Growth of Functions

Joe Song

February 11, 2020

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

Contents

1 Motivation

What: In run time analysis of algorithms, we are primarily concerned with the *growth rate* of run time as a function of input size.

Why: This makes such studies generalizable across different computers on which one implements an algorithm.

2 The Θ (theta) notation

Definition 1 ($\Theta(g(n))$ and asymptotically tight bound)**.**

$$\Theta(g(n)) = \{f(n) : \textit{There exist positive constants } c_1, c_2 \textit{ and } n_0 \textit{ such that } 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \textit{ for all } n \ge n_0\} \quad (1)$$

$g(n)$ is an ***asymptotically tight bound*** *of $f(n)$.*

Notations:

1. $f(n) \in \Theta(g(n))$ (mathematically accurate)
2. $f(n) = \Theta(g(n))$ (conventionally used). Here equal sign indicates a set membership relation.

Example 1.

$$\frac{1}{2}n^2 - 3n$$

is $\Theta(n^2)$*, or equivalently in a different notation*

$$\frac{1}{2}n^2 - 3n = \Theta(n^2)$$

Need to find $c_1, c_2, n_0 > 0$ *such that*

$$c_1 n^2 \le \frac{1}{2}n^2 - 3n \le c_2 n^2 \quad \textit{for any } n \ge n_0$$

The first inequality requires

$$(\frac{1}{2} - c_1)n \ge 3$$

One can take $c_1 = 1/4$*, which leads to* $n \ge 12$*, to make the inequality hold.*

The second inequality requires

$$(c_2 - \frac{1}{2})n \ge -3$$

One can take $c_2 = 1$*, which leads to* $n \ge -6$*, to make the inequality hold.*

To make both inequalities hold, one can take

$$c_1 = 1/4, c_2 = 1, n_0 = 12$$

Example 2. $6n^3 \ne \Theta(n^2)$

By contradiction: The definition of Θ *requires*

$$6n^3 \le c_2 n^2 \quad \textit{for any } n \ge n_0$$

implying

$$n \le \frac{c_2}{6} \quad \textit{for any given } c_2 > 0$$

which contradicts the $n \ge n_0$ *requirement.*

2.1 Summary of proof approach

General steps:

1. Derive an n-greater-than inequality in the form of $n \geq n_0$,
2. Do not solve for c_1 or c_2
3. In the process of deriving $n \geq n_0$, determine some values of n_0, c_1, and c_2 to make the inequality happen.

2.2 Polynomials

Example 3. *$f(n) = an^2 + bn + c$, $(a > 0)$ is $\Theta(n^2)$.*

1. $f(n) \geq c_1 n^2 \geq 0$

$$an^2 + bn + c \geq c_1 n^2 \geq 0 \tag{2}$$
$$\Leftarrow an^2 + bn - |c|n \geq c_1 n^2 \quad (n \geq 1, c_1 > 0) \tag{3}$$
$$\Leftarrow (a - c_1)n^2 + (b - |c|)n \geq 0 \tag{4}$$
$$\Leftarrow n \geq \frac{|c| - b}{a - c_1}, c_1 < a \tag{5}$$
$$\Leftarrow n \geq n_1, n_1 = \max\left\{1, 2\frac{|c| - b}{a}\right\}, c_1 = \frac{a}{2} \tag{6}$$

2. $f(n) \leq c_2 n^2$

$$an^2 + bn + c \leq c_2 n^2 \tag{7}$$
$$\Leftarrow an^2 + |b|n^2 + |c|n^2 \leq c_2 n^2 \quad (n \geq 1, c_2 > 0) \tag{8}$$
$$\Leftarrow n \geq 1, c_2 \geq a + |b| + |c| \tag{9}$$
$$\Leftarrow n \geq n_2, n_2 = 1, c_2 = a + |b| + |c| \tag{10}$$

To summarize, $f(n) = \Theta(n^2)$ *if*

$$c_1 = a/2,\ c_2 = a + |b| + |c|,\ n_0 = \max\left\{1, 2\frac{|b| + |c|}{a}\right\}$$

Theorem 1. *Polynomial function of n*

$$p(n) = \sum_{i=0}^{d} a_i n^i, \quad (a_d > 0)$$

has

$$p(n) = \Theta(n^d)$$

Proof.

