Solving Recurrences II
COMS10007 2020, Lecture 14

Dr. John Lapinskas
(substituting for Dr. Christian Konrad)

March 18th 2020

The recursion tree method

This lecture we'll ignore O-notation and divisibility issues, and focus on examples of the recurrences themselves. Consider the recurrence

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Step 1: Use the recursion tree method to get a good guess at a solution.

As with the mergesort analysis, we view this as a tree.

Each recursive invocation corresponds to a child node — so the root has three children, each of which has three children, and so on.

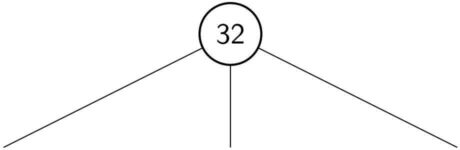
Each node gets labelled with the non-recursive running time at that step, and then $T(n)$ is the sum of all the labels in the tree.

A concrete example

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

$n = 64$

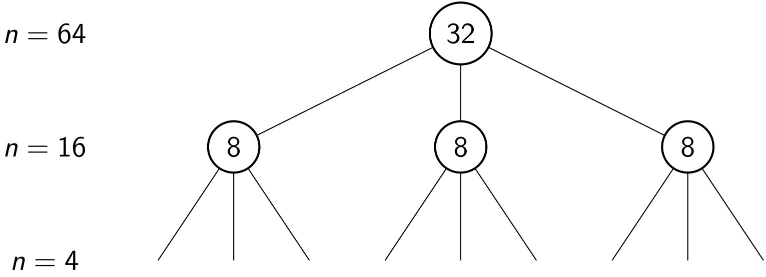
$n = 16$



$$T(64) = 3T(16) + 32$$

A concrete example

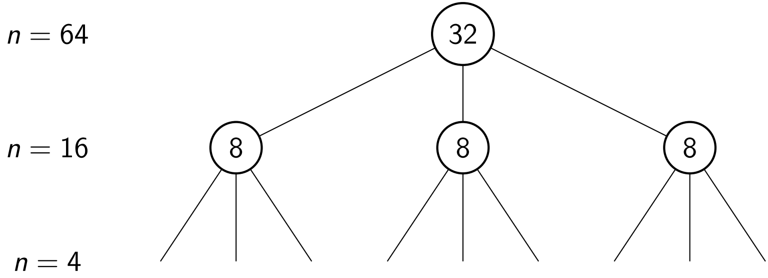
$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$



$$T(64) = 3T(16) + 32 = \mathbf{9T(4)} + \mathbf{3 \cdot 8} + 32$$

A concrete example

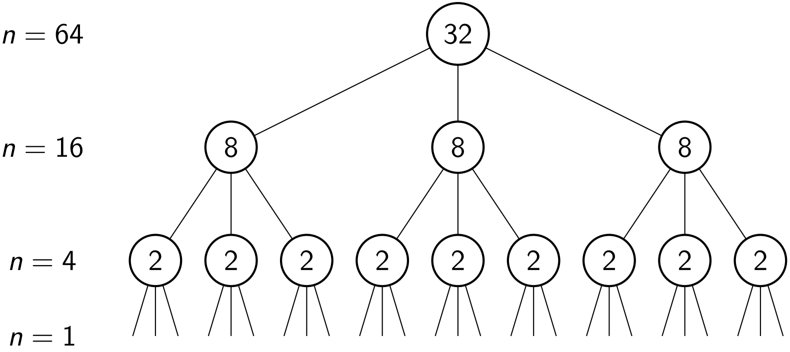
$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$



$$T(64) = 9T(4) + 3 \cdot 8 + 32$$

A concrete example

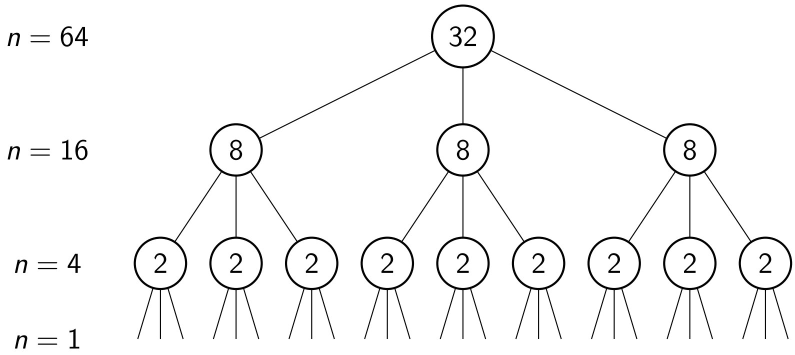
$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of 4.}$$



$$T(64) = 9T(4) + 3 \cdot 8 + 32 = \mathbf{27\,T(1)} + \mathbf{9 \cdot 2} + 3 \cdot 8 + 32$$

A concrete example

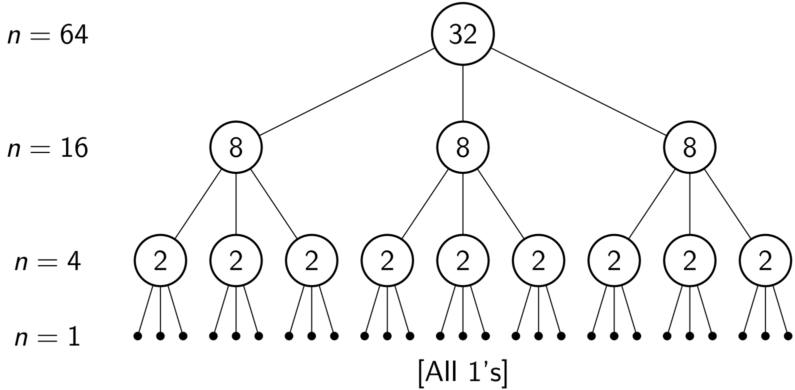
$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$



$$T(64) = 27\,T(1) + 9 \cdot 2 + 3 \cdot 8 + 32$$

A concrete example

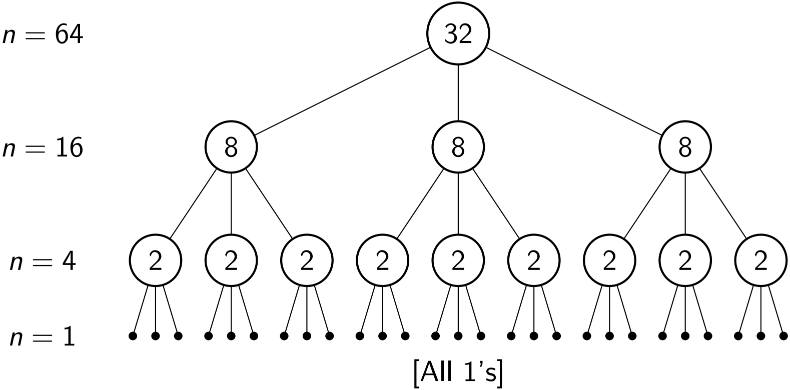
$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$



$$T(64) = 27T(1) + 9 \cdot 2 + 3 \cdot 8 + 32 = \mathbf{27 \cdot 1} + 9 \cdot 2 + 3 \cdot 8 + 32$$

