Asymptotic Orders of Recurrence Functions

Joe Song

March 3, 2020

The lecture notes are mostly based on Chapter 4.3 of Cormen, Leiserson, Rivest, and Stein. Introduction to Algorithms. 3rd Ed. 2009. MIT Press. Cambridge, Massachusetts.

Contents

Introduction

Definition 1 (Recurrence)**.** *A recurrence is an equation or inequality that defines the value of a function by the values of the same function on smaller input sizes.*

Boundary condition: $T(n) = \Theta(1)$ for small n.

Motivation: What are the growth rates of the following two functions?

$$f_1(n) = f_1(n-1) + 2$$

and

$$f_2(n) = f_2\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n$$

Do they have the same growth rate?

They are both linear $\Theta(n)$, though not obviously.

Computer programs can be written recursively leading to these recurrence equations.

That is why we need to study how to evaluate the growth rate of a recurrence function.

1 Substitution method

Two steps:

1. Guess a solution – a closed asymptotic form;
2. Use mathematical induction to prove the solution works by the definition of asymptotic bounds.

Example 1.

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

Guess:

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

Mathematical induction:

Must prove the proposition

$$T(n) \leq cn \lg n$$

but not

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

in the induction step! Because using the asymptotic order within an induction proof is logically inconsistent.

Proof. (by induction)

Hypothesis:

$$T(\lfloor n/2 \rfloor) \leq c \lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor)$$

Induction:

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

Base case(s):

- When $n = 1$, $T(1) = 1 \nleq cn \lg n = 0$ for any $c > 0$!

  We assume $T(1)$ to be a constant 1 for ease of argument but without loss of generality.

- When $n = 2$, $T(2) = 4$. To make $T(2) = 4 \leq c2 \lg 2$, we must have $c \geq 2$;

- We also want to consider $n = 3$ in the base case, because it will be reduced to input of size 1 by recurrence.

$T(3) = 5$. To make $T(3) = 5 \le c3\lg 3$, we must have $c \ge 5/(3\lg 3) = 1.52$.

Thus $c = 2$ will make the base cases work for $n > 1$ as well as the induction.

□

Example 2.

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

Guess:

$$T(n) = \Omega(n \lg n) \tag{1}$$

Mathematical induction attempt 1:

$$T(n) \ge cn \lg n$$

Proof. (by induction)

Hypothesis:

$$T(\lfloor n/2 \rfloor) \ge c\lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor)$$

Induction:

$$\begin{aligned} T(n) &\ge 2c\lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor) + n \\ &\ngeq cn \lg(n/2) + n \quad (\because \lfloor n/2 \rfloor \ngeq n/2) \end{aligned}$$

The induction does not necessarily follow from here. This attempt is thus unsuccessful.

□

Mathematical induction attempt 2:

$$T(n) \ge c(n+1)\lg(n+1) \ (\ge cn \lg n)$$

Proof. (by induction)

Hypothesis:

$$T(\lfloor n/2 \rfloor) \geq c(\lfloor n/2 \rfloor + 1)\lg(\lfloor n/2 \rfloor + 1)$$

Induction:

$$\begin{aligned}
T(n) \geq & 2c(\lfloor n/2 \rfloor + 1)\lg(\lfloor n/2 \rfloor + 1) + n & (2)\\
\geq & 2c[(n+1)/2]\lg[(n+1)/2] + n \quad (\because \lfloor n/2 \rfloor + 1 \geq (n+1)/2) & (3)\\
= & c(n+1)\lg(n+1) - c(n+1)\lg 2 + n & (4)\\
= & c(n+1)\lg(n+1) - c(n+1) + n & (5)\\
\geq & c(n+1)\lg(n+1) \quad (\text{if } -c(n+1) + n \geq 0) & (6)
\end{aligned}$$

Proof of $\lfloor n/2 \rfloor + 1 \geq (n+1)/2$:

When even $n = 2k$ $(k \geq 1)$:

$$\lfloor n/2 \rfloor + 1 = k + 1 > k + \frac{1}{2} = (n+1)/2$$

When odd $n = 2k - 1$ $(k \geq 1)$:

$$\lfloor n/2 \rfloor + 1 = \lfloor (2k-1)/2 \rfloor + 1 = k - 1 + 1 = k = (n+1)/2$$

