Recurrences

Dr. Gur Saran Adhar

Reference clrs, Chapter 4, Page 62-
Robert Sedgewick,
Udi Manber Page -

Introduction to Recurrences

A **recurrence** is an equation or inequality that describes a function in terms of its value on smaller inputs.
For example:

$$T(n) = \begin{cases} \Theta(1) & \text{if n} = 1 \\ 2T(n/2) + \Theta(n) & \text{if n} > 1 \end{cases}$$

`Reference clrs page 62`

A simple Example

$$T(n) = 2T(n/2) + cn$$

Let $n = 2^k$ then

$$T(2^k) = 2T(2^{k-1}) + c2^k$$

$$T(2^{k-1}) = \ 2T(2^{k-2}) + c2^{k-1}$$

Therefore,

$$\begin{aligned} T(2^k) &= 2\{2T(2^{k-2}) + c2^{k-1}\} + c2^k \\ &= 2^2T(2^{k-2}) + c2^k + c2^k \\ &= 2^2T(2^{k-2}) + 2.c2^k \end{aligned}$$

Similarly,

$$T(2^{k-2}) = \ 2T(2^{k-3}) + c2^{k-2}$$

Therefore,

$$\begin{aligned} T(2^k) &= 2^2T(2^{k-2}) + 2.c2^k \\ &= 2^2\{2T(2^{k-3}) + c2^{k-2}\} + 2.c2^k \\ &= 2^3T(2^{k-3}) + 3c2^k \end{aligned}$$

Thus,

$$T(2^k) = \ 2^mT(2^{k-m}) + m.c2^k$$

When $k = m$, since $2^k = n$ and therefore $k = \log(n)$, we have,

$$\begin{aligned} T(2^k) &= 2^kT(1) + k.c.2^k \\ &= n.T(1) + \log(n).c.n \\ &= O(n\log(n)) \end{aligned}$$

Another simple Example

$$T(n) = T(n/2) + 1$$

Let $n = 2^k$ then

$$T(2^k) = T(2^{k-1}) + 1$$

$$T(2^{k-1}) = \ T(2^{k-2}) + 1$$

Therefore,

$$\begin{aligned} T(2^k) &= \{T(2^{k-2}) + 1\} + 1\} \\ &= T(2^{k-2}) + 2 \end{aligned}$$

Similarly,

$$T(2^{k-2}) = \ T(2^{k-3}) + 1$$

Therefore,

$$\begin{aligned} T(2^k) &= T(2^{k-2}) + 2 \\ &= \{T(2^{k-3}) + 1\} + 2\} \\ &= T(2^{k-3}) + 3 \end{aligned}$$

Thus,

$$T(2^k) = \ T(2^{k-m}) + m$$

When $k = m$, since $2^k = n$ and therefore $k = \log(n)$, we have,

$$\begin{aligned} T(2^k) &= T(1) + \log(n) \\ &= O(\log(n)) \end{aligned}$$

Three methods

1. **substitution method**: Guess a bound and check by induction
2. **recursion tree method**: convert the equation to a tree whose nodes represent the cost
3. **master method**: for recurrences of the form

$$T(n) = aT(n/b) + f(n)$$

where $a \geq 1, b > 1$ and $f(n)$ is given function.

Substitution method:
For example to solve a recurrence such as:

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

Let $n = 2^m$, that is $\log(n) = m$. The recurrence can be rewritten as:

$$T(2^m) = 2T(2^{m/2}) + m$$

Now let $T(2^m) = S(m)$, then the above recurrence can be rewritten as:

$$S(m) = 2S(m/2) + m$$

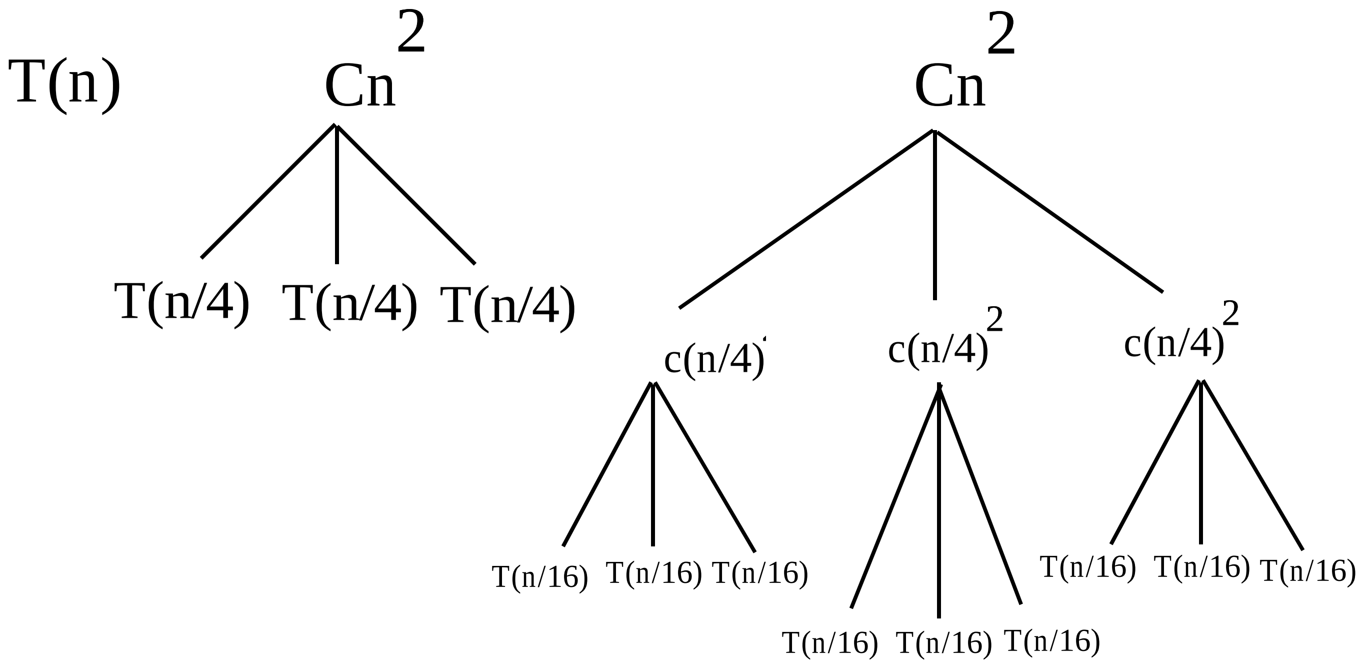
Which solves into (by earlier example):