1. $p(n) \geq c_1 n^d$ for some $c_1 > 0$

$$\begin{aligned}\sum_{i=0}^{d} a_i n^i &\geq a_d n^d - \sum_{i=0}^{d-1} |a_i| n^i \\ &\geq a_d n^d - \sum_{i=0}^{d-1} |a_i| n^{d-1} \quad (\text{for } n \geq 1)\end{aligned}$$

Then we solve n for

$$a_d n^d - \sum_{i=0}^{d-1} |a_i| n^{d-1} \geq c_1 n^d$$

and get

$$n \geq \frac{\sum_{i=0}^{d-1} |a_i|}{a_d - c_1} \quad \text{provided } a_d - c_1 > 0$$

We can pick

$$c_1 = a_d/2, \quad n_1 = \max\left\{1, \frac{2\sum_{i=0}^{d-1}|a_i|}{a_d}\right\}$$

2. $p(n) \le c_2 n^d$ for some $c_2 > 0$.

$$\begin{aligned}\sum_{i=0}^{d} a_i n^i &\le \sum_{i=0}^{d} |a_i| n^i \\ &\le \sum_{i=0}^{d} |a_i| n^{d} \quad (\text{for } n \ge 1) \\ &= \left(\sum_{i=0}^{d} |a_i|\right) n^d\end{aligned}$$

Let $c_2 = \sum_{i=0}^{d} |a_i|$ and $n_2 = 1$, we have

$$\sum_{i=0}^{d} a_i n^i \le c_2 n^d \quad (n \ge n_2)$$

Overall, we pick $n_0 = \max(n_1, n_2)$. Thus we have proved that

$$0 \le c_1 n^d \le p(n) \le c_2 n^d \quad \text{for any } n \ge n_0$$

□

3 The O (big-oh) notation

Definition 2 ($O(g(n))$ and asymptotically upper bound)**.**

$$O(g(n)) = \{f(n) : \textit{There exist positive constants } c \textit{ and } n_0 \textit{ such that } 0 \le f(n) \le cg(n) \textit{ for all } n \ge n_0\} \quad (11)$$

$g(n)$ is an ***asymptotic upper bound*** *of $f(n)$.*

Properties of O-notation:

- $\Theta(g(n)) \subseteq O(g(n))$.
- $f(n) = \Theta(g(n))$ implies $f(n) = O(g(n))$.
- O-notation is used to describe the worst-case running time of an algorithm.

Example 4. *$an^2 + bn + c = \Theta(n^2)$ $(a > 0)$ implies $an^2 + bn + c = O(n^2)$.*

Example 5. *(Student exercise)*

Is $an + b$ $O(n^2)$*?*

Answer: Yes. Take $c = a + |b|$ and $n_0 = 1$.

4 The Ω (big-omega) notation

Definition 3 ($\Omega(g(n))$ and asymptotically lower bound)**.**

$$\Omega(g(n)) = \{f(n) : \textit{There exist positive constants } c \textit{ and } n_0 \textit{ such that } 0 \le cg(n) \le f(n) \textit{ for all } n \ge n_0\} \tag{12}$$

*$g(n)$ is an **asymptotically lower bound** of $f(n)$.*

Theorem 2. *For any two functions $f(n)$ and $g(n)$, we have*

$$f(n) = \Theta(g(n))$$

if and only if

$$f(n) = O(g(n)) \textit{ and } f(n) = \Omega(g(n))$$

Example 6.

$$an^2 + bn + c = \Theta(n^2), \quad (a > 0)$$

implies

$$an^2 + bn + c = O(n^2)$$

and

$$an^2 + bn + c = \Omega(n^2)$$

Example 7. *Insertion-Sort has the best-case running time of $\Omega(n)$ and the worst-case running time of $O(n^2)$. It implies the running time of Insertion-Sort is between $\Omega(n)$ and $O(n^2)$.*

5 o (little-oh) notation

Definition 4 ($o(g(n))$)**.**

$$o(g(n)) = \{f(n) : \textit{ for ANY positive constant } c \textit{ there exists some constant } n_0 > 0 \textit{ such that } 0 \le f(n) < cg(n) \textit{ for all } n \ge n_0\} \quad (13)$$

Interpretation: Little omega provides a loose asymptotic upper bound.