Now let's determine some $c > 0$

$$\begin{aligned}
& -c(n+1) + n \geq 0\\
\Leftarrow & n \geq \frac{c}{1-c}, 1 - c > 0\\
\Leftarrow & n \geq 1, c = 0.5
\end{aligned}$$

Base case(s): When $n = 1$, $T(1) = 1 \geq c(1+1)\lg(1+1) = 2c$. Thus, $c \leq 0.5$ will make the base case work.

Therefore, $c = 0.5$ will make both the base case and the induction work for $n \geq 1$.

$\square$

Example 3.

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

Guess: $T(n) = O(n)$.

Proposition: $T(n) \leq cn$.

Mathematical Induction:

Proof.

Hypothesis: $T(\lfloor n/2 \rfloor) \leq c\lfloor n/2 \rfloor$ and $T(\lceil n/2 \rceil) \leq c\lceil n/2 \rceil$.

Induction:

$$T(n) \leq c\lfloor n/2 \rfloor + c\lceil n/2 \rceil + 1 = cn + 1 \quad (\because n \in Z)$$

which does not imply $T(n) \leq cn$ for any choice of c.

STOP! Make another guess for $T(n)$!

Base cases:

New proposition: $T(n) \leq cn - b$. Then the induction becomes

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

$\square$

Example 4. *Why we cannot use O or Ω in induction?*

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

Proposition: $T(n) = O(n)$

By hypothesis $T(\lfloor n/2 \rfloor) \le c\lfloor n/2 \rfloor$.

Thus,

$$\begin{aligned} T(n) \le& 2(c\lfloor n/2 \rfloor) + n \\ \le& cn + n \\ =& O(n) \quad \leftarrow \textit{WRONG!} \end{aligned}$$

Remark: *This obviously contradicts* $T(n) = \Omega(n \lg n)$ *proven in Eq. (1). We MUST prove the exact algebraic form of the inductive hypothesis, instead of the asymptotic form.*

Example 5.

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

Renaming $m = \lg n$ *yields*

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

Let

$$S(m) = T(2^m)$$

We get

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

From Example 1, $S(m) = O(m \lg m)$. *Thus*

$$T(n) = S(\lg n) = O(\lg n \lg \lg n) = O(\lg n \lg^{(1)} n)$$

2 Recursion tree method

Roles of a recursion tree method:

- Estimate the order of a recurrence function.
- Approximation/simplification allowed.
- The estimate can be used as a guess required by the substitution method.

The sum of geometric series:

$$a^0 + a^1 + a^2 + \cdots + a^n = \frac{1 - a^{n+1}}{1 - a} \quad a \neq 1$$

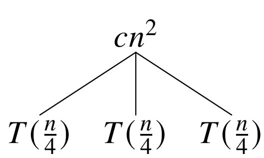
Example 6.

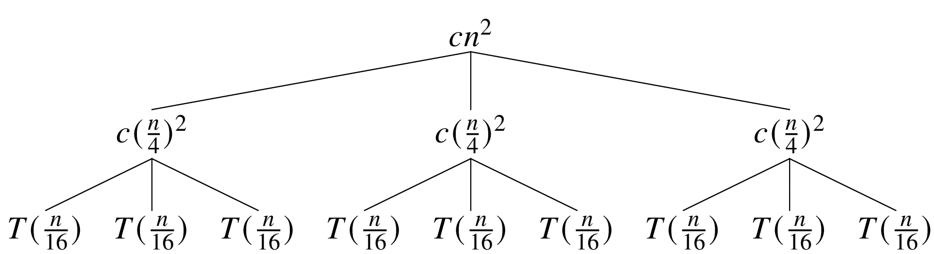
$$T(n) = 3T(\lfloor n/4 \rfloor) + \Theta(n^2)$$

