p \leq h-k_1-1 < \ldots < h-k_r-1 < h-2$, and no other level among the levels $p,\ldots,h-2$ contains a surplus node, then the number of unary nodes in T is*

$$2^{p-1}\cdot(f_{h-p+2}-1)+s\cdot(f_{h-p-1}-1)-f_{h-p}+\sum_{i=1}^{r}(f_{k_i-1}-1).$$

Proof: Let $Unary(T)$ be the number of unary nodes in brother tree T. As in the proof of Lemma 3.16, the brother tree T' in Figure 11 has the same detailed profile as T and, therefore, contains the same number of unary nodes.

The binary prefix of T' contains no unary nodes. Therefore, in the same way that we calculated the weight of the brother tree T', we can calculate $Unary(T')$ by summing up the number of unary nodes in the subtrees rooted at level $p-1$. Thus, we have

$$\begin{aligned}
Unary(T') &= 2\cdot s\cdot Unary(Fib(h-p)) \\
&\quad +(2^{p-1}-s-1)\cdot Unary(Fib(h-p+1)) \\
&\quad +\sum_{j=0}^{h-p-1}(1+Unary(Fib(j)))+\sum_{i=1}^{r}Unary(Fib(k_i-2)) \\
&= 2^{p-1}\cdot(Unary(Fib(h-p+1)) \\
&\quad +s\cdot(2\cdot Unary(Fib(h-p))-Unary(Fib(h-p+1)) \\
&\quad -(Unary(Fib(h-p+1))-1) \\
&\quad +\sum_{j=0}^{h-p-1}(1+Unary(Fib(j)))+\sum_{i=1}^{r}Unary(Fib(k_i-2))
\end{aligned}$$

Since $Unary(Fib(c)) = f_{c+1}-1$, we have

$$\begin{aligned}
Unary(T') &= 2^{p-1}\cdot(f_{h-p+2}-1)+s\cdot(2\cdot(f_{h-p+1}-1)-(f_{h-p+2}-1)) \\
&\quad -(f_{h-p+2}-1)+\sum_{j=0}^{h-p-1}(1+(f_j-1))+\sum_{i=1}^{r}(f_{k_i-1}-1).
\end{aligned}$$

But the sum $\sum_{j=0}^{h-p+1} f_j = f_{h-p+3}-1$, so we have

$$\begin{aligned}
Unary(T') &= 2^{p-1}\cdot(f_{h-p+2}-1)+s\cdot(2\cdot(f_{h-p+1}-1)-(f_{h-p+2}-1)) \\
&\quad -(f_{h-p+2}-1)+(f_{h-p+1}-1)+\sum_{i=1}^{r}(f_{k_i-1}-1).
\end{aligned}$$

Also, since $2 \cdot f_{h-p+1} - f_{h-p+2} = f_{h-p+1} + (f_{h-p} + f_{h-p-1}) - (f_{h-p+1} + f_{h-p}) = f_{h-p-1}$, we get

$$\begin{aligned} \mathit{Unary}(T') &= 2^{p-1} \cdot (f_{h-p+2} - 1) + s \cdot (f_{h-p-1} - 1) \\ &\quad -(f_{h-p+2} - 1) + (f_{h-p+1} - 1) + \sum_{i=1}^{r} (f_{k_i - 1} - 1). \end{aligned}$$

Finally, since $-f_{h-p+2} + f_{h-p+1} = -f_{h-p+1} - f_{h-p} + f_{h-p+1} = -f_{h-p}$, we get

$$\begin{aligned} \mathit{Unary}(T') &= 2^{p-1} \cdot (f_{h-p+2} - 1) + s \cdot (f_{h-p-1} - 1) \\ &\quad -f_{h-p} + \sum_{i=1}^{r} (f_{k_i - 1} - 1), \end{aligned}$$

which is the number of unary nodes in brother tree T, also. □

4 Characterizing Unbalanced AVL Trees

Recall that the number of unbalanced nodes in an AVL tree, T', is exactly equal to the number of unary nodes in the brother tree *expand*(T'). Thus, the maximum number of unbalanced nodes that an AVL tree of weight w and height h can contain is the number of unary nodes in a locally SCP brother tree of weight w and height h.

We have shown how to find the unique description of all locally SCP F^+ trees of a given weight w and height h. We would like a corresponding description for the corresponding AVL trees under the contract operation.

Each of the locally SCP F^+ trees has a maximum complete binary prefix of height p, where p is the largest integer such that $2^{p-1} \cdot f_{h-p+3} \leq w$. Since the binary prefix contains no unary nodes, it is unaffected by the contract operation. Thus, the corresponding AVL trees are also completely binary on levels $0, \ldots, p-1$.

Describing the effect of the contract operation on the levels beneath the binary prefix is more complex. Certain levels of the F^+ brother trees contain single surplus nodes. Which level such a surplus node ends up on in the corresponding AVL tree depends on the number of unary nodes in the path from the root to the surplus node in the brother tree.

However, the effect of the contract operation on the locally SCP F^+ brother tree, T, pictured in Figure 11, is easy to describe. The description uses the definition of *Fibonacci AVL trees*, which are defined in Figure 12.