A concrete example

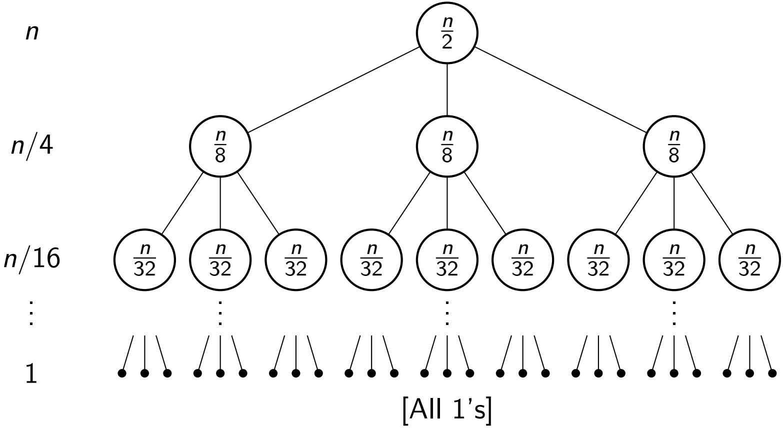
$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of 4.}$$



$$T(64) = 27 \cdot 1 + 9 \cdot 2 + 3 \cdot 8 + 32 = 101.$$

The general case

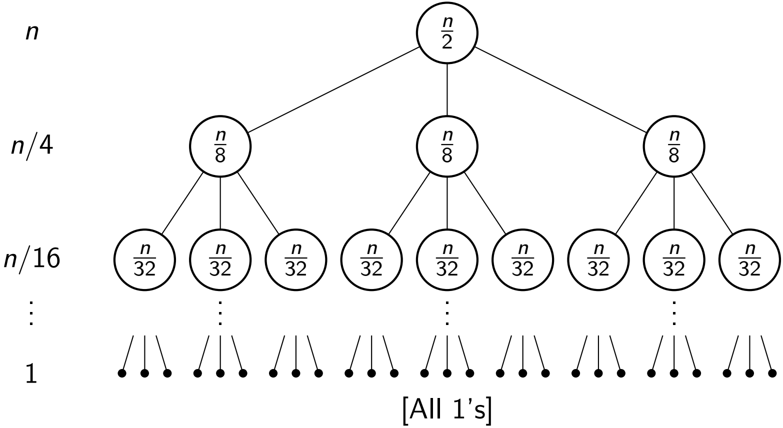
$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$



Except for the bottom, level i has 3^{i-1} nodes, each with cost $n/(2 \cdot 4^{i-1})$.

The general case

$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n$ is a power of 4.



Except for the bottom, level i has 3^{i-1} nodes, each with cost $n/(2 \cdot 4^{i-1})$.
Let $\#(\text{levels}) = t$. Then $n/4^{t-1} = 1$, so we have $t = 1 + \log_4 n$.

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \leq \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n$ is a power of 4.

There are $1 + \log_4 n$ levels in total.
For $i \leq \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2}$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \leq \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1}$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \le \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \le \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i}$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \le \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \le \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i} = \frac{n}{2} \cdot O(1)$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \leq \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \leq \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i} = \frac{n}{2} \cdot O(1) = O(n).$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of 4.}$$

There are $1 + \log_4 n$ levels in total.
For $i \le \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \le \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i} = \frac{n}{2} \cdot O(1) = O(n).$$

And we have

$$3^{\log_4 n}$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \leq \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \leq \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i} = \frac{n}{2} \cdot O(1) = O(n).$$

And we have

$$3^{\log_4 n} = 2^{\log(3) \cdot \log_4(n)}$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \leq \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \leq \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i} = \frac{n}{2} \cdot O(1) = O(n).$$

And we have

$$3^{\log_4 n} = 2^{\log(3) \cdot \log_4(n)} = 2^{\log(3) \cdot \frac{\log(n)}{\log(4)}}$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \leq \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \leq \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i} = \frac{n}{2} \cdot O(1) = O(n).$$

And we have

$$3^{\log_4 n} = 2^{\log(3) \cdot \log_4(n)} = 2^{\log(3) \cdot \frac{\log(n)}{\log(4)}} = n^{\frac{\log(3)}{\log(4)}}$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \le \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \le \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i} = \frac{n}{2} \cdot O(1) = O(n).$$

And we have

$$3^{\log_4 n} = 2^{\log(3) \cdot \log_4(n)} = 2^{\log(3) \cdot \frac{\log(n)}{\log(4)}} = n^{\frac{\log(3)}{\log(4)}} = o(n).$$

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

There are $1 + \log_4 n$ levels in total.
For $i \le \log_4 n$, level i has 3^{i-1} nodes with time cost $n/(2 \cdot 4^{i-1})$ each.
Level $1 + \log_4 n$ has $3^{\log_4 n}$ nodes with time cost 1 each.

$T(n)$ is the total time cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} + 3^{\log_4 n}.$$

Summing the geometric series gives

$$\sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \frac{n}{2} = \frac{n}{2} \cdot \sum_{i=1}^{\log_4 n} \left(\frac{3}{4}\right)^{i-1} \le \frac{n}{2} \cdot \sum_{i=0}^{\infty} \left(\frac{3}{4}\right)^{i} = \frac{n}{2} \cdot O(1) = O(n).$$

And we have

$$3^{\log_4 n} = 2^{\log(3)\cdot\log_4(n)} = 2^{\log(3)\cdot\frac{\log(n)}{\log(4)}} = n^{\frac{\log(3)}{\log(4)}} = o(n).$$

So overall, we expect $T(n) = O(n)$. In other words, **the root dominates**.

Formal proof via substitution

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \leq Cn$ for all $n \geq 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Formal proof via substitution

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \leq Cn$ for all $n \geq 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Base case $\boldsymbol{n = 1}$: We have $T(1) = 1 \leq C \cdot 1$ whenever $\mathbf{C \geq 1}$. ✓

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \le Cn$ for all $n \ge 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Base case $n = 1$: We have $T(1) = 1 \le C \cdot 1$ whenever $\mathbf{C \ge 1}$. ✓

Inductive step: Suppose that for all $1 \le n' \le n-1$, $T(n') \le Cn'$.
Then we must prove $T(n) \le Cn$.

