4. Recurrences

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

- ***Substitution method***
- ***Recursion-tree method***
- ***Master method***

  (Assumption: $a \geq 1$, $b > 1$)

Technicalities

- We neglect certain technical details when we state and solve recurrences. A good example of a detail that is often glossed over is the assumption of integer arguments to functions.
- Also, boundary conditions are ignored.
- Besides, we omit floors, ceilings.

4.1 The substitution method : Mathematical induction

- The substitution method for solving recurrence entails two steps:
  1. Guess the form of the solution.
  2. Use mathematical induction to find the constants and show that the solution works.

Example

$$\begin{cases} T(n) = 2T(\lfloor n/2 \rfloor) + n \\ T(1) = 1 \end{cases}$$

- (We may omit the initial condition later.)

1. Guess $T(n) = O(n \log n)$

2a. Assume the bound holds for $\lfloor n/2 \rfloor$, i.e.,
$T(\lfloor n/2 \rfloor) \le c \lfloor n/2 \rfloor \log \lfloor n/2 \rfloor$

2b. Substituting into the recurrence:

$$
\begin{aligned}
T(n) &\leq 2(c\lfloor n/2 \rfloor \log(\lfloor n/2 \rfloor)) + n \\
&\leq cn \log(n/2) + n \\
&= cn \log n - cn \log 2 + n \\
&= cn \log n - cn + n \\
&\leq cn \log n \quad (\text{if } c \geq 1)
\end{aligned}
$$

Initial condition $1 = T(1) < cn \log 1 = 0$

(contradiction)

However, $4 = T(2) < cn \log 2$ (if $c \geq 4$)

Remarks of making a good guess

- $T(n) = 2T(\lfloor n/2 \rfloor + 17) + n$
- We guess $T(n) = O(n \log n)$

  (because the form is similar to what we solved before)

- Another approach:

  Step 1: Making guess provides loose upper and lower bound

  Step 2: Then improve the gap between those bounds

Show the solution of

- Show that the solution to $T(n) = 2T(\lfloor n/2 \rfloor + 17) + n$ is $O(n \lg n)$

- Solution:

Assume $a > 0, b > 0, c > 0$ and $T(n) \le an \lg n - b \lg n - c$

$$\begin{aligned}
T(n) &\le 2\left[a\left(\frac{n}{2}+17\right)\lg\left(\frac{n}{2}+17\right) - b\lg\left(\frac{n}{2}+17\right) - c\right] + n \\
&\le (an + 34a)\lg\left(\frac{n}{2}+17\right) - 2b\lg\left(\frac{n}{2}+17\right) - 2c + n \\
&\le an\lg\left(\frac{n}{2}+17\right) + an\lg(2^{1/a}) + (34a - 2b)\lg\left(\frac{n}{2}+17\right) - 2c \\
&\le an\lg n + (34a - 2b)\lg n - 2c
\end{aligned}$$

- How?

$$\begin{aligned}
& an\lg\left(\frac{n}{2}+17\right)+an\lg(2^{1/a})+(34a-2b)\lg\left(\frac{n}{2}+17\right)-2c \\
\leq \quad & an\lg n - b\lg n - c
\end{aligned}$$

- By

$$\begin{aligned}
& \rightarrow & & n \geq \frac{n}{2}+17, \qquad \text{if } n \geq 34 \\
& \rightarrow & & n \geq \left(\frac{n}{2}+17\right)2^{1/a}, 2^{1/2} \leq 1.5, \qquad \text{if } n \geq 102 \\
& \rightarrow & & 34a-2b \leq -b, \qquad \text{if } b \geq 34a \\
& \rightarrow & & c > 0, -c > -2c \\
T(n) & \leq & & an\lg n - b\lg n - c, \\
\Rightarrow T(n) & = & & O(n\lg n)
\end{aligned}$$

