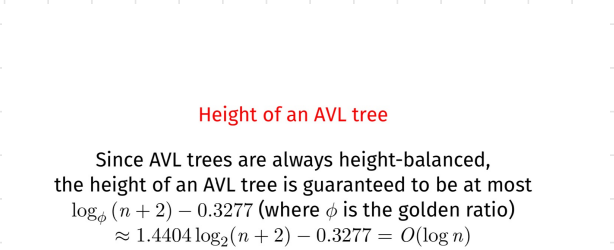
(max/min)

Height of Height-balanced Trees

Harry Zhang

Background/How I fell in this rabbit hole

One day when I was bored, I was scrolling through the AVL tree lecture slides and I came across this:



There was no explanation to this, and I was kind of like huh? What's the golden ratio doing here? And why 0.3277? On the Wikipedia page for AVL trees, they also have this formula with a very brief explanation but do cite a book: Sorting and Searching by Donald Knuth.

Now, this book does contain a proof which goes into some depth. However, take a look at the theorem they prove:

Theorem A (Adelson-Velsky and Landis). *The height of a balanced tree with N internal nodes always lies between* $\lg(N+1)$ *and* $1.4405 \lg(N+2) - 0.3277$.

It refers to the height of a balanced tree with N internal nodes as opposed to N total nodes.

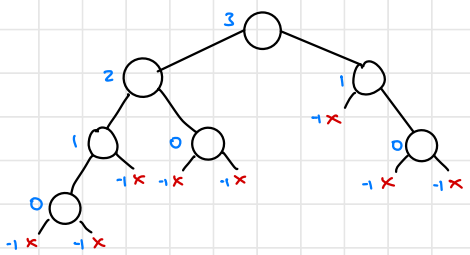
So this got me thinking: can I make sense of these numbers, and give a better interval for height-balanced trees with n total nodes?

Some definitions just in case

Just in case someone random is reading this, I will clear up some definitions relating to height so we are all on the same page.

1. The **height** of a binary tree is:
   - -1 if the tree is empty
   - the length of the longest path from the root node to a leaf node.

For example, in this tree, the height of each subtree is given in blue:



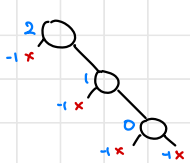
An alternative way to calculate height is:

height(NULL) = -1

height(t) = max[height(t→left), height(t→right)] + 1

2. A binary tree is **height-balanced** if for every node, the absolute difference in height of the left and right subtree is at most 1.

For example, the tree above is height-balanced. This tree is not:



The Theorem

For any height-balanced tree with $n \geq 0$ nodes, the height of the tree h lies in the interval

$$\log_2(n+1) - 1 \leq h \leq \log_\phi(n + a) + b$$

where ϕ is the golden ratio, with

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

$$a = 1 + \frac{(1-\phi)^2}{\sqrt{5}} \approx 1.1708$$

$$b = \log_\phi(\sqrt{5}) - 3 \approx -1.3277.$$

Equality holds on both sides when we have an empty tree, i.e. $n = 0$, and the height becomes -1, as expected.

Admittedly, this interval is a bit difficult to read, so after approximating the constants, we get

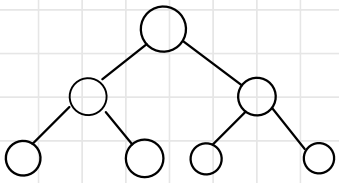
$$\log_2(n+1) - 1 \leq h < 1.4405 \log_2(n + 1.171) - 1.3277.$$

The proof for this is done in two parts - one for each inequality.

Proof Part One - Lower Bound

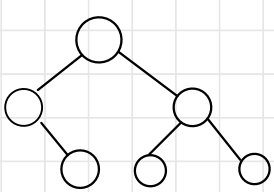
This bound is much easier to prove, so we will do it first. The idea is to go, given a height h, what is the maximum number of nodes a tree can have?