Formal proof via substitution

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \le Cn$ for all $n \ge 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Base case $n = 1$: We have $T(1) = 1 \le C \cdot 1$ whenever $C \geq 1$. ✓

Inductive step: Suppose that for all $1 \le n' \le n-1$, $T(n') \le Cn'$.
Then we must prove $T(n) \le Cn$.

By the induction hypothesis, we have

$$T(n) = 3T(n/4) + n/2$$

Formal proof via substitution

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \leq Cn$ for all $n \geq 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Base case $n = 1$: We have $T(1) = 1 \leq C \cdot 1$ whenever $C \geq 1$. ✓

Inductive step: Suppose that for all $1 \leq n' \leq n - 1$, $T(n') \leq Cn'$.
Then we must prove $T(n) \leq Cn$.

By the induction hypothesis, we have

$$T(n) = 3T(n/4) + n/2 \leq 3Cn/4 + n/2$$

Formal proof via substitution

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \leq Cn$ for all $n \geq 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Base case $\boldsymbol{n = 1}$: We have $T(1) = 1 \leq C \cdot 1$ whenever $\boldsymbol{C \geq 1}$. ✓

Inductive step: Suppose that for all $1 \leq n' \leq n-1$, $T(n') \leq Cn'$.
Then we must prove $T(n) \leq Cn$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) &= 3T(n/4) + n/2 \leq 3Cn/4 + n/2 \\ &= Cn - Cn/4 + n/2 \end{aligned}$$

Formal proof via substitution

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \leq Cn$ for all $n \geq 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Base case $n = 1$: We have $T(1) = 1 \leq C \cdot 1$ whenever $\mathbf{C \geq 1}$. ✓

Inductive step: Suppose that for all $1 \leq n' \leq n-1$, $T(n') \leq Cn'$.
Then we must prove $T(n) \leq Cn$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) &= 3T(n/4) + n/2 \leq 3Cn/4 + n/2 \\ &= Cn - Cn/4 + n/2 = Cn + \left(\tfrac{1}{2} - \tfrac{C}{4}\right)n. \end{aligned}$$

Formal proof via substitution

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \leq Cn$ for all $n \geq 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Base case $n = 1$: We have $T(1) = 1 \leq C \cdot 1$ whenever $\mathbf{C \geq 1}$. ✓

Inductive step: Suppose that for all $1 \leq n' \leq n-1$, $T(n') \leq Cn'$.
Then we must prove $T(n) \leq Cn$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) &= 3T(n/4) + n/2 \leq 3Cn/4 + n/2 \\ &= Cn - Cn/4 + n/2 = Cn + \left(\tfrac{1}{2} - \tfrac{C}{4}\right)n. \end{aligned}$$

This is at most Cn iff $\mathbf{C \geq 2}$. ✓

Formal proof via substitution

$$T(1) = 1, \qquad T(n) = 3T(n/4) + n/2, \qquad n \text{ is a power of } 4.$$

Guess: $T(n) \le Cn$ for all $n \ge 1$ (C to be determined).

Now that we have a good guess, proving it formally is a standard induction.

Base case $n = 1$: We have $T(1) = 1 \le C \cdot 1$ whenever $\mathbf{C \ge 1}$. ✓

Inductive step: Suppose that for all $1 \le n' \le n-1$, $T(n') \le Cn'$.
Then we must prove $T(n) \le Cn$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) &= 3T(n/4) + n/2 \le 3Cn/4 + n/2 \\ &= Cn - Cn/4 + n/2 = Cn + \left(\tfrac{1}{2} - \tfrac{C}{4}\right)n. \end{aligned}$$

This is at most Cn iff $\mathbf{C \ge 2}$. ✓

We have proved $T(n) \le 2n$ for all $n \ge 1$, and hence $T(n) = O(n)$. □

Another example

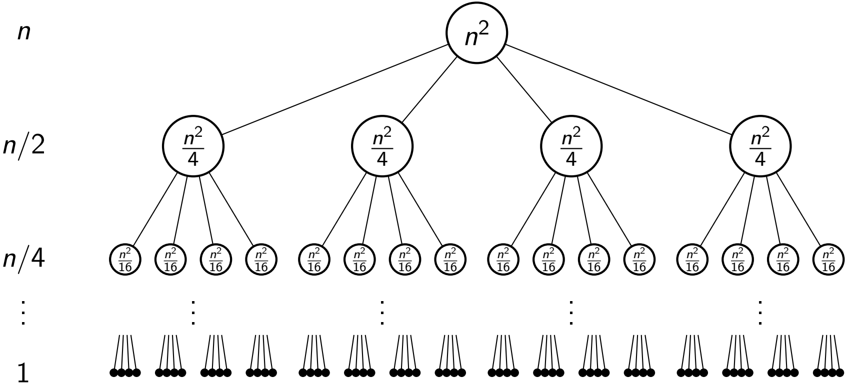
Now consider the recurrence

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of 2.}$$

Another example

Now consider the recurrence

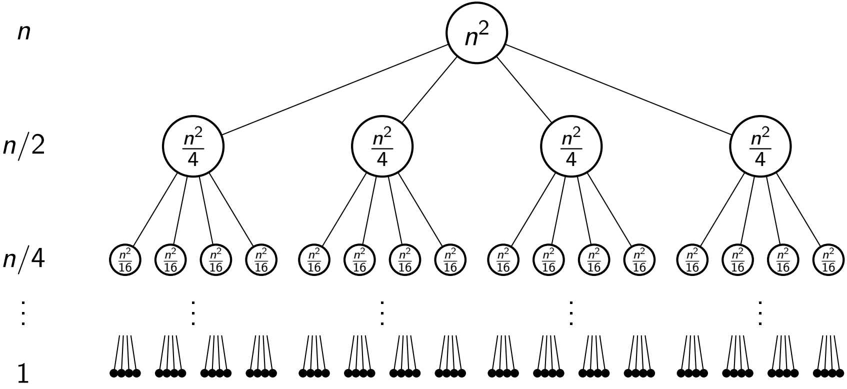
$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of 2.}$$



Another example

Now consider the recurrence

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

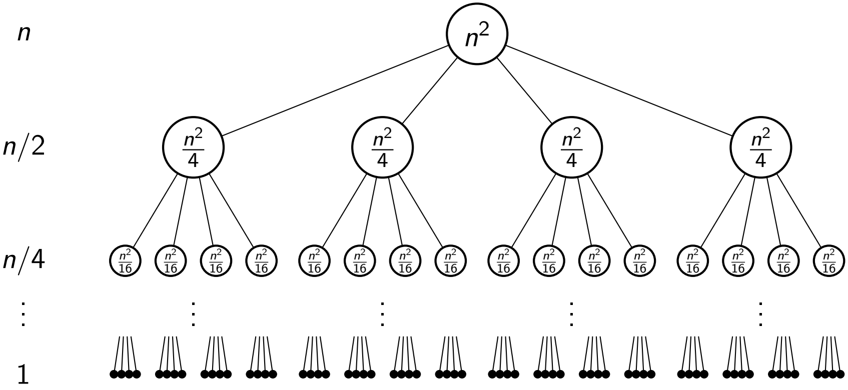


Except for the bottom, level i has 4^{i-1} nodes, each with cost $(n/2^{i-1})^2$.