Subtleties

$$T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + 1$$

- Guess $T(n) = O(n)$
- Assume $T(n) \le cn$

$$T(n) \le c\lfloor n/2 \rfloor + c\lceil n/2 \rceil + 1 \le cn + 1 \not\le cn$$

- However, assume $T(n) \le cn - b$

$$T(n) \le (c\lfloor n/2 \rfloor - b) + (c\lceil n/2 \rceil - b) + 1$$

$$\le cn - 2b + 1 \le cn - b \quad \text{(Choose } b \ge 1\text{)}$$

Avoiding pitfalls

$$\begin{cases} T(n) = 2T(\lfloor n/2 \rfloor) + n \\ T(1) = 1 \end{cases}$$

- Assume $T(n) \le O(n)$
- Hence $T(n) \le cn$

  $$T(n) \le 2(c\lfloor n/2 \rfloor) + n \le cn + n = O(n)$$

  (Since c is a constant)
- ***(WRONG!)*** You cannot find such a c.

Changing variables

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

Let $m = \lg n$

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

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

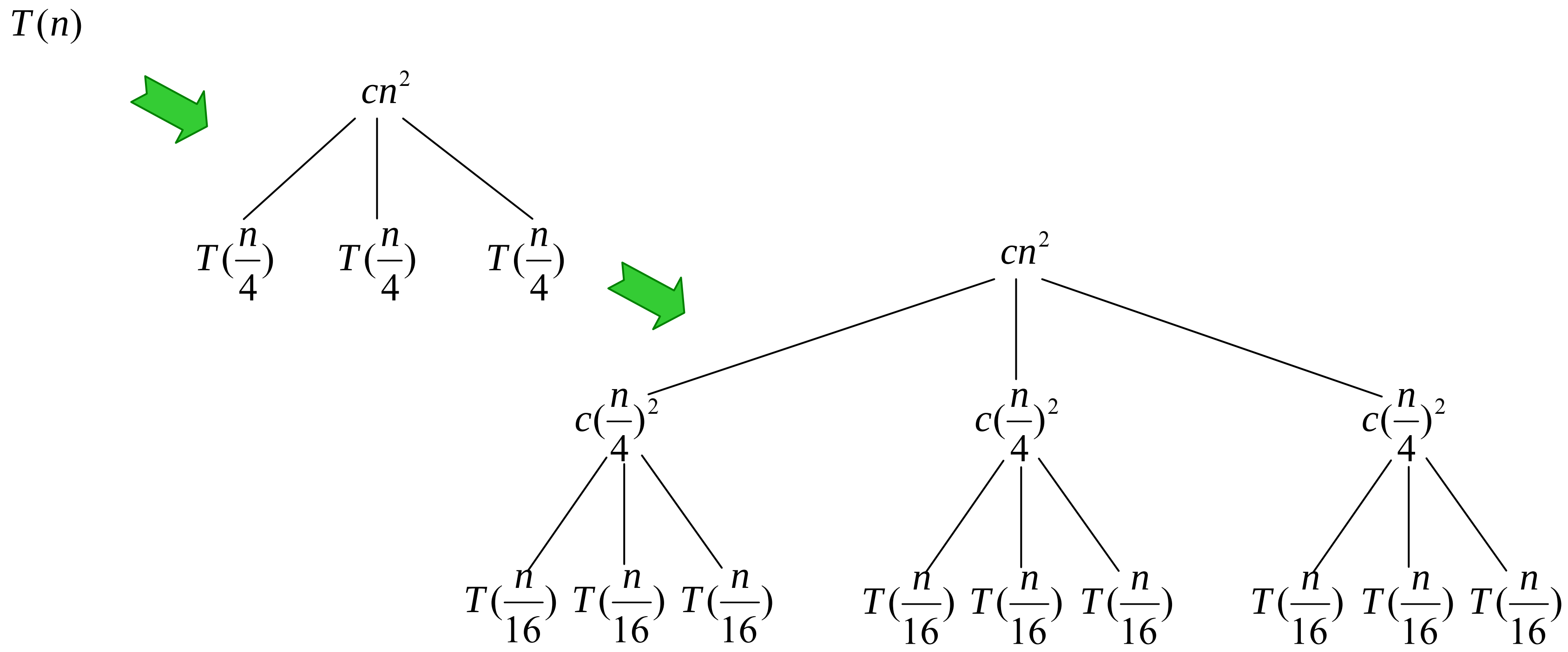
$$\Rightarrow S(m) = O(m \lg m)$$

$$\Rightarrow T(n) = T(2^m) = S(m) = O(m \lg m)$$

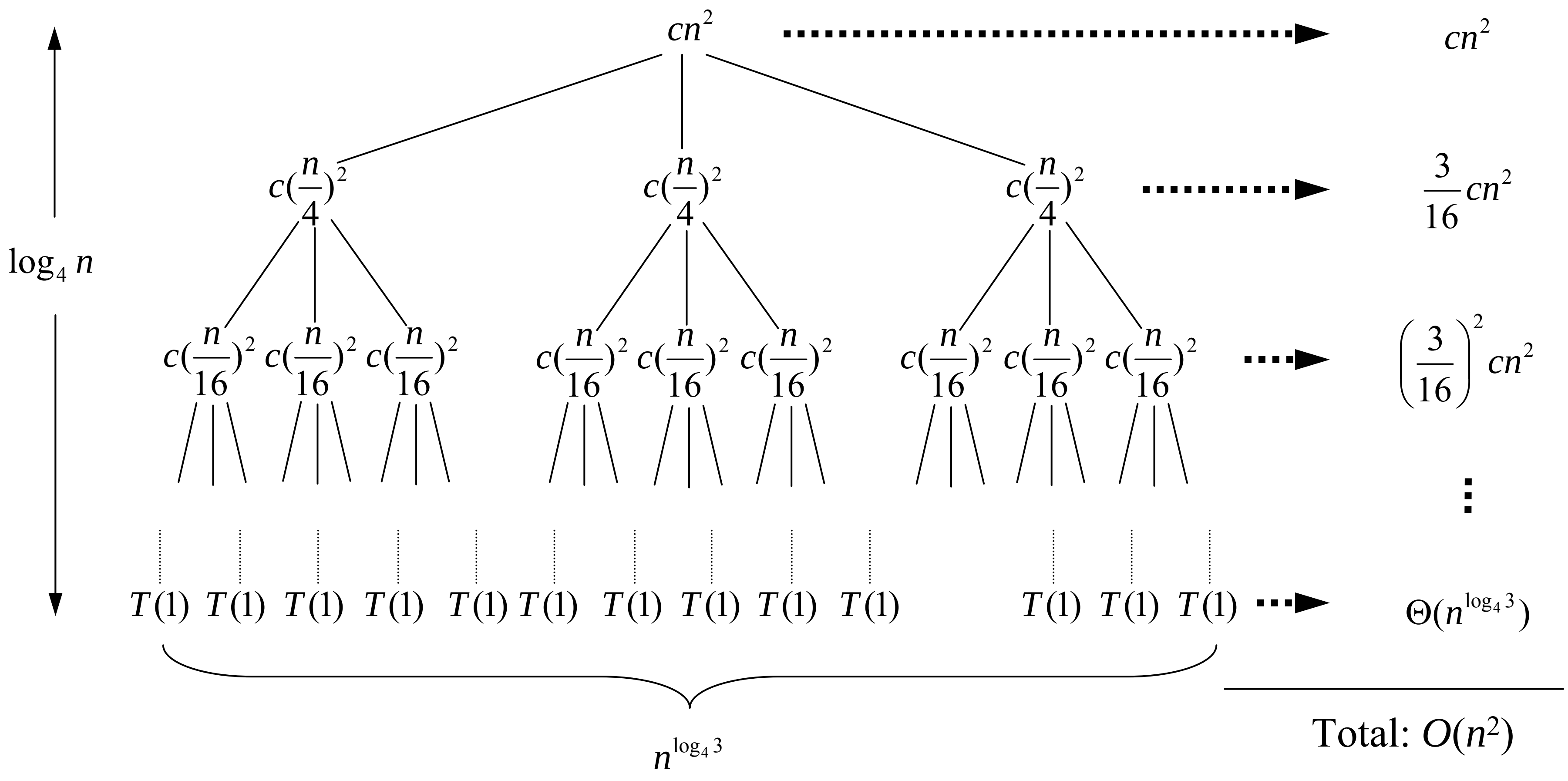
$$= O(\lg n \lg \lg n)$$

4.2 The recursion-tree method

- $T(n) = 3T(\lfloor n/4 \rfloor) + \Theta(n^2)$



$T(n) = 3T(\lfloor n/4 \rfloor) + \Theta(n^2)$



The cost of the entire tree

$$T(n) = cn^2 + \frac{3}{16}cn^2 + \left(\frac{3}{16}\right)^2 cn^2 + \ldots + \left(\frac{3}{16}\right)^{\log_4 n - 1} cn^2 + \Theta(n^{\log_4 3})$$