The answer is when we have a perfect binary tree, which is a binary tree where all interior nodes have exactly two children and all leaf nodes are on the same level. For example, the perfect binary tree with height 2 is:

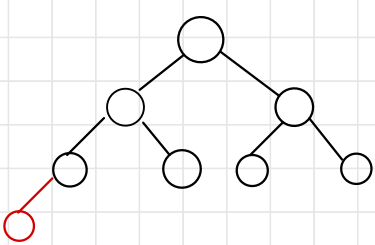


n = 7
h = 2

It is clear that this tree is height-balanced, and contains the maximum number of nodes for a tree with height 2. If we delete a leaf node, there are less nodes for the same height. On the other hand, if we add a node to any leaf, this will cause the height to increase by 1:



n = 6
h = 2



n = 8
h = 3

Now, our perfect binary tree with height 2 contains $1 + 2 + 4 = 7$ nodes. In general, a perfect binary tree with height h contains

$$1 + 2 + 4 + \cdots + 2^h = 1 + 2^1 + 2^2 + \cdots + 2^h$$

$$= 2^{h+1} - 1 \text{ nodes.}$$

Since this is the maximum number of nodes in a height-balanced binary tree, we can deduce that for any height-balanced binary tree with n nodes and height h,

$$n \leq 2^{h+1} - 1$$

$$n + 1 \leq 2^{h+1}$$

$$\log_2(n+1) \leq h + 1$$

$$\log_2(n+1) - 1 \leq h$$

as expected.

Proof Part 2 - Upper Bound

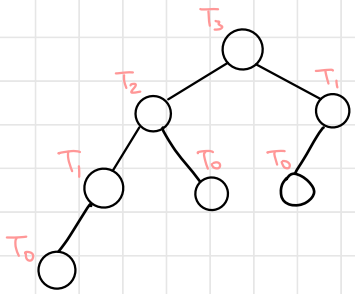
The proof of the upper bound will follow a similar structure:

1. Given a height h, determine what type of height-balanced tree has the minimum number of nodes
2. Find an expression for the number of nodes in this type of tree.
3. Use a mathematical identity to simplify this expression
4. Re-arrange an inequality to get the upper bound for h.

1. Fibonacci Trees

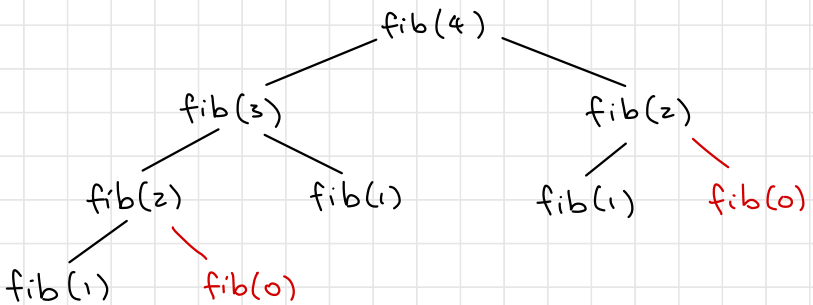
We begin with the following argument. Let T_h be a height-balanced binary tree with height h and the minimal number of nodes. Then, the subtrees of the root must have height $h-1$ and $h-2$. Since we want all subtrees to be height-balanced, and T_h to have minimal nodes, we may assume that the subtrees are T_{h-1} and T_{h-2}.

Then, we can recursively repeat this process for each of the subtrees until we hit T_0, which is a leaf node or T_{-1}, which is an empty tree. For example, here is T_3:



It turns out this tree is also known as a Fibonacci tree of order 4. How you can kind of think of a Fibonacci tree is that it's a tree that shows the recursive function calls needed to calculate the n^{th} Fibonacci number, except the fib(0) calls are empty trees.

To illustrate, this tree



has the same structure as T_3 if we exclude nodes for fib(0). Also note that the subtrees of the root are essentially T_2 and T_1.

This can generalise to any height h, and so we have deduced that the height-balanced tree with height h and the minimal number of nodes is the Fibonacci tree of order $h+1$.

2. Fibonacci Trees and the Fibonacci sequence

The Fibonacci sequence is a very well-known sequence given by

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

The first few values in the sequence are

0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, ...

We are going to find an interesting relationship between the number of nodes in a Fibonacci tree and the Fibonacci sequence.

Let t_k denote the number of nodes in the Fibonacci tree of order k. We will prove that

$$t_k = F_{k+2} - 1 \qquad (*)$$

for all $k \geq 0$. As a sanity check, we can see that $t_4 = 7$ by counting the number of nodes in the previous tree, and F_6 (the 7^{th} number in the above sequence) is 8.

We will proceed via induction. In the case when $k = 0$,

$$\text{LHS} = t_0 = 0$$

since the order 0 tree has no nodes. Also,

$$\text{RHS} = F_2 - 1 = 1 - 1 = 0.$$

Similarly, when $k = 1$,

$$\text{LHS} = t_1 = 1,$$
$$\text{RHS} = F_3 - 1 = 2 - 1 = 1.$$

Hence, $(*)$ is true for $k = 0$ and $k = 1$.