Another example

Now consider the recurrence

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of 2.}$$



Except for the bottom, level i has 4^{i-1} nodes, each with cost $(n/2^{i-1})^2$. Let $\#(\text{levels}) = t$. Then $n/2^{t-1} = 1$, so we have $t = 1 + \log n$.

Analysing the tree

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of 2.}$$

There are $1 + \log n$ levels in total.
For $i \leq \log n$, level i has 4^{i-1} nodes with cost $(n/2^{i-1})^2$ each.
Level $1 + \log n$ has $4^{\log n} = 2^{2\log n} = n^2$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log n} 4^{i-1} \cdot \left(\frac{n}{2^{i-1}}\right)^2 + n^2$$

Analysing the tree

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

There are $1 + \log n$ levels in total.
For $i \leq \log n$, level i has 4^{i-1} nodes with cost $(n/2^{i-1})^2$ each.
Level $1 + \log n$ has $4^{\log n} = 2^{2 \log n} = n^2$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log n} 4^{i-1} \cdot \left(\frac{n}{2^{i-1}}\right)^2 + n^2 = \sum_{i=1}^{\log n} n^2 + n^2$$

Analysing the tree

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

There are $1 + \log n$ levels in total.
For $i \leq \log n$, level i has 4^{i-1} nodes with cost $(n/2^{i-1})^2$ each.
Level $1 + \log n$ has $4^{\log n} = 2^{2\log n} = n^2$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$\begin{aligned} T(n) &= \sum_{i=1}^{\log n} 4^{i-1} \cdot \left(\frac{n}{2^{i-1}}\right)^2 + n^2 = \sum_{i=1}^{\log n} n^2 + n^2 \\ &= (\log(n) + 1)n^2 \end{aligned}$$

Analysing the tree

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of 2.}$$

There are $1 + \log n$ levels in total.
For $i \leq \log n$, level i has 4^{i-1} nodes with cost $(n/2^{i-1})^2$ each.
Level $1 + \log n$ has $4^{\log n} = 2^{2\log n} = n^2$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$\begin{aligned} T(n) &= \sum_{i=1}^{\log n} 4^{i-1} \cdot \left(\frac{n}{2^{i-1}}\right)^2 + n^2 = \sum_{i=1}^{\log n} n^2 + n^2 \\ &= (\log(n) + 1)n^2 = O(n^2 \log n). \end{aligned}$$

Analysing the tree

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of 2.}$$

There are $1 + \log n$ levels in total.
For $i \leq \log n$, level i has 4^{i-1} nodes with cost $(n/2^{i-1})^2$ each.
Level $1 + \log n$ has $4^{\log n} = 2^{2\log n} = n^2$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$\begin{aligned} T(n) &= \sum_{i=1}^{\log n} 4^{i-1} \cdot \left(\frac{n}{2^{i-1}}\right)^2 + n^2 = \sum_{i=1}^{\log n} n^2 + n^2 \\ &= (\log(n) + 1)n^2 = O(n^2 \log n). \end{aligned}$$

In other words, **every level costs the same**.

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

Guess: $T(n) \le Cn^2 \log n$ for all $n \ge \mathbf{2}$ (C to be determined).

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

Guess: $T(n) \le Cn^2 \log n$ for all $n \ge \mathbf{2}$ (C to be determined).

Base case $n = 2$: We have $T(2) = 4T(1) + 4 = 8$, and $C \cdot 2^2 \log 2 = 4C$.
So $T(2) \le Cn^2 \log n$ whenever $\boldsymbol{C \ge 2}$. ✓

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

Guess: $T(n) \le Cn^2 \log n$ for all $n \ge \mathbf{2}$ (C to be determined).

Base case $n = 2$: We have $T(2) = 4T(1) + 4 = 8$, and $C \cdot 2^2 \log 2 = 4C$.
So $T(2) \le Cn^2 \log n$ whenever $\boldsymbol{C \ge 2}$. ✓

Inductive step: Suppose that for all $2 \le n' \le n-1$, $T(n') \le Cn'^2 \log n'$.
Then we must prove $T(n) \le Cn^2 \log n$.

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

Guess: $T(n) \leq Cn^2 \log n$ for all $n \geq \mathbf{2}$ (C to be determined).

Base case $n = 2$: We have $T(2) = 4T(1) + 4 = 8$, and $C \cdot 2^2 \log 2 = 4C$.
So $T(2) \leq Cn^2 \log n$ whenever $\boldsymbol{C \geq 2}$. ✓

Inductive step: Suppose that for all $2 \leq n' \leq n-1$, $T(n') \leq Cn'^2 \log n'$.
Then we must prove $T(n) \leq Cn^2 \log n$.

By the induction hypothesis, we have

$$T(n) = 4T(n/2) + n^2$$

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

Guess: $T(n) \le Cn^2 \log n$ for all $n \ge \mathbf{2}$ (C to be determined).

Base case $n = 2$: We have $T(2) = 4T(1) + 4 = 8$, and $C \cdot 2^2 \log 2 = 4C$.
So $T(2) \le Cn^2 \log n$ whenever $\boldsymbol{C \ge 2}$. ✓

Inductive step: Suppose that for all $2 \le n' \le n-1$, $T(n') \le Cn'^2 \log n'$.
Then we must prove $T(n) \le Cn^2 \log n$.

By the induction hypothesis, we have

$$T(n) = 4T(n/2) + n^2 \le 4C\left(\frac{n}{2}\right)^2 \log\left(\frac{n}{2}\right) + n^2$$

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

Guess: $T(n) \le Cn^2 \log n$ for all $n \ge \mathbf{2}$ (C to be determined).

Base case $\boldsymbol{n = 2}$: We have $T(2) = 4T(1) + 4 = 8$, and $C \cdot 2^2 \log 2 = 4C$.
So $T(2) \le Cn^2 \log n$ whenever $\boldsymbol{C \ge 2}$. ✓

Inductive step: Suppose that for all $2 \le n' \le n-1$, $T(n') \le Cn'^2 \log n'$.
Then we must prove $T(n) \le Cn^2 \log n$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) &= 4T(n/2) + n^2 \le 4C\left(\tfrac{n}{2}\right)^2 \log\left(\tfrac{n}{2}\right) + n^2 \\ &= Cn^2(\log n - 1) + n^2 \end{aligned}$$

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

Guess: $T(n) \le Cn^2 \log n$ for all $n \ge \mathbf{2}$ (C to be determined).