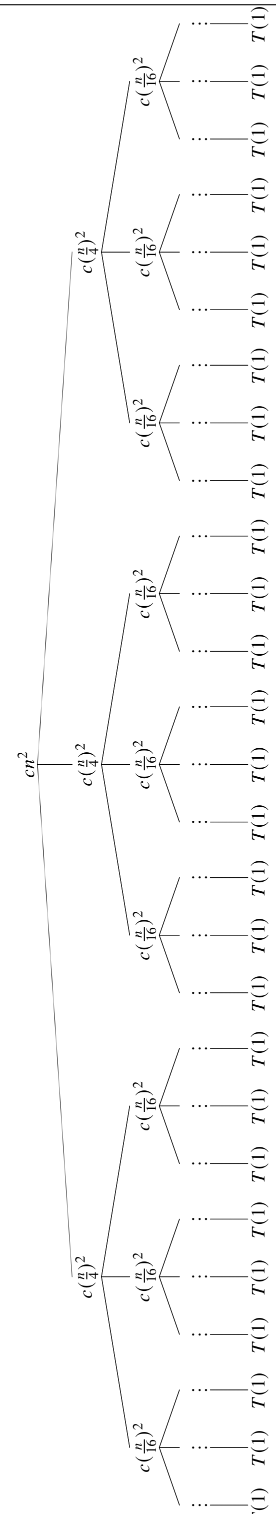
Simplification (that would not change the order of $T(n)$ in this case):

- *We assume n is exact power of 4 to approximate $\lfloor n/4 \rfloor$ by $n/4$*

$T(n)$






cn^2
$c(\frac{n}{4})^2$
$c(\frac{n}{4})^2$
$c(\frac{n}{4})^2$
$c(\frac{n}{16})^2$
$c(\frac{n}{16})^2$
$c(\frac{n}{16})^2$
$c(\frac{n}{16})^2$
$c(\frac{n}{16})^2$
$c(\frac{n}{16})^2$
$c(\frac{n}{16})^2$
$c(\frac{n}{16})^2$
$c(\frac{n}{16})^2$
$T(1)$

The depth of the root node is 0;

The depth of a leaf node is $\log_4 n$*;*

For a node at depth i, its value is approximated by $c(n/4^i)^2$.

There are 3^i *nodes at depth i.*

The sum of all nodes at depth i is

$$c(n/4^i)^2 3^i = cn^2(3/16)^i$$

There are $3^{\log_4 n} = n^{\log_4 3}$ *leaf nodes, each with value* $T(1)$.

From depth 0 *to* $\log_4 n$*, the total approximates* $T(n)$ *by*

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

Remarks:

- *The root node dominates the total;*
- *As* $O(n^2)$ *is the upper bound and* $T(n)$ *is evidently* $\Omega(n^2)$ *by its definition,* $T(n)$ *must be* $\Theta(n^2)$.

Proposition: $T(n)$ is $O(n^2)$.

Proof. We want to show

$$T(n) \le dn^2 \quad \text{for some } d > 0$$

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

The last inequality holds as long as $d \ge \frac{16}{13}c$. $\square$

Example 7.

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

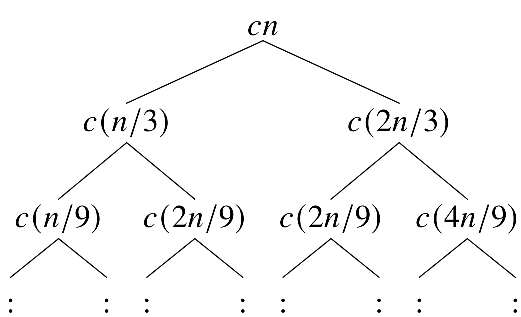


Figure 1: Recursion tree for $T(n) = T(n/3) + T(2n/3) + O(n)$.

There are at most $1 + \log_{3/2} n$ levels by looking at the rightmost branch of the tree.

Each depth has a total of cn.

Therefore, we guess $T(n) = cn(1 + \log_{3/2} n) = O(n \lg n)$.

We will show that $T(n) = O(n \lg n)$ *by applying substitution method on* $T(n) \leq dn \lg n$ *for some constant* d.