Theorem 4.1 *Let w and h be integers such that $f_{h+2} \leq w \leq 2^h$. Then, there exists an AVL tree, T', of weight w and height h that contains the maximum number of unbalanced nodes for an AVL tree of weight w and height h, and AVL tree T' is completely described by the following:*

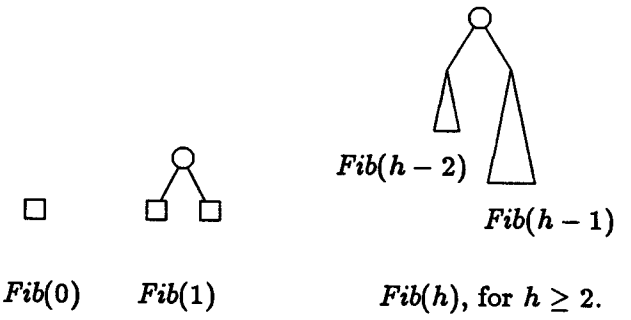


Figure 12: The recursive definition of Fibonacci AVL trees.

1. *Levels* $0, \ldots, p-1$ *are completely binary, where* p *is the largest integer such that* $2^{p-1} \cdot f_{h-p+3} \leq w$.

2. *If* $p < h$, *then the* 2^{p-1} *subtrees rooted at level* $p-1$ *can be divided into the following groups:*

   (a) *Each of* s *nodes on level* $p-1$ *is the binary parent of two Fibonacci AVL trees of height* $h-p$, *where*

   $$\frac{w - 2^{p-1} \cdot f_{h-p+3} - f_{h-p}}{f_{h-p}} < s \leq \frac{w - 2^{p-1} \cdot f_{h-p+3}}{f_{h-p}}.$$

   (b) *A further* $2^{p-1} - s - 1$ *nodes on level* $p-1$ *are the roots of Fibonacci AVL trees of height* $h-p+1$.

   (c) *Let* $\sum_{i=1}^{r} f_{k_i}$ *be the unique Fibonacci representation of*

   $$w - 2^{p-1} \cdot f_{h-p+3} - s \cdot f_{h-p}.$$

   The final subtree rooted at level $p-1$ *consists of a chain of binary nodes of length* $h-p+1$. *Each node on the chain (except the last one) has a child that is the next node in the chain and one other child. (The last node in the chain has two external children.) The other child of the binary node on level* j *in tree* T' *is the root of a Fibonacci AVL tree of height* $h-j-2$, *except when* $j = h - k_i - 1$. *When* $j = h - k_i - 1$, *the other child is the root of a Fibonacci AVL tree of height* k_i.

Proof: The brother tree T of Figure 11 has a binary prefix of height p, s surplus nodes on level $p-1$, and a single surplus node on each of levels $h-k_1-1, \ldots, h-k_r-1$. The surplus nodes on levels $h-k_1-1, \ldots, h-k_r-1$ appear only on the "spine" of the rightmost subtree rooted on level $p-1$. The spine is completely binary, so the surplus nodes are not moved by the

contract operation. Thus, the corresponding AVL tree, T', looks just like the brother tree except that the Fibonacci subtrees are Fibonacci AVL trees. □

5 Conclusion

Although we have characterized AVL trees with the maximum numbers of unbalanced nodes for their heights and weights, there remain some unanswered questions.

The most obvious problem left open is the characterization of pessimally unbalanced AVL trees for a given weight. Experimental evidence suggests that the pessimally unbalanced AVL trees of a given weight have the maximum height for AVL trees of that weight. So far, we have not been able to prove this conjecture.

Also, the relationship between unbalanced AVL trees and comparison cost pessimal AVL trees remains an open problem.

Finally, it may be possible to enumerate all locally SCP brother trees of a given height and weight using the inverse of the transformation given in Lemma 3.5. We first find the locally SCP F^+ tree with the given height and weight, and then repeatedly apply the inverse transformation in some sequence to enumerate all the other locally SCP brother tree of that height and weight.

References

[KW87] Rolf Klein and Derick Wood. The node visit cost of brother trees. *Information and Computation*, 75(2):107–129, 1987.

[KW89a] Rolf Klein and Derick Wood. On the path length of binary trees. *Journal of the ACM*, 1989. To appear.

[KW89b] Rolf Klein and Derick Wood. A tight upper bound for the path length of AVL trees. *Theoretical Computer Science*, 1989. To appear.

[Lek52] C. G. Lekkerkerker. Voorstelling van natuurlijke getallen door een som van getallen van fibonacci. *Simon Stevin*, 29:190–195, 1952.

[OPR+84] Thomas Ottmann, D. Stott Parker, Arnold L. Rosenberg, Hans-Werner Six, and Derick Wood. Minimal-cost brother trees. *SIAM Journal on Computing*, 13(1):197–217, 1984.

[OW80] Thomas Ottmann and Derick Wood. 1-2 brother trees or AVL trees revisited. *The Computer Journal*, 23(3):248–255, August 1980.