$$= \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + \Theta\left(n^{\log_4 3}\right)$$

$$= \frac{(3/16)^{\log_4 n} - 1}{(3/16) - 1} cn^2 + \Theta(n^{\log_4 3}).$$

$$T(n) = \sum_{i=0}^{\log_4 n - 1} \left(\frac{3}{16}\right)^i cn^2 + \Theta\left(n^{\log_4 3}\right)$$

$$< \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i cn^2 + \Theta\left(n^{\log_4 3}\right)$$

$$= \frac{1}{1-(3/16)} cn^2 + \Theta\left(n^{\log_4 3}\right)$$

$$= \frac{16}{13} cn^2 + \Theta(n^{\log_4 3})$$

$$= O(n^2)$$

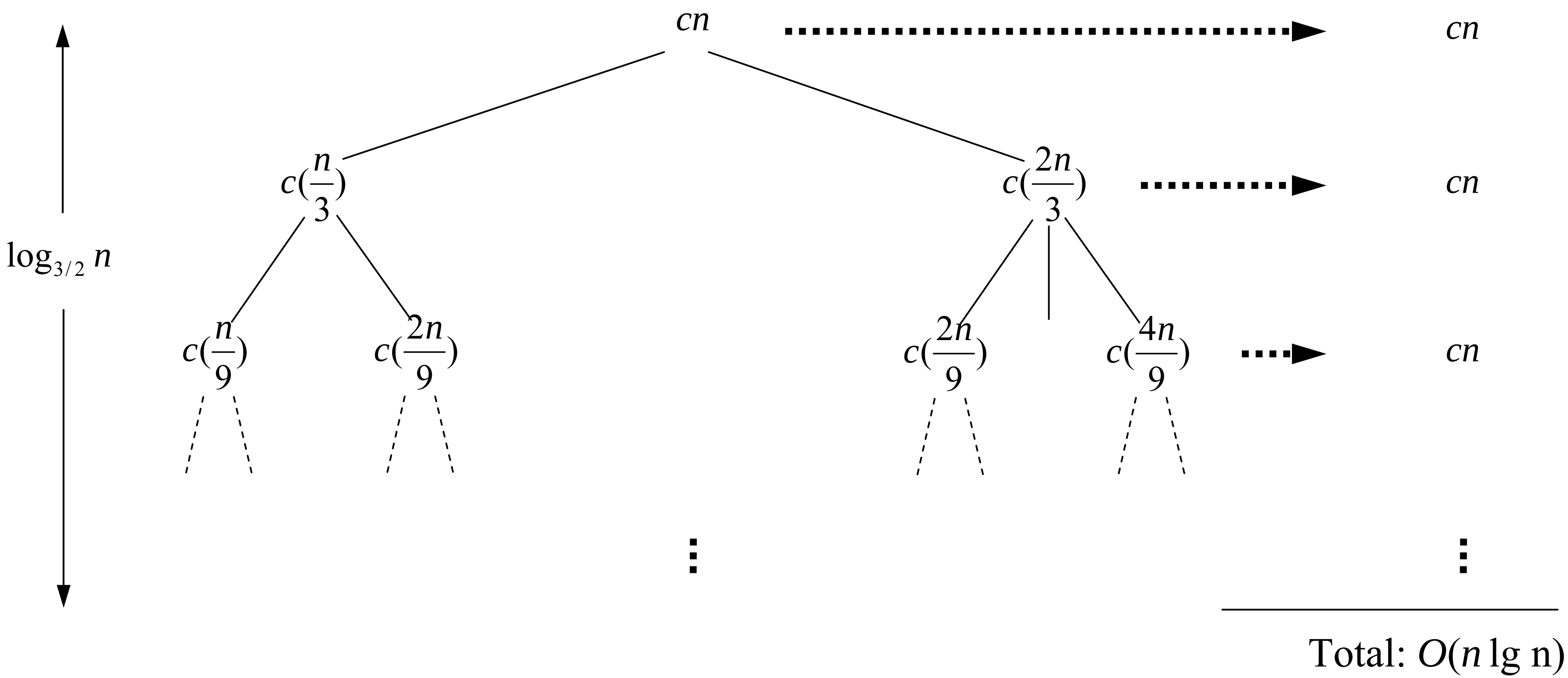
Substitution method

- We want to Show that T(n) $\le dn^2$ for some constant $d > 0$. using the same constant $c > 0$ as before, we have

$$\begin{aligned} T(n) &\le 3T(\lfloor n/4 \rfloor) + cn^2 \\ &\le 3d\lfloor n/4 \rfloor^2 + cn^2 \\ &\le 3d(n/4)^2 + cn^2 \\ &= \frac{3}{16}dn^2 + cn^2 \\ &\le dn^2, \end{aligned}$$

Where the last step holds as long as $d \ge (16/13)c$.

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



Substitution method

$$
\begin{aligned}
T(n) &\le T(n/3) + T(2n/3) + cn \\
&\le d(n/3)\lg(n/3) + d(2n/3)\lg(2n/3) + cn \\
&= (d(n/3)\lg n - d(n/3)\lg 3) + (d(2n/3)\lg n - d(2n/3)\lg(3/2)) + cn \\
&= dn\lg n - d((n/3)\lg 3 + (2n/3)\lg(3/2)) + cn \\
&= dn\lg n - d((n/3)\lg 3 + (2n/3)\lg 3 - (2n/3)\lg 2 + cn \\
&= dn\lg n - dn(\lg 3 - 2/3) + cn \\
&\le dn\lg n,
\end{aligned}
$$

As long as $d \ge c/\lg 3 - (2/3)$

4.3 The master method

- Theorem 4.1 ***(Master theorem)***
  Let a≥1 and b>1 and be constants, let $f(n)$ be a function, and $T(n)$ be defined on the nonnegative integers by the recurrence