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

as long as $d \geq c/(\lg 3 - (2/3))$.

Remarks: Were the tree complete, the number of leaves would be $2^{\log_{3/2} n} = n^{\log_{3/2} 2} = n^{1.7} = \omega(n \lg n)$. *However, as the tree is generally incomplete and by our analysis above,* $T(n) \neq \omega(n \lg n)$.

3 The Master method

The recurrence

$$T(n) = aT(n/b) + f(n), \quad a \geq 1, b > 1, f(n) > 0 \text{ (for } n > n_0 > 0)$$

can be thought of as a model for divide-and-conquer algorithms:

Divide: The problem is divided into a subproblems;

Conquer: Each subproblem has an input size of n/b and is solved in time $T(n/b)$;

Combine: $f(n)$ is the total cost for dividing the problem and combining results of the subproblems.

Example 8 (Merge-Sort)**.**

$$a = 2; \quad b = 2; \quad f(n) = \Theta(n)$$

Theorem 1 (Master Theorem)**.** *Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be an asymptotically positive function, and let $T(n)$ be defined on the nonnegative integers by the recurrence*

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

where we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ can be bounded asymptotically as follows.

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

   Note: $\log_b a - \epsilon = (\log_b a) - \epsilon$

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

3. *If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if*

   $$af(n/b) \leq cf(n)$$

   for some constant $c < 1$ and all sufficiently large n, then $T(n) = \Theta(f(n))$.

   Note: $\log_b a + \epsilon = (\log_b a) + \epsilon$

Remarks 1. *The order of $T(n)$ is determined by $n^{\log_b a}$ and the order of $f(n)$.*

1. *If $n^{\log_b a}$ is "polynomially" larger than $f(n)$, $T(n) = \Theta(n^{\log_b a})$;*

2. *If $n^{\log_b a}$ and $f(n)$ are the same order, then*

   $$T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n);$$

3. *If $f(n)$ is "polynomially" larger than $n^{\log_b a}$, $T(n) = \Theta(f(n))$;*

Limitations of the Master Theorem: *It does not cover all possible asymptotically positive $f(n)$.*

Example 9. *An example that the Master Theorem does not cover.*

$T(n) = T(n/2) + n(2 - \cos n\pi)$. *$T(n)$ does satisfy the asymptotic condition for case 3*

$$n(2 - \cos n\pi) \geq n = \Omega(n^{\log_2 1+0.5}) \quad (\epsilon = 0.5)$$

but it violates the regularity condition. It is shown as follows.

$$af(n/b) = n - \frac{n}{2}\cos\frac{n\pi}{2}$$

The maxima of $af(n/b)$ is $\frac{3}{2}n$, occurring at $n = 2, 6, 10, 14, \ldots$

The minima of $cf(n) = cn(2 - \cos n\pi)$ is cn, occurring at $n = 0, 2, 4, 6, 8, 10, 12, 14, \ldots$ Thus we cannot find $0 < c < 1$ to make

$$af(n/b) < cf(n)$$

for $n = 2, 6, 10, \ldots$. Therefore, function $f(n)$ violates the regularity condition in case 3.

The correct solution (not from the textbook): Let

$$T_1(n) = T_1(n/2) + n \text{ and } T_1(1) = T(1)$$

and

$$T_2(n) = T_2(n/2) + 3n \text{ and } T_2(1) = T(1)$$

By the Master theorem, both $T_1(n)$ and $T_2(n)$ can be solved by case 3 to satisfy

$$T_1(n) = T_2(n) = \Theta(n)$$

It also follows by definitions of $T_1(n)$ and $T_2(n)$ that

$$T_1(n) \le T(n) \le T_2(n)$$

Therefor, we have

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

Example 10.

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

As

$$n = O(n) = O(n^{\log_3 9 - 1}) \quad (\epsilon = 1),$$

we have case 1:

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

Example 11.

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

As

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

we have case 2:

$$T(n) = \Theta(\lg n)$$

Example 12.

$$T(n) = 3T(n/4) + n \lg n$$

Case 3.

$$n \lg n = \Omega(n) = \Omega(n^{\log_4 3 + \log_4 (4/3)})$$

Thus we can choose $\epsilon = \log_4(4/3) > 0$.