Base case $n = 2$: We have $T(2) = 4T(1) + 4 = 8$, and $C \cdot 2^2 \log 2 = 4C$.
So $T(2) \le Cn^2 \log n$ whenever $\boldsymbol{C \ge 2}$. ✓

Inductive step: Suppose that for all $2 \le n' \le n-1$, $T(n') \le Cn'^2 \log n'$.
Then we must prove $T(n) \le Cn^2 \log n$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) &= 4T(n/2) + n^2 \le 4C\left(\tfrac{n}{2}\right)^2 \log\left(\tfrac{n}{2}\right) + n^2 \\ &= Cn^2(\log n - 1) + n^2 = Cn^2 \log n + (1 - C)n^2. \end{aligned}$$

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of 2.}$$

Guess: $T(n) \leq Cn^2 \log n$ for all $n \geq \mathbf{2}$ (C to be determined).

Base case $n = 2$: We have $T(2) = 4T(1) + 4 = 8$, and $C \cdot 2^2 \log 2 = 4C$.
So $T(2) \leq Cn^2 \log n$ whenever $\boldsymbol{C \geq 2}$. ✓

Inductive step: Suppose that for all $2 \leq n' \leq n-1$, $T(n') \leq Cn'^2 \log n'$.
Then we must prove $T(n) \leq Cn^2 \log n$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) &= 4T(n/2) + n^2 \leq 4C\left(\tfrac{n}{2}\right)^2 \log\left(\tfrac{n}{2}\right) + n^2 \\ &= Cn^2(\log n - 1) + n^2 = Cn^2 \log n + (1-C)n^2. \end{aligned}$$

This is at most $Cn^2 \log n$ iff $\boldsymbol{C \geq 1}$. ✓

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 4T(n/2) + n^2, \qquad n \text{ is a power of } 2.$$

Guess: $T(n) \leq Cn^2 \log n$ for all $n \geq \mathbf{2}$ (C to be determined).

Base case $n = 2$: We have $T(2) = 4T(1) + 4 = 8$, and $C \cdot 2^2 \log 2 = 4C$.
So $T(2) \leq Cn^2 \log n$ whenever $\boldsymbol{C \geq 2}$. ✓

Inductive step: Suppose that for all $2 \leq n' \leq n-1$, $T(n') \leq Cn'^2 \log n'$.
Then we must prove $T(n) \leq Cn^2 \log n$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) = 4T(n/2) + n^2 &\leq 4C\left(\tfrac{n}{2}\right)^2 \log\left(\tfrac{n}{2}\right) + n^2 \\ &= Cn^2(\log n - 1) + n^2 = Cn^2 \log n + (1 - C)n^2. \end{aligned}$$

This is at most $Cn^2 \log n$ iff $\boldsymbol{C \geq 1}$. ✓

We have proved $T(n) \leq 2n^2 \log n$ for all $n \geq 2$, so $T(n) = O(n^2 \log n)$. □

A third example

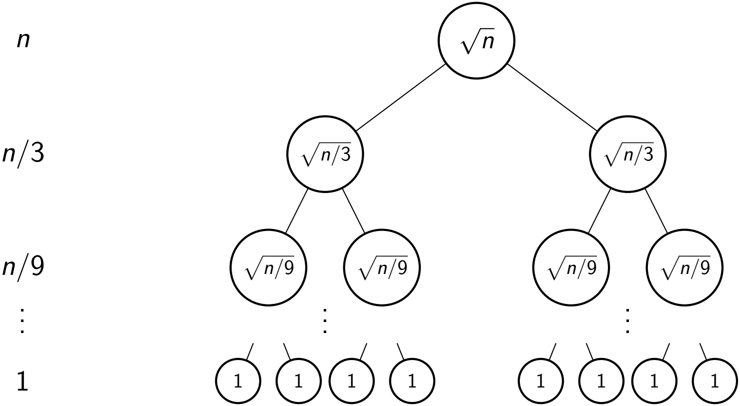
Now consider the recurrence

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

A third example

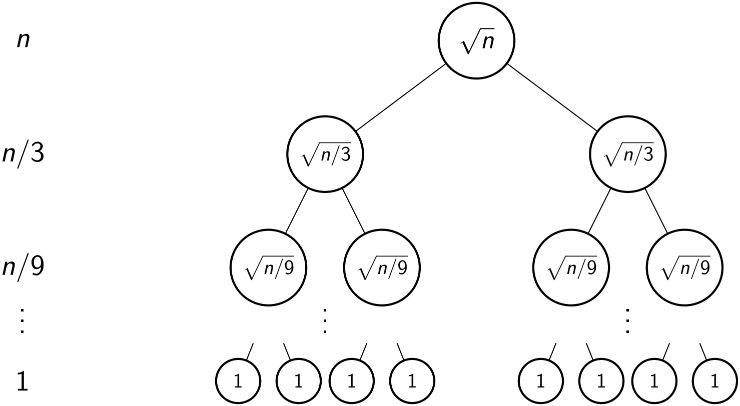
Now consider the recurrence

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$



Now consider the recurrence

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

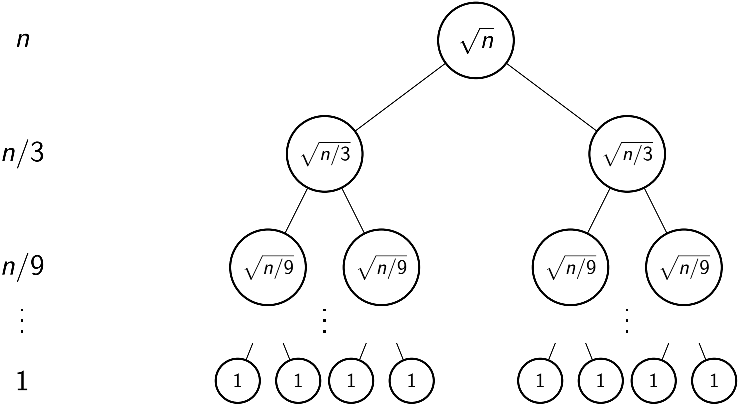


Except for the bottom, level i has 2^{i-1} nodes, each with cost $\sqrt{n/3^{i-1}}$.

A third example

Now consider the recurrence

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$



Except for the bottom, level i has 2^{i-1} nodes, each with cost $\sqrt{n/3^{i-1}}$.
Let $\#(\text{levels}) = t$. Then $n/3^{t-1} = 1$, so we have $t = 1 + \log_3 n$.