$$S(m) = O(m \log(m))$$

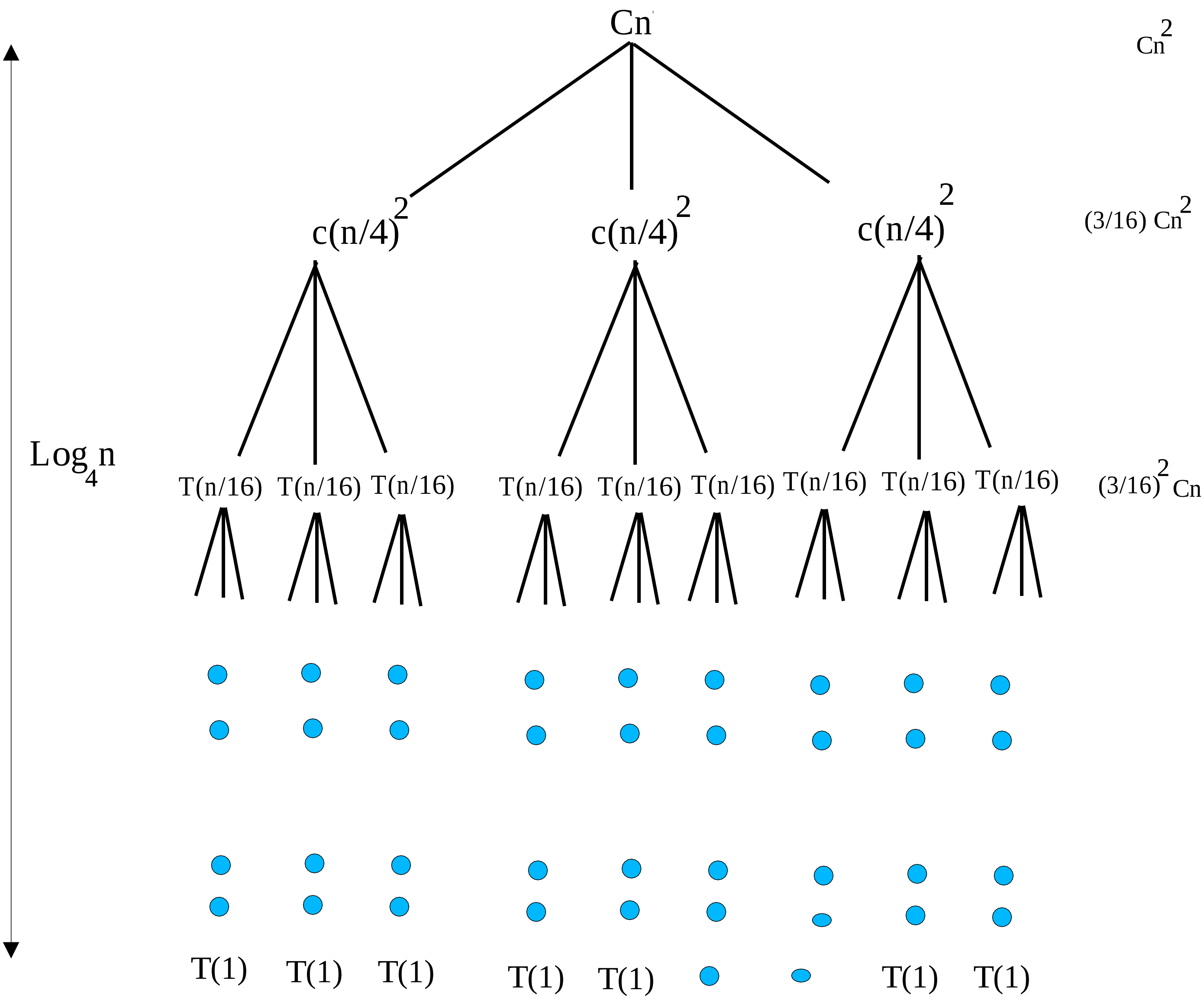
Thus,

$$T(n) = T(2^m) = S(m) = O(\log(n). \log\log(n))$$

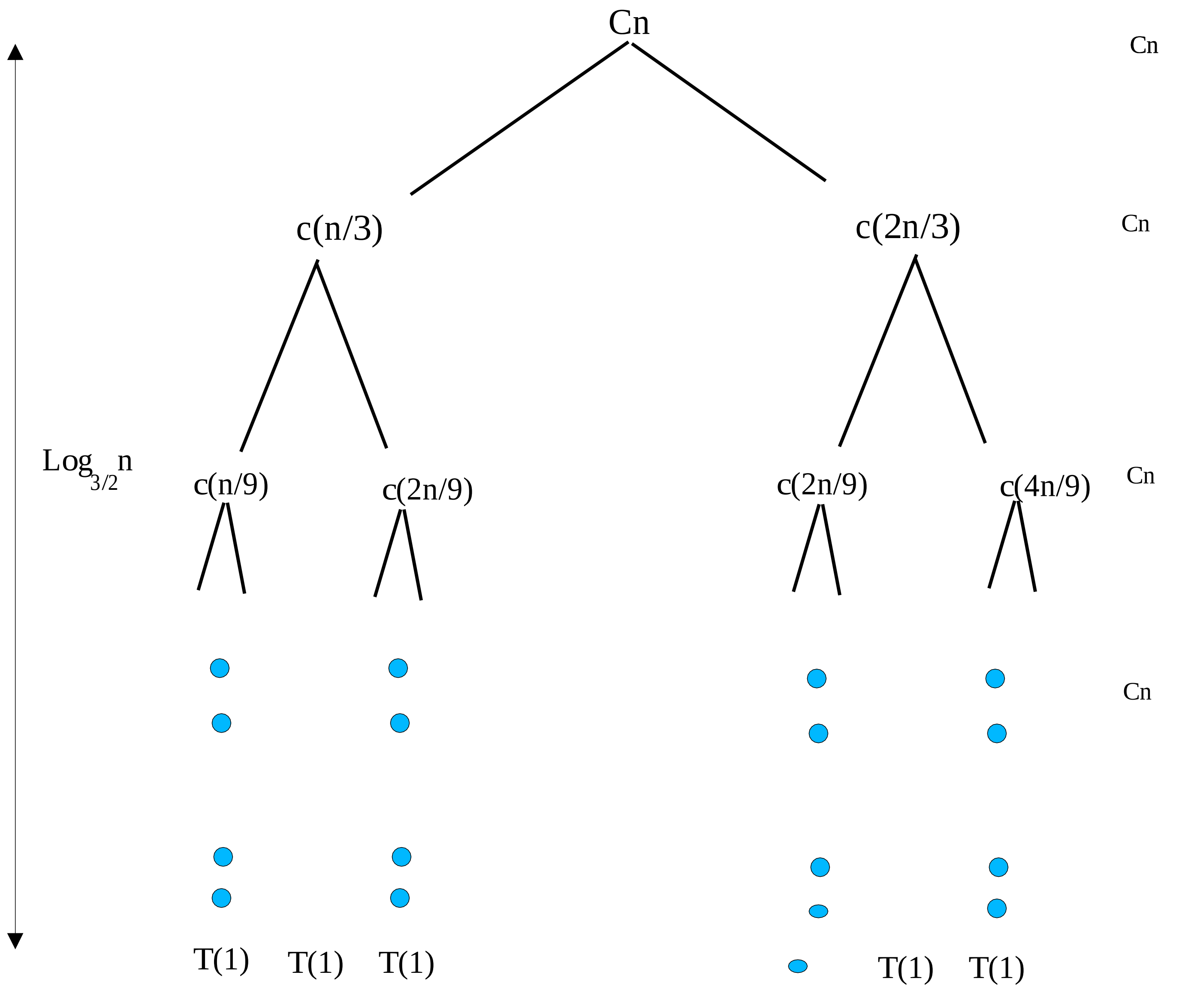
SOLVING $T(n) = 3T(n/4) + O(n)^2$



SOLVING $T(n) = 3T(n/4) + O(n^2)$



SOLVING: $T(n) = T(n/3) + T(2n/3) + cn$



Master Method (reference clrs p-73)

$$T(n) = aT(n/b) + f(n)$$

then

- $$T(n) = \Theta(n^{\log_b a})$$
  if $f(n) = O(n^{\log_b a - \epsilon})$ for some $\epsilon > 0$
- $$T(n) = \Theta(n^{\log_b a} \log n)$$
  if $f(n) = \Theta(n^{\log_b a})$
- $$T(n) = \Theta(f(n))$$
  if $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some $\epsilon > 0$ and if $af(n/b) \leq cf(n)$ for some $c < 1$.