Recursion Tree Method

In a ***recursion tree***, each node represents the cost of a single subproblem somewhere in the set of recursive function invocations. We sum the costs within each level of the tree to obtain a set of per-level costs, and then we sum all the per-level costs to determine the total cost of all levels of the recursion.

Q1: Use recursion tree method to provide a good guess for the recurrence

$$T(n) = \begin{cases} 3T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + \Theta(n^2) & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$$

Because we know/assume that floors and ceilings usually do not matter when solving recurrences, we create a recursion tree for the recurrence $3T\left(\frac{n}{4}\right) + cn^2$, having written out the implied constant coefficient $c > 0$. **Figures (a-d)** on the following page shows how the recursion tree is derived. For convenience, we assume that n is an exact power of 4 so that all subproblem sizes are integers.

The total cost over all nodes at depth i, for $i = 0,1,2,\dots,\log_4 n - 1$, is $3^i c(n/4^i)^2 = (3/16)^2 cn^2$. The bottom level, at depth $\log_4 n$, has $3^{\log_4 n} = n^{\log_4 3}$ nodes, each contributing cost $T(1)$, for a total cost of $\Theta\left(n^{\log_4 3}\right)$. The total cost would be

$$\begin{aligned} T(n) &= \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + \Theta\left(n^{\log_4 3}\right) \\ &\le \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i cn^2 + \Theta\left(n^{\log_4 3}\right) \\ &= \frac{1}{1 - \left(\frac{3}{16}\right)} cn^2 + \Theta\left(n^{\log_4 3}\right) \\ &= \frac{16}{13} cn^2 + \Theta\left(n^{\log_4 3}\right) \\ &= O(n^2) \end{aligned}$$

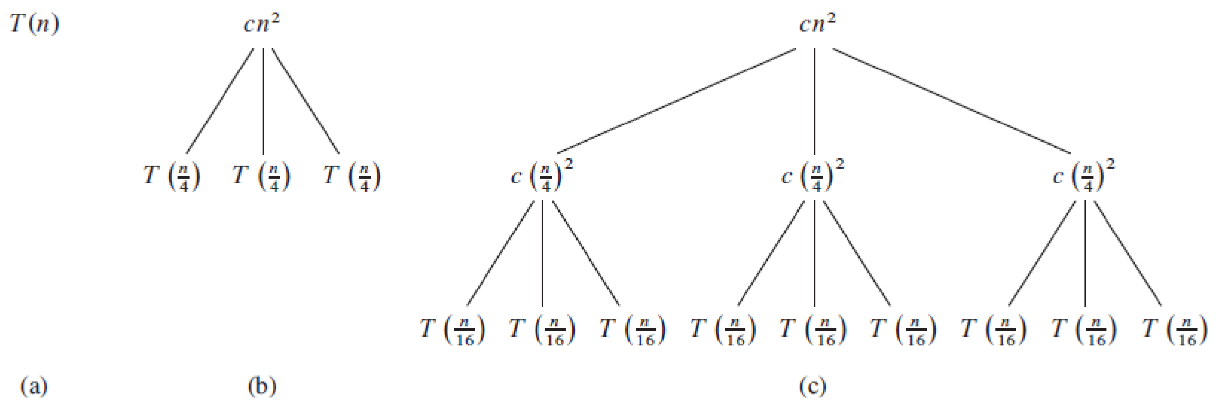
Thus, we have derived a **guess** of $T(n) = O(n^2)$ for our original recurrence $T(n) = 3T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + \Theta(n^2)$.

Note: As the coefficients of cn^2 form a ***decreasing*** *geometric series* $\left(\left|\frac{3}{16}\right| < 1\right)$, we have used the formula