Analysing the tree

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

There are $1 + \log_3 n$ levels in total.
For $i \le \log_3 n$, level i has 2^{i-1} nodes with cost $\sqrt{n/3^{i-1}}$ each.
Level $1 + \log_3 n$ has $2^{\log_3 n}$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_3 n} 2^{i-1} \cdot \sqrt{n/3^{i-1}} + 2^{\log_3 n}$$

Analysing the tree

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

There are $1 + \log_3 n$ levels in total.
For $i \leq \log_3 n$, level i has 2^{i-1} nodes with cost $\sqrt{n/3^{i-1}}$ each.
Level $1 + \log_3 n$ has $2^{\log_3 n}$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_3 n} 2^{i-1} \cdot \sqrt{n/3^{i-1}} + 2^{\log_3 n} = \sqrt{n} \sum_{i=1}^{\log_3 n} \left(\frac{2}{\sqrt{3}}\right)^{i-1} + 2^{\log_3 n}.$$

Analysing the tree

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

There are $1 + \log_3 n$ levels in total.
For $i \leq \log_3 n$, level i has 2^{i-1} nodes with cost $\sqrt{n/3^{i-1}}$ each.
Level $1 + \log_3 n$ has $2^{\log_3 n}$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_3 n} 2^{i-1} \cdot \sqrt{n/3^{i-1}} + 2^{\log_3 n} = \sqrt{n} \sum_{i=1}^{\log_3 n} \Big(\frac{2}{\sqrt{3}}\Big)^{i-1} + 2^{\log_3 n}.$$

Since $2 > \sqrt{3}$, this sum is dominated by its last term; formally, we have

$$\sum_{i=1}^{\log_3 n} \Big(\frac{2}{\sqrt{3}}\Big)^{i-1}$$

Analysing the tree

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

There are $1 + \log_3 n$ levels in total.
For $i \leq \log_3 n$, level i has 2^{i-1} nodes with cost $\sqrt{n/3^{i-1}}$ each.
Level $1 + \log_3 n$ has $2^{\log_3 n}$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_3 n} 2^{i-1} \cdot \sqrt{n/3^{i-1}} + 2^{\log_3 n} = \sqrt{n} \sum_{i=1}^{\log_3 n} \Big(\frac{2}{\sqrt{3}}\Big)^{i-1} + 2^{\log_3 n}.$$

Since $2 > \sqrt{3}$, this sum is dominated by its last term; formally, we have

$$\sum_{i=1}^{\log_3 n} \Big(\frac{2}{\sqrt{3}}\Big)^{i-1} = \Big(\frac{2}{\sqrt{3}}\Big)^{\log_3 n - 1} \sum_{j=0}^{\log_3 -1} \Big(\frac{\sqrt{3}}{2}\Big)^{j}$$

Analysing the tree

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

There are $1 + \log_3 n$ levels in total.
For $i \leq \log_3 n$, level i has 2^{i-1} nodes with cost $\sqrt{n/3^{i-1}}$ each.
Level $1 + \log_3 n$ has $2^{\log_3 n}$ nodes with time cost 1 each.

$T(n)$ is the total cost over the whole tree, so

$$T(n) = \sum_{i=1}^{\log_3 n} 2^{i-1} \cdot \sqrt{n/3^{i-1}} + 2^{\log_3 n} = \sqrt{n} \sum_{i=1}^{\log_3 n} \Big(\frac{2}{\sqrt{3}}\Big)^{i-1} + 2^{\log_3 n}.$$

Since $2 > \sqrt{3}$, this sum is dominated by its last term; formally, we have

$$\sum_{i=1}^{\log_3 n} \Big(\frac{2}{\sqrt{3}}\Big)^{i-1} = \Big(\frac{2}{\sqrt{3}}\Big)^{\log_3 n - 1} \sum_{j=0}^{\log_3 -1} \Big(\frac{\sqrt{3}}{2}\Big)^{j} = \Theta\Big(\Big(\frac{2}{\sqrt{3}}\Big)^{\log_3 n}\Big).$$

Analysing the tree (part 2)

$$T(n) = \sqrt{n}\sum_{i=1}^{\log_3 n}\Big(\frac{2}{\sqrt{3}}\Big)^{i-1} + 2^{\log_3 n},$$

$$\sum_{i=1}^{\log_3 n}\Big(\frac{2}{\sqrt{3}}\Big)^{i-1} = \Theta\Big(\Big(\frac{2}{\sqrt{3}}\Big)^{\log_3 n}\Big).$$

Analysing the tree (part 2)

$$T(n) = \sqrt{n} \sum_{i=1}^{\log_3 n} \left(\frac{2}{\sqrt{3}}\right)^{i-1} + 2^{\log_3 n},$$

$$\sum_{i=1}^{\log_3 n} \left(\frac{2}{\sqrt{3}}\right)^{i-1} = \Theta\left(\left(\frac{2}{\sqrt{3}}\right)^{\log_3 n}\right).$$

We have

$$\sqrt{n}\left(\frac{2}{\sqrt{3}}\right)^{\log_3 n} = \sqrt{n} \cdot \frac{2^{\log_3 n}}{3^{(\log_3 n)/2}}$$

Analysing the tree (part 2)

$$T(n) = \sqrt{n} \sum_{i=1}^{\log_3 n} \left(\frac{2}{\sqrt{3}}\right)^{i-1} + 2^{\log_3 n},$$

$$\sum_{i=1}^{\log_3 n} \left(\frac{2}{\sqrt{3}}\right)^{i-1} = \Theta\Big(\left(\frac{2}{\sqrt{3}}\right)^{\log_3 n}\Big).$$

We have

$$\sqrt{n}\left(\frac{2}{\sqrt{3}}\right)^{\log_3 n} = \sqrt{n} \cdot \frac{2^{\log_3 n}}{3^{(\log_3 n)/2}} = \sqrt{n} \cdot \frac{2^{\log_3 n}}{n^{1/2}}$$

Analysing the tree (part 2)

$$T(n) = \sqrt{n} \sum_{i=1}^{\log_3 n} \left(\frac{2}{\sqrt{3}}\right)^{i-1} + 2^{\log_3 n},$$

$$\sum_{i=1}^{\log_3 n} \left(\frac{2}{\sqrt{3}}\right)^{i-1} = \Theta\left(\left(\frac{2}{\sqrt{3}}\right)^{\log_3 n}\right).$$

We have

$$\sqrt{n}\left(\frac{2}{\sqrt{3}}\right)^{\log_3 n} = \sqrt{n} \cdot \frac{2^{\log_3 n}}{3^{(\log_3 n)/2}} = \sqrt{n} \cdot \frac{2^{\log_3 n}}{n^{1/2}} = 2^{\log_3 n}.$$

Analysing the tree (part 2)

$$T(n) = \sqrt{n}\sum_{i=1}^{\log_3 n}\Big(\frac{2}{\sqrt{3}}\Big)^{i-1} + 2^{\log_3 n},$$

$$\sum_{i=1}^{\log_3 n}\Big(\frac{2}{\sqrt{3}}\Big)^{i-1} = \Theta\bigg(\Big(\frac{2}{\sqrt{3}}\Big)^{\log_3 n}\bigg).$$

We have

$$\sqrt{n}\Big(\frac{2}{\sqrt{3}}\Big)^{\log_3 n} = \sqrt{n}\cdot\frac{2^{\log_3 n}}{3^{(\log_3 n)/2}} = \sqrt{n}\cdot\frac{2^{\log_3 n}}{n^{1/2}} = 2^{\log_3 n}.$$

So $T(n) = O(2^{\log_3 n}) = O(n^{1/\log(3)})$, and **the leaves dominate**.