Property: $f(n) \in o(g(n))$ if and only if

$$\lim_{n \to +\infty} \frac{f(n)}{g(n)} = 0 \quad (14)$$

Example 8.

$$2n = o(n^2)$$

$$2n < cn^2 \Leftarrow n > 2/c, \forall c > 0$$

So for any c, we can always find $n_0 = \lceil 1 + 2/c \rceil$.

Example 9.

$$2n^2 \neq o(n^2)$$

Method 1:

$$2n^2 < cn^2 \implies c > 2$$

The inequality will hold for some c but not any $c > 0$, violating the definition of little-oh. Thus $2n^2 \neq o(n^2)$.

Method 2:

$$\lim_{n \to +\infty} \frac{2n^2}{n^2} = 2 \neq 0$$

which does not satisfy the property of little-oh in Eq. (14).

Student exercise: Is it true that

Example 10.

$$2n^3 = o(n^2)$$

Justify your answer.

Answer: No.

$$2n^3 < cn^2 \implies n < c/2$$

So no n_0 *can be found such that the left inequality will hold for ANY* $n > n_0$. *Thus* $2n^3 \neq o(n^2)$.

5.1 Logarithm and polynomial running time

The linear-logarithm inequality:

$$x - 1 \geq \ln x \quad (x > 0) \tag{15}$$

Proof. When $x \geq 1$, we have

$$x - 1 = \int_1^x 1 \; dt \tag{16}$$

$$\ln x = \int_1^x \frac{1}{t} \; dt \tag{17}$$

As

$$1 \geq 1/t \quad (1 \leq t \leq x) \tag{18}$$

and by the property of integrable functions, we have

$$\int_1^x 1 \; dt \geq \int_1^x \frac{1}{t} \; dt \tag{19}$$

Therefore,

$$x - 1 \geq \ln x \quad (x \geq 1) \tag{20}$$

The intuition is as follows. Both functions start at $x = 1$ with the same value of 0. As x increases, $x - 1$ grows at a rate of 1 faster than $\ln x$ at a rate of ≤ 1, we must have the former is no less than the latter.

When $0 < x \le 1$, we have

$$x - 1 = \int_x^1 -1 \, dt \tag{21}$$

$$\ln x = \int_x^1 -\frac{1}{t} \, dt \tag{22}$$

As

$$-1 \ge -1/t \quad (0 < x \le t \le 1) \tag{23}$$

and by the property of integrable functions, we have

$$\int_x^1 -1 \, dt \ge \int_x^1 -\frac{1}{t} \, dt \tag{24}$$

Thus we have

$$x - 1 \ge \ln x \quad (0 < x \le 1) \tag{25}$$

The intuition is as follows. Both functions start at $x = 1$ with the same value of 0. As x decreases, $x - 1$ reduces at a rate of 1 slower than $\ln x$ at a rate of ≥ 1, we must have the former is no less than the latter.

Combining both Eq. (20) and (25), we have

$$x - 1 \ge \ln x \quad (x > 0) \tag{26}$$

which is our original claim in Eq. (15). □

Example 11. *Show that* $\lg n \equiv \log_2 n = o(n)$.