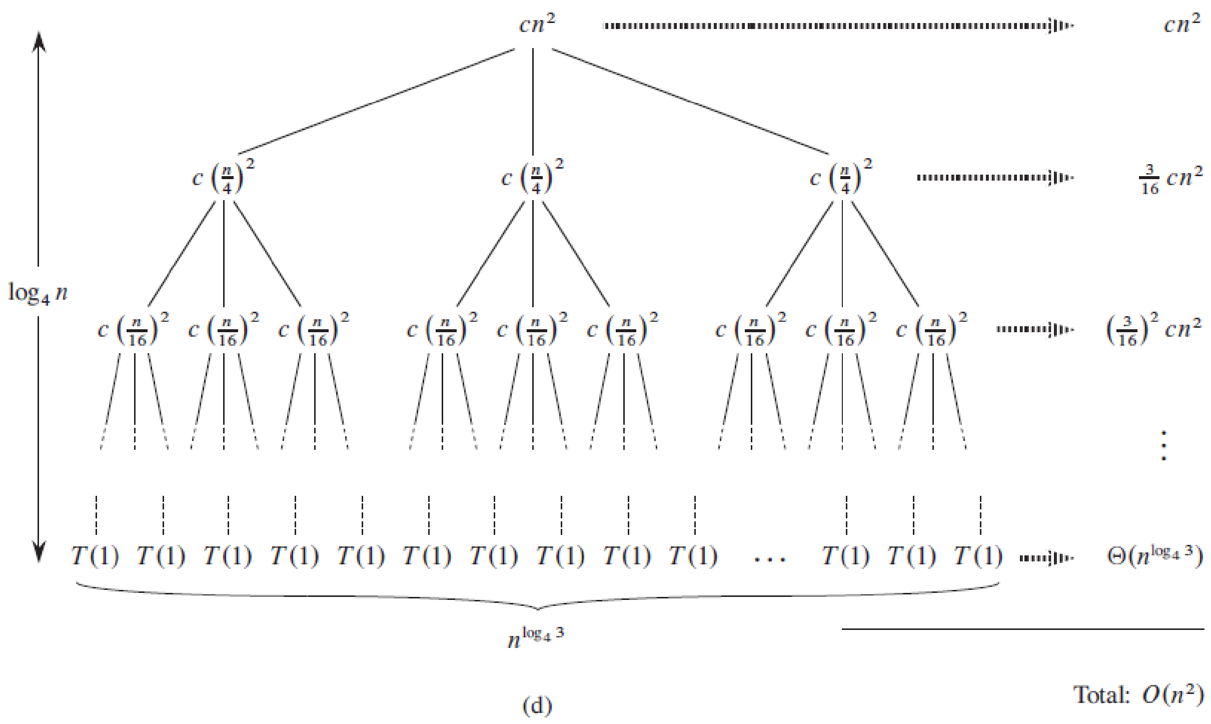
$$\sum_{k=0}^{\infty} x^k = \frac{1}{1-x}$$

for the bounded sum.

Recursion Tree Method



$T(n)$
cn^2
$T\left(\frac{n}{4}\right)$
$c\left(\frac{n}{4}\right)^2$
$T\left(\frac{n}{16}\right)$
(a)
(b)
(c)

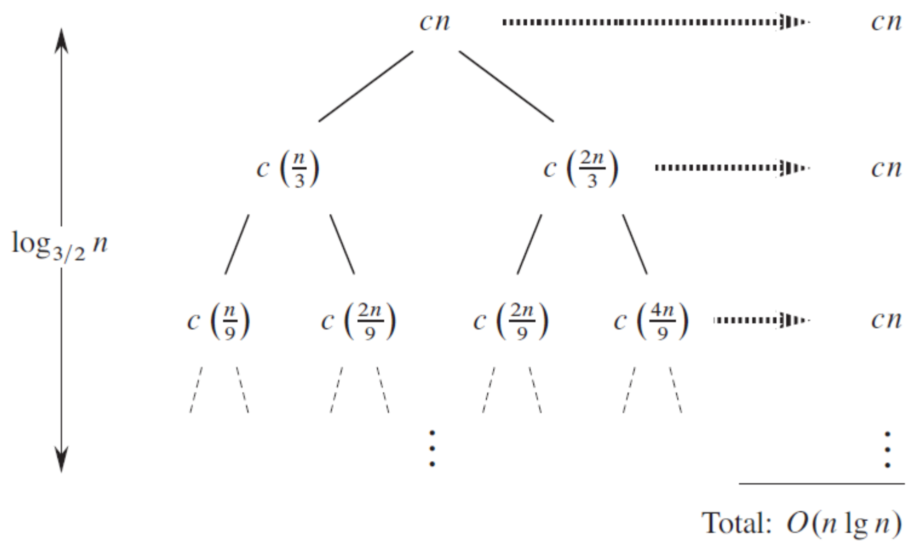


Recursion Tree Method

Q2: Use recursion tree method to provide a good guess for the recurrence

$$T(n) = \begin{cases} T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + O(n) & \text{if } n > 1 \\ 1 & \text{if } n = 1 \end{cases}$$

Following figure shows a recursion tree for the given recurrence


cn
$c\left(\frac{n}{3}\right)$
$c\left(\frac{2n}{3}\right)$
$c\left(\frac{n}{9}\right)$
$c\left(\frac{2n}{9}\right)$
$c\left(\frac{2n}{9}\right)$
$c\left(\frac{4n}{9}\right)$
$\log_{3/2} n$
cn
cn
cn
Total: $O(n \lg n)$

From the tree, we can easily **guess** that $T(n) = O(n \lg n)$.