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \le Cn^{1/\log(3)}$ for all $n \ge 1$ (C to be determined).

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

Guess: $T(n) \le Cn^{1/\log(3)}$ for all $n \ge 1$ (C to be determined).

Base case $\boldsymbol{n = 1}$: We have $T(1) = 1 \le C \cdot 1^{1/\log(3)}$ whenever $\boldsymbol{C \ge 1}$. ✓

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \le Cn^{1/\log(3)}$ for all $n \ge 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1 \le C \cdot 1^{1/\log(3)}$ whenever $C \ge 1$. ✓

Inductive step: Suppose that for all $1 \le n' \le n-1$, $T(n') \le Cn'^{1/\log(3)}$.
Then we must prove $T(n) \le Cn^{1/\log(3)}$.

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \le Cn^{1/\log(3)}$ for all $n \ge 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1 \le C \cdot 1^{1/\log(3)}$ whenever $C \ge 1$. ✓

Inductive step: Suppose that for all $1 \le n' \le n-1$, $T(n') \le Cn'^{1/\log(3)}$.
Then we must prove $T(n) \le Cn^{1/\log(3)}$.

By the induction hypothesis, we have

$$T(n) = 2T(n/3) + \sqrt{n}$$

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

Guess: $T(n) \leq Cn^{1/\log(3)}$ for all $n \geq 1$ (C to be determined).

Base case $\boldsymbol{n = 1}$: We have $T(1) = 1 \leq C \cdot 1^{1/\log(3)}$ whenever $\boldsymbol{C \geq 1}$. ✓

Inductive step: Suppose that for all $1 \leq n' \leq n-1$, $T(n') \leq Cn'^{1/\log(3)}$.
Then we must prove $T(n) \leq Cn^{1/\log(3)}$.

By the induction hypothesis, we have

$$T(n) = 2T(n/3) + \sqrt{n} \leq 2C(n/3)^{1/\log(3)} + \sqrt{n}$$

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \leq Cn^{1/\log(3)}$ for all $n \geq 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1 \leq C \cdot 1^{1/\log(3)}$ whenever $\boldsymbol{C \geq 1}$. ✓

Inductive step: Suppose that for all $1 \leq n' \leq n-1$, $T(n') \leq Cn'^{1/\log(3)}$.
Then we must prove $T(n) \leq Cn^{1/\log(3)}$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) &= 2T(n/3) + \sqrt{n} \leq 2C(n/3)^{1/\log(3)} + \sqrt{n} \\ &= \frac{2}{3^{1/\log(3)}} Cn^{1/\log(3)} + \sqrt{n}. \end{aligned}$$

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \leq Cn^{1/\log(3)}$ for all $n \geq 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1 \leq C \cdot 1^{1/\log(3)}$ whenever $C \geq 1$. ✓

Inductive step: Suppose that for all $1 \leq n' \leq n-1$, $T(n') \leq Cn'^{1/\log(3)}$.
Then we must prove $T(n) \leq Cn^{1/\log(3)}$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) = 2T(n/3) + \sqrt{n} &\leq 2C(n/3)^{1/\log(3)} + \sqrt{n} \\ &= \frac{2}{3^{1/\log(3)}} Cn^{1/\log(3)} + \sqrt{n}. \end{aligned}$$

We have $3^{1/\log(3)} = 2^{\log(3)/\log(3)} = 2$, so $T(n) \leq Cn^{1/\log(3)} + \sqrt{n}$.

Formal proof by substitution

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

Guess: $T(n) \leq Cn^{1/\log(3)}$ for all $n \geq 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1 \leq C \cdot 1^{1/\log(3)}$ whenever $C \geq 1$. ✓

Inductive step: Suppose that for all $1 \leq n' \leq n-1$, $T(n') \leq Cn'^{1/\log(3)}$.
Then we must prove $T(n) \leq Cn^{1/\log(3)}$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) = 2T(n/3) + \sqrt{n} &\leq 2C(n/3)^{1/\log(3)} + \sqrt{n} \\ &= \frac{2}{3^{1/\log(3)}} Cn^{1/\log(3)} + \sqrt{n}. \end{aligned}$$

We have $3^{1/\log(3)} = 2^{\log(3)/\log(3)} = 2$, so $T(n) \leq Cn^{1/\log(3)} + \sqrt{n}$...
which isn't quite good enough.

But we know how to deal with this: add a correction term!

Formal proof by substitution (attempt 2)

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \leq Cn^{1/\log(3)} - \sqrt{n}$ for all $n \geq 1$ (C to be determined).

Formal proof by substitution (attempt 2)

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \leq Cn^{1/\log(3)} - \boldsymbol{\sqrt{n}}$ for all $n \geq 1$ (C to be determined).

Base case $\boldsymbol{n = 1}$: We have $T(1) = 1$, and $C \cdot 1^{1/\log(3)} - \sqrt{1} = C - 1$.
So the base case works whenever $\boldsymbol{C \geq 2}$. ✓

Formal proof by substitution (attempt 2)

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

Guess: $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$ for all $n \ge 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1$, and $C \cdot 1^{1/\log(3)} - \sqrt{1} = C - 1$.
So the base case works whenever $C \ge 2$. ✓

Inductive step: Suppose for all $n' \le n - 1$, $T(n') \le Cn'^{1/\log(3)} - \sqrt{n'}$.
Then we must prove $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$.

Formal proof by substitution (attempt 2)

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

Guess: $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$ for all $n \ge 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1$, and $C \cdot 1^{1/\log(3)} - \sqrt{1} = C - 1$.
So the base case works whenever $C \ge 2$. ✓

Inductive step: Suppose for all $n' \le n - 1$, $T(n') \le Cn'^{1/\log(3)} - \sqrt{n'}$.
Then we must prove $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$.

By the induction hypothesis, we have

$$T(n) = 2T(n/3) + \sqrt{n}$$

Formal proof by substitution (attempt 2)

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$ for all $n \ge 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1$, and $C \cdot 1^{1/\log(3)} - \sqrt{1} = C - 1$.
So the base case works whenever $C \ge 2$. ✓

Inductive step: Suppose for all $n' \le n - 1$, $T(n') \le Cn'^{1/\log(3)} - \sqrt{n'}$.
Then we must prove $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$.