Regularity condition:

$$af(n/b) = 3(n/4)\lg(n/4)$$

and

$$cf(n) = cn \lg n$$

To have

$$\begin{aligned}
&3(n/4)\lg(n/4) < cn \lg n \\
\Leftarrow &3/4 \lg(n/4) < c \lg n \quad (n > 0) \\
\Leftarrow &3/4 \lg n \leq c \lg n
\end{aligned}$$

Thus $c = 3/4 \in (0, 1)$ *will satisfy the regularity condition.*

$$T(n) = \Theta(n \lg n)$$

Example 13.

$$T(n) = 2T(n/2) + n \lg n$$

If we apply case 3, we get

$$T(n) = \Theta(n \lg n) \longleftarrow \textit{WRONG!}$$

because $T(n) = n \lg n$ *does not satisfy*

- *case 1.* $T(n) = n \lg n \neq O(n^{\log_2 2 - \epsilon}) = O(n^{1-\epsilon})$
- *case 2.* $T(n) = n \lg n \neq \Theta(n^{\log_2 2}) = \Theta(n)$
- *case 3.* $T(n) = n \lg n \neq \Omega(n^{\log_2 2 + \epsilon}) = \Omega(n^{1+\epsilon})$

This example does not belong to any case of Master Theorem.

Quiz 4 (10 points)

Use the Master method to determine the growth rate of $T(n)$ defined as

$$A: \quad T(n) = 2T(n/2) + n2^n$$
$$B: \quad T(n) = 2T(n/2) + n3^n$$

You must determine a $\epsilon > 0$ for case 1 or 3; If it is case 3, you must show the regularity condition holds.

Correct answer 5 points. Correct proof 5 points.

Quiz 4 (10 points)

Use the Master method to determine the growth rate of $T(n)$ defined as

$$A: \quad T(n) = 2T(n/2) + n2^n$$
$$B: \quad T(n) = 2T(n/2) + n3^n$$

You must determine a $\epsilon > 0$ for case 1 or 3; If it is case 3, you must show the regularity condition holds.

Correct answer 5 points. Correct proof 5 points.

Solution:

A:

Case 3. $T(n) = \Theta(n2^n)$.

As 2^n is $\omega(n)$, we have

$$n2^n = \Omega(n^{\log_2 2+1})$$

where $\epsilon = 1$.

Regularity condition:

$$\begin{aligned}
& af(n/b) = 2(n/2)(2^{n/2}) < cn2^n = cf(n) \\
\Leftrightarrow & n(\sqrt{2})^n < cn2^n \\
\Leftrightarrow & \frac{\sqrt{2}}{2} n(\sqrt{2})^{n-1} < cn2^{n-1} \\
\Leftarrow & \frac{\sqrt{2}}{2} n2^{n-1} < cn2^{n-1} \\
\Leftarrow & c > \frac{\sqrt{2}}{2} \quad (n > 0)
\end{aligned}$$

where we can choose $c = \frac{\sqrt{3}}{2} \in (0, 1)$ to satisfy the regularity condition.

B:

Case 3. $T(n) = \Theta(n3^n)$.

As 3^n is $\omega(n)$, we have

$$n3^n = \Omega(n^{\log_2 2+1})$$

where $\epsilon = 1$.

Regularity condition:

$$\begin{aligned}
&af(n/b) = 2(n/2)(3^{n/2}) < cn3^n = cf(n) \\
\Leftrightarrow & n(\sqrt{3})^n < cn3^n \\
\Leftrightarrow & \frac{\sqrt{3}}{3} n(\sqrt{3})^{n-1} < cn3^{n-1} \\
\Leftarrow & \frac{\sqrt{3}}{3} n3^{n-1} < cn3^{n-1} \\
\Leftarrow & c > \frac{\sqrt{3}}{3} \quad (n > 0)
\end{aligned}$$

where we can choose $c = \frac{\sqrt{3}}{3} \in (0, 1)$ to satisfy the regularity condition.