  $$T(n) = aT(n / b) + f(n)$$
  where we interpret n/b mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$.

  1. If $f(n) = O(n^{\log_b a-\epsilon})$ for some constant $\varepsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.

  2. If $f(n) = \Theta(n^{\log_b a})$ then $T(n) = \Theta(n^{\log_b a} \log n)$.

  3. If $f(n) = \Omega(n^{\log_b a+\epsilon})$ for some constant $\varepsilon > 0$ and if for $a f(n/b) < c f(n)$ some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$.

- Proof. (In section 4.4 by recursive tree)

- $T(n) = 9T(n/3) + n$

$$a = 9, b = 3, f(n) = n$$

$$n^{\log_3 9} = n^2, f(n) = O(n^{\log_3 9 - 1})$$

$$Case1 \Rightarrow T(n) = \Theta(n^2)$$

- $T(n) = T(2n/3) + 1$

$$a = 1, b = 3/2, f(n) = 1$$

$$n^{\log_{3/2} 1} = n^0 = 1 = f(n),$$

$$Case2 \Rightarrow T(n) = \Theta(\log n)$$

- $T(n) = 3T(n/4) + n\log n$

$$a = 3, b = 4, f(n) = n\log n$$

$$n^{\log_4 3} = n^{0.793}, f(n) = O(n^{\log_4 3+\varepsilon})$$

$Case 3$

Check

$$af(n/b) = 3(\frac{n}{4})\log(\frac{n}{4}) \le \frac{3n}{4}\log n = cf(n)$$

for $c = \frac{3}{4}$, **and sufficiently large** n

$$\Rightarrow T(n) = \Theta(n\log n)$$

- The master method does not apply to the recurrence $T(n) = 2T(n/2) + n \lg n$, even though it has the proper form: $a = 2$, $b = 2$, $f(n) = n \lg n$, and $n^{\log_b a} = n$. It might seem that case 3 should apply, since $f(n) = n \lg n$ is asymptotically larger than $n^{\log_b a} = n$.
- The problem is that it is not polynomially larger.