By the induction hypothesis, we have

$$T(n) = 2T(n/3) + \sqrt{n} \le 2C(n/3)^{1/\log(3)} - 2\sqrt{n/3} + \sqrt{n}$$

Formal proof by substitution (attempt 2)

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of } 3.$$

Guess: $T(n) \le Cn^{1/\log(3)} - \boldsymbol{\sqrt{n}}$ for all $n \ge 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1$, and $C \cdot 1^{1/\log(3)} - \sqrt{1} = C - 1$.
So the base case works whenever $\boldsymbol{C \ge 2}$. ✓

Inductive step: Suppose for all $n' \le n-1$, $T(n') \le Cn'^{1/\log(3)} - \sqrt{n'}$.
Then we must prove $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) = 2T(n/3) + \sqrt{n} &\le 2C(n/3)^{1/\log(3)} - \boldsymbol{2\sqrt{n/3}} + \sqrt{n} \\ &\le \frac{2}{3^{1/\log(3)}} Cn^{1/\log(3)}. \end{aligned}$$

Formal proof by substitution (attempt 2)

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \leq Cn^{1/\log(3)} - \sqrt{n}$ for all $n \geq 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1$, and $C \cdot 1^{1/\log(3)} - \sqrt{1} = C - 1$.
So the base case works whenever $C \geq 2$. ✓

Inductive step: Suppose for all $n' \leq n - 1$, $T(n') \leq Cn'^{1/\log(3)} - \sqrt{n'}$.
Then we must prove $T(n) \leq Cn^{1/\log(3)} - \sqrt{n}$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) = 2T(n/3) + \sqrt{n} &\leq 2C(n/3)^{1/\log(3)} - 2\sqrt{n/3} + \sqrt{n} \\ &\leq \frac{2}{3^{1/\log(3)}} Cn^{1/\log(3)}. \end{aligned}$$

We have $3^{1/\log(3)} = 2^{\log(3)/\log(3)} = 2$, so this is at most $Cn^{1/\log(3)}$ **always**.

Formal proof by substitution (attempt 2)

$$T(1) = 1, \qquad T(n) = 2T(n/3) + \sqrt{n}, \qquad n \text{ is a power of 3.}$$

Guess: $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$ for all $n \ge 1$ (C to be determined).

Base case $n = 1$: We have $T(1) = 1$, and $C \cdot 1^{1/\log(3)} - \sqrt{1} = C - 1$. So the base case works whenever $C \ge 2$. ✓

Inductive step: Suppose for all $n' \le n - 1$, $T(n') \le Cn'^{1/\log(3)} - \sqrt{n'}$. Then we must prove $T(n) \le Cn^{1/\log(3)} - \sqrt{n}$.

By the induction hypothesis, we have

$$\begin{aligned} T(n) = 2T(n/3) + \sqrt{n} &\le 2C(n/3)^{1/\log(3)} - 2\sqrt{n/3} + \sqrt{n} \\ &\le \frac{2}{3^{1/\log(3)}} Cn^{1/\log(3)}. \end{aligned}$$

We have $3^{1/\log(3)} = 2^{\log(3)/\log(3)} = 2$, so this is at most $Cn^{1/\log(3)}$ **always**.

We have proved $T(n) \le 2n^{1/\log(3)} - \sqrt{n}$ for all n, so $T(n) = O(n^{1/\log(3)})$. □

Recap

These examples fell into three categories:

1. For $T(n) = 2T(n/3) + \sqrt{n}$, the leaves dominated.
2. For $T(n) = 4T(n/2) + n^2$, the levels were all equal.
3. For $T(n) = 3T(n/4) + n/2$, the root dominated.

Recap

These examples fell into three categories:

1. For $T(n) = 2T(n/3) + \sqrt{n}$, the leaves dominated.
2. For $T(n) = 4T(n/2) + n^2$, the levels were all equal.
3. For $T(n) = 3T(n/4) + n/2$, the root dominated.

Wouldn't it be nice if we could put **any** recurrence relation of the form $T(n) = aT(n/b) + f(n)$ into one of those three categories?

Then we could just write the answer down without having to solve it...

Recap

These examples fell into three categories:

1. For $T(n) = 2T(n/3) + \sqrt{n}$, the leaves dominated.
2. For $T(n) = 4T(n/2) + n^2$, the levels were all equal.
3. For $T(n) = 3T(n/4) + n/2$, the root dominated.

Wouldn't it be nice if we could put **any** recurrence relation of the form $T(n) = aT(n/b) + f(n)$ into one of those three categories?

Then we could just write the answer down without having to solve it...

Actually, we can!

The Master Theorem (non-examinable!)

The Master Theorem: Suppose $T(1) = O(1)$ and, for $n > 1$, $T(n) = aT(n/b) + f(n)$ for some constants $a, b > 0$ and some function $f : \mathbb{N} \to \mathbb{R}$. Let $\xi = \log_b a$ be the **critical exponent**. Then:

1. If $f(n) = O(n^{\xi-\varepsilon})$ for some $\varepsilon > 0$, then $T(n) = \Theta(n^\xi)$.
   In other words, **the leaves dominate**.
2. If $f(n) = \Theta(n^\xi)$, then $T(n) = \Theta(n^\xi \log n)$.
   In other words, **the levels are roughly equal**.
3. If $f(n) = \Omega(n^{\xi+\varepsilon})$ for some $\varepsilon > 0$ **and** $af(n/b) = O(f(n))$, then $T(n) = \Theta(f(n))$. In other words, **the root dominates**.

The Master Theorem (non-examinable!)

The Master Theorem: Suppose $T(1) = O(1)$ and, for $n > 1$, $T(n) = aT(n/b) + f(n)$ for some constants $a, b > 0$ and some function $f : \mathbb{N} \to \mathbb{R}$. Let $\xi = \log_b a$ be the **critical exponent**. Then:

1. If $f(n) = O(n^{\xi-\varepsilon})$ for some $\varepsilon > 0$, then $T(n) = \Theta(n^{\xi})$.
   In other words, **the leaves dominate**.
2. If $f(n) = \Theta(n^{\xi})$, then $T(n) = \Theta(n^{\xi} \log n)$.
   In other words, **the levels are roughly equal**.
3. If $f(n) = \Omega(n^{\xi+\varepsilon})$ for some $\varepsilon > 0$ **and** $af(n/b) = O(f(n))$, then $T(n) = \Theta(f(n))$. In other words, **the root dominates**.

The condition $af(n/b) = O(f(n))$ in the last case always holds when f is a polynomial in n, but rules out weird cases like

$$f(n) = \begin{cases} n & \text{if } n \text{ is odd,} \\ n^2 & \text{if } n \text{ is even.} \end{cases}$$

Next time:
Christian's triumphant return!*

*Return may not be triumphant or even physical, rules and restrictions apply.