Now, assume for the inductive hypothesis that $(*)$ is true for $k = m$ and $k = m+1$, for some non-negative integer m. That is,

$$t_m = F_{m+2} - 1$$
$$t_{m+1} = F_{m+3} - 1.$$

We can now prove that $t_{m+2} = F_{m+4} - 1$ as follows:

$$\begin{aligned}
\text{LHS} = t_{m+2} &= 1 + t_{m+1} + t_m && \text{(subtrees are Fibonacci trees)} \\
&= 1 + (F_{m+3} - 1) + (F_{m+2} - 1) && \text{(by assumptions)} \\
&= (F_{m+2} + F_{m+3}) - 1 \\
&= F_{m+4} - 1 && \text{(Fibonacci recursive relation)} \\
&= \text{RHS}.
\end{aligned}$$

Hence, (*) is true by induction. So, now we have a simple expression for the number of nodes in a Fibonacci tree, and combining this with part 1, we can say that a height-balanced tree with height h must have at least $F_{h+3} - 1$ nodes.

3. Closed form expression for Fibonacci numbers

The expression for the number of nodes in terms of Fibonacci numbers is a bit unwieldy, and doesn't explain how the height is logarithmic to the number of nodes. Instead, we must use an alternate representation for the k^{th} Fibonacci number:

$$F_k = \frac{\phi^k - (1-\phi)^k}{\sqrt{5}} = \frac{1}{\sqrt{5}}\left[\left(\frac{1+\sqrt{5}}{2}\right)^k - \left(\frac{1-\sqrt{5}}{2}\right)^k\right] \quad (\#)$$

Again, we will prove this by induction. When $k=0$,

$$LHS = F_0 = 0$$

$$RHS = \frac{1}{\sqrt{5}}\left[\phi^0 - (1-\phi)^0\right] = \frac{1}{\sqrt{5}}[1-1] = 0$$

so (#) is true for $k=0$. When $k=1$,

$$LHS = F_1 = 1$$

$$RHS = \frac{1}{\sqrt{5}}\left[\phi^1 - (1-\phi)^1\right] = \frac{1}{\sqrt{5}}[2\phi - 1] = \frac{1}{\sqrt{5}}\left[2\left(\frac{1+\sqrt{5}}{2}\right) - 1\right] = \frac{1}{\sqrt{5}}[1+\sqrt{5}-1] = 1$$

so (#) is also true for $k=1$.

Now for the inductive hypothesis, we assume that (#) is true for $k=m$ and $k=m+1$ for some non-negative m.

That is,

$$F_m = \frac{1}{\sqrt{5}}\left[\phi^m - (1-\phi)^m\right],$$
$$F_{m+1} = \frac{1}{\sqrt{5}}\left[\phi^{m+1} - (1-\phi)^{m+1}\right].$$

Now, we want to show that

$$F_{m+2} = \frac{1}{\sqrt{5}}\left[\phi^{m+2} - (1-\phi)^{m+2}\right]$$

So

$$\text{LHS} = F_{m+2} = F_{m+1} + F_m \qquad \text{(recursive relation)}$$

$$= \frac{1}{\sqrt{5}}\left[\phi^{m+1} - (1-\phi)^{m+1}\right] + \frac{1}{\sqrt{5}}\left[\phi^m - (1-\phi)^m\right]$$

$$= \frac{1}{\sqrt{5}}\left[\phi^m(\phi+1) - (1-\phi)^m(2-\phi)\right]$$

$$= \frac{1}{\sqrt{5}}\left[\phi^m\left(\frac{1+\sqrt{5}}{2}+1\right) - (1-\phi)^m\left(2-\frac{1+\sqrt{5}}{2}\right)\right]$$

$$= \frac{1}{\sqrt{5}}\left[\phi^m\left(\frac{3+\sqrt{5}}{2}\right) - (1-\phi)^m\left(\frac{3-\sqrt{5}}{2}\right)\right]$$

$$= \frac{1}{\sqrt{5}}\left[\phi^m\left(\frac{1+2\sqrt{5}+5}{4}\right) - (1-\phi)^m\left(\frac{1-2\sqrt{5}+5}{4}\right)\right]$$

$$= \frac{1}{\sqrt{5}}\left[\phi^m\left(\frac{1+\sqrt{5}}{2}\right)^2 - (1-\phi)^m\left(\frac{1-\sqrt{5}}{2}\right)^2\right]$$

$$= \frac{1}{\sqrt{5}}\left[\phi^{m+2} - (1-\phi)^{m+2}\right]$$

$$= \text{RHS}.$$

Thus (#) holds for $k = m+2$, and by induction, all non-negative integers k.