Proof. (Option 1: By definition of little oh).

$$\lg n < cn \qquad \text{for all } c > 0, n > n_0$$

$$\Leftrightarrow \frac{1}{2} \lg n < \frac{1}{2} cn$$

$$\Leftrightarrow \lg n^{\frac{1}{2}} < \frac{1}{2}cn$$

By the linear-logarithm inequality (15)

$$\ln x \le x - 1 < x, \quad x > 0$$

we have

$$\lg x \le (x-1)\lg e < x \lg e, \quad x > 0$$

Thus, we have

$$\Leftarrow \lg n^{\frac{1}{2}} < \sqrt{n}\lg e < \frac{1}{2}cn$$

$$\Leftarrow n > \left(\frac{2\lg e}{c}\right)^2$$

□

Proof. (Option 2: By limit). Using the L'Hôpital's rule, we have

$$\lim_{n\to\infty}\frac{\lg n}{n} = \lim_{n\to\infty}\frac{1/n}{\ln 2} = 0$$

□

Example 12. *Show* $\lg^k n = o(n^\epsilon)$ $(k \ge 0, \epsilon > 0)$.

Proof.

$$\lg^k n < cn^\epsilon \Leftarrow$$
$$\lg n < c^{1/k} n^{\epsilon/k} \Leftarrow$$
$$(\epsilon/k)\lg n < (\epsilon/k)c^{1/k} n^{\epsilon/k} \Leftarrow$$

$$\lg n^{\epsilon/k} < (\epsilon/k)c^{1/k}n^{\epsilon/k} \Leftarrow$$

$$m = n^{\epsilon/k}, d = (\epsilon/k)c^{1/k}$$

$$\lg m < dm$$

From Example 11, we know that the above is true for any $d > 0$ as long as

$$m > \left(\frac{2\lg e}{d}\right)^2$$

Solving for n, we get

$$n > \left(\frac{2k\lg e}{\epsilon c^{1/k}}\right)^{2k/\epsilon} = n_0$$

Therefore, for any $c > 0$, we can find n_0 to satisfy the definition

$$\lg^k n < cn^\epsilon$$

when $n > n_0$. □

6 ω (little-omega) notation

Definition 5 ($\omega(g(n))$).

$$\omega(g(n)) = \{f(n) : \textit{for ANY positive constant } c \textit{ there exists some constant } n_0 > 0 \textit{ such that } 0 \le cg(n) < f(n) \textit{ for all } n \ge n_0\} \quad (27)$$

Definition 6 ($\omega(g(n))$)**.** *$f(n) \in \omega(g(n))$ if and only if $g(n) \in o(f(n))$.*

Interpretation: Little omega provides an asymptotical lower bound which is not tight.

Property: $f(n) \in \omega(g(n))$ if and only if

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

Example 13.

$$\frac{n^2}{2} = \omega(n)$$

Example 14.

$$\frac{n^2}{2} \neq \omega(n^2)$$

7 Asymptotic notations in equations and inequalities

Set membership: when an asymptotic notation stands alone on the right-hand side of an equation.

Example 15. $n = O(n^2)$.

Anonymous function: when in a formula, an asymptotic notation stands for some anonymous function that we do not care to name.

Example 16.

$$2n^2 + 3n + 1 = 2n^2 + \Theta(n)$$

means that

$$2n^2 + 3n + 1 = 2n^2 + f(n)$$

for some function $f(n)$ in the set $\Theta(n)$. In this case

$$f(n) = 3n + 1$$

8 Factorials

Stirling's approximation:

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \left[1 + \Theta\left(\frac{1}{n}\right)\right] \tag{28}$$

Another approximation of $n!$:

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\alpha_n} \quad n \geq 1 \tag{29}$$

where

$$\frac{1}{12n+1} < \alpha_n < \frac{1}{12n} \tag{30}$$

Asymptotic bounds on factorials and their functions:

$$n! = o(n^n) \tag{31}$$

Proof.

$$\begin{aligned}
& n! = o(n^n) && (32)\\
\Leftrightarrow & n! < cn^n \quad (\forall c > 0, \exists n_0 > 0, n > n_0) && (33)\\
\Leftarrow & n! \leq n^{n-1} < cn^n && (34)\\
\Leftrightarrow & n^{n-1} < cn^n && (35)\\
\Leftrightarrow & n > \frac{1}{c} \quad \left(n_0 = \left\lceil \frac{1}{c} \right\rceil\right) && (36)
\end{aligned}$$

$\square$

$$n! = \omega(2^n) \tag{37}$$

Proof.

$$\begin{aligned}
& