# Application: AVL trees and the golden ratio

AVL trees are used for storing information in an efficient manner.

- We will see exactly how in the data structures course.
- This slide set takes a look at how high an AVL tree of a given size can be.

## The golden ratio

The golden ratio is an irrational number $\phi = \frac{1+\sqrt{5}}{2} \cong 1.618$ with many interesting properties. Among them

- $\phi - 1 = 1/\phi$
- $\phi = 1 + \cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\ddots_{1}}}}$
- $\phi$ turns up in many geometric figures including pentagrams and dodecahedra
- It is the ratio, in the limit, of successive members of the Fibonacci sequence

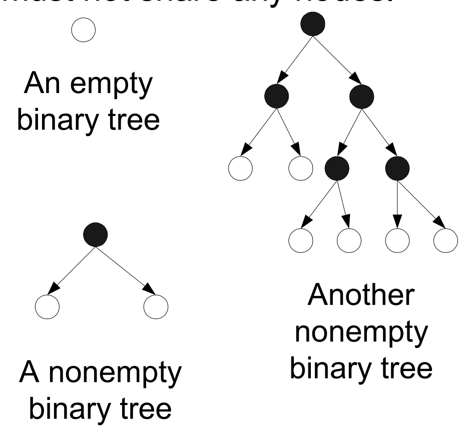




# Binary trees

A binary tree is either

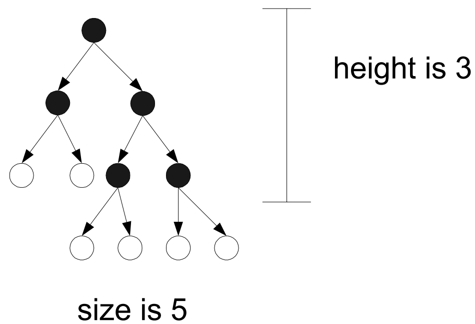
- The empty binary tree, for which I'll write ○
- Or a point (called a **node**) connected to two smaller binary trees (called its **children**)
- The children must not share any nodes.



# The height and size of a binary tree

The **size** of a binary tree is the number of nodes it has.

The **height** of a binary tree is number of levels of nodes it has




Note that ◯ has height 0 and size 0.

Clearly a binary tree of size $n$ can have a height of up to $n$.

When binary trees are used to store data:

- The amount of information stored is proportional to size of tree
- The time to access data is proportional to the height

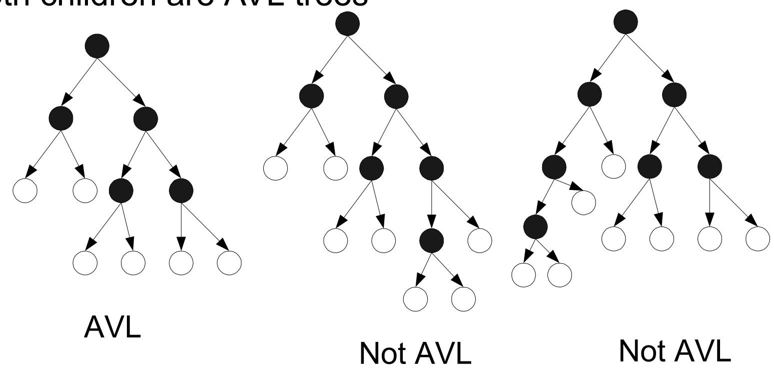




# AVL trees

AVL trees are binary trees with the following restrictions.

- The empty tree is an AVL tree
- A nonempty binary tree is AVL if
  - $*$ the height difference of the children is at most 1, and
  - $*$ both children are AVL trees



# The question

We wish to access large amounts of data quickly.

- Remember amount of information is proportional to size of tree
- and access time is proportional to the height of the tree.

So the question is how high can an AVL tree of a given size be?

We start by asking a closely related question:

- How small can an AVL tree of a given height be?

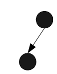
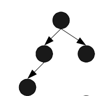
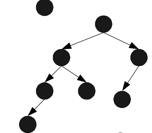
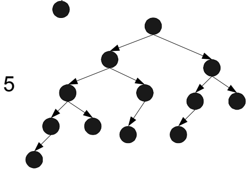




# How small can an AVL tree of a given height be?

Let's make a table with the smallest AVL tree of each height

(empty trees are implied)

| Height | Smallest tree | Size |
|---|---|---|
| 0 |  | 0 |
| 1 |  | 1 |
| 2 |  | 2 |
| 3 |  | 4 |
| 4 |  | 7 |
| 5 |  | 12 |

# The minsize function

In the table, each tree (of height $h > 1$) has, as children, smallest trees of heights $h-2$ and $h-1$

So we have

$$\begin{aligned}\text{minsize}(0) &= 0\\ \text{minsize}(1) &= 1\\ \text{minsize}(h) &= \text{minsize}(h-1) + \text{minsize}(h-2) + 1, \text{ for } h \geq 2\end{aligned}$$

Note the recurrence is not homogeneous.

Try a few values

$$0, 1, 2, 4, 7, 12, 20, 33, 54$$

Compare with the Fibonacci sequence

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55$$

We find

$$\text{minsize}(h) = \text{fib}(h+1) - 1$$

where

$$\begin{aligned}\text{fib}(0) &= 1\\ \text{fib}(1) &= 1\\ \text{fib}(n) &= \text{fib}(n-1) + \text{fib}(n-2), \text{ for } n \geq 2\end{aligned}$$

We can prove this by (complete induction).





Since fib is defined by a linear homogeneous recurrence relation of degree 2 we can solve it

$$\text{fib}(n) = \frac{1}{\sqrt{5}} \times \phi^{n+1} - \frac{1}{\sqrt{5}} \times (\frac{-1}{\phi})^{n+1} \quad \text{for all } n \in \mathbb{N}$$

where

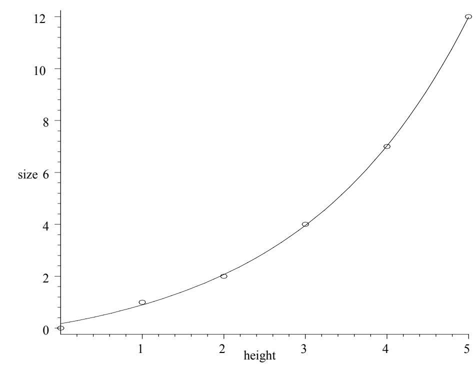
$$\phi = \frac{1+\sqrt{5}}{2}$$

Consider $\frac{1}{\sqrt{5}} \times \phi^{n+1} - \frac{1}{\sqrt{5}} \times (\frac{-1}{\phi})^{n+1}$ for $n \in \mathbb{R}$ and $n \geq 0$.

The first term is real, the second is complex.

As $n$ gets big, the complex term becomes small.

So we get $\text{minsize}(h) \cong \frac{1}{\sqrt{5}} \times \phi^{h+2} - 1$




minsize$(h)$ dots $\quad \frac{1}{\sqrt{5}} \times \phi^{h+2} - 1$ line

# The maximum height per given size

| Height | 0 | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| Min size | 0 | 1 | 2 | 4 | 7 | 12 |

Let $h'$ be the height of a tree of size $s'$. We know that for all $h$,

$$h' \geq h \rightarrow s' \geq \text{minsize}(h)$$

Contrapositively: For all $h$,

$$s' < \text{minsize}(h) \rightarrow h' < h$$

| Size | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Max height | 0 | 1 | 2 | 2 | 3 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 5 | 5 | 5 |

Note that for $s$ such that $\text{minsize}(h-1) < s \leq \text{minsize}(h)$

$$\text{maxheight}(s) = h$$

$\text{maxheight}(s)$ is approximately an inverse of $\text{minsize}(h)$



Discrete Math. for Engineering, 2004. Application slides 5 Theodore Norvell, Memorial University

So invert $\frac{1}{\sqrt{5}} \times \phi^{h+2} - 1$

$$\begin{aligned} & s = \frac{1}{\sqrt{5}} \times \phi^{h+2} - 1 \\ \Leftrightarrow\ & \sqrt{5}\,(s+1) = \phi^{h+2} \\ \Leftrightarrow\ & \log_\phi \sqrt{5}\,(s+1) = h+2 \\ \Leftrightarrow\ & \log_\phi \sqrt{5}\,(s+1) - 2 = h \\ \Leftrightarrow\ & \log_\phi 2 \times \log_2(s+1) + \log_\phi \sqrt{5} - 2 = h \end{aligned}$$

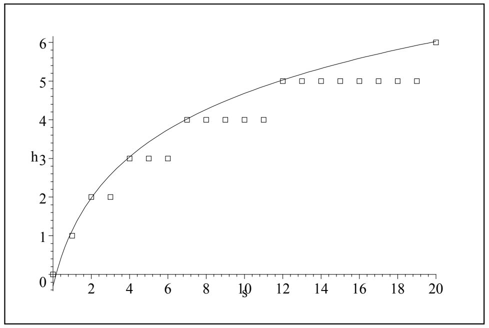
so $\text{maxheight}(s) \cong 1.44 \times \log_2(s+1) - 0.3$

For example

$$\begin{aligned} \text{maxheight}(10^6) &\cong 29 \\ \text{maxheight}(10^9) &\cong 43 \\ \text{maxheight}(10^{12}) &\cong 58 \end{aligned}$$

This means large amounts of data can be accessed in a small amount of time, if we store the data in AVL trees.

# Graphing maxheight




$\mathrm{maxheight}(s)$ dots $1.44 \times \log_2(s+1) - 0.3$ line