Therefore, we can express the minimum number of nodes in a height balanced tree of height h as

$$\frac{\phi^{h+3} - (1-\phi)^{h+3}}{\sqrt{5}} - 1.$$

4. Re-arranging for h.

Finally we have our inequality

$$\frac{\phi^{h+3} - (1-\phi)^{h+3}}{\sqrt{5}} - 1 \leq n,$$

let's do some rearranging

$$\frac{\phi^{h+3} - (1-\phi)^{h+3}}{\sqrt{5}} \leq n+1$$

$$\phi^{h+3} - (1-\phi)^{h+3} \leq \sqrt{5}(n+1).$$

Unfortunately, there is no way to get h on its own simply by rearranging. So, we will try find an upper bound for the $(1-\phi)^{h+3}$ term. Obviously,

$$(1-\phi)^{h+3} \leq |1-\phi|^{h+3}$$

and note that $0 < |1-\phi| \approx 0.618 < 1$. Hence, if we increase h, the value of $|1-\phi|^{h+3}$ will decrease towards 0. So, the upper bound of $(1-\phi)^{h+3}$ occurs at the minimum possible h, which is -1. That is,

$$(1-\phi)^{h+3} \leq |1-\phi|^{-1+3}$$

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

Now, we can finally reach our desired expression, as

$$\phi^{h+3} - (1-\phi)^2 \leq \sqrt{5}(n+1)$$

$$\phi^{h+3} \leq \sqrt{5}(n+1) + (1-\phi)^2$$

$$h+3 \leq \log_\phi \left[\sqrt{5}(n+1) + (1-\phi)^2\right]$$

$$h \leq \log_\phi \left[\sqrt{5}\left(n+1+\frac{(1-\phi)^2}{\sqrt{5}}\right)\right] - 3$$

$$h \leq \log_\phi\left(n+1+\frac{(1-\phi)^2}{\sqrt{5}}\right) + \log_\phi(\sqrt{5}) - 3$$

as required.

And there we have it! By putting together the results from part 1 and part 2, we get

$$\log_2(n+1) - 1 \leq h \leq \log_\phi\left(n+1+\frac{(1-\phi)^2}{\sqrt{5}}\right) + \log_\phi(\sqrt{5}) - 3$$

for any height-balanced tree! This completes the proof.

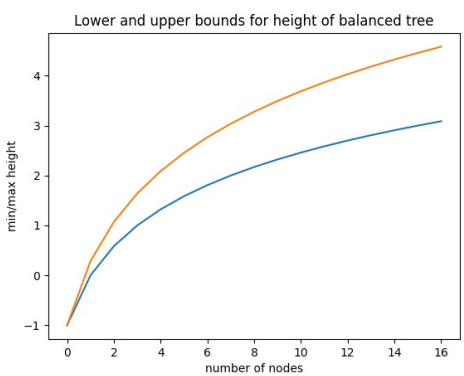
□

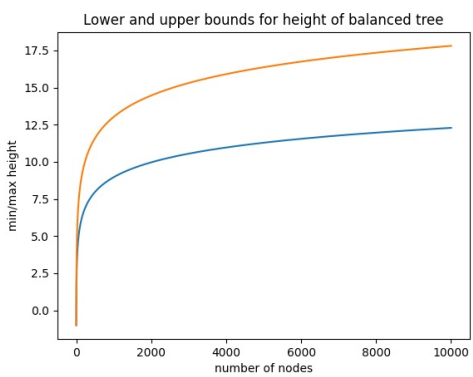
Bonus - Plotting the Lower / Upper Bound Curves

I made a Python script to plot the lower and upper bounds for the height. Here is the code:

```python
from math import log, sqrt
import matplotlib.pyplot as plt
import numpy as np

phi = (1 + sqrt(5)) / 2
a = 1 + (1 - phi) ** 2 / sqrt(5)
b = log(5, phi) / 2 - 3
num_nodes = 17

idx = np.arange(num_nodes)
lower = [log(i + 1, 2) - 1 for i in range(num_nodes)]
higher = [log(i + a, phi) + b for i in range(num_nodes)]

plt.plot(idx, lower)
plt.plot(idx, higher)
plt.title("Lower and upper bounds for height of balanced tree")
plt.xlabel("number of nodes")
plt.ylabel("min/max height")

plt.show()
```

and here are some of the plots I made with it:





Looks like what you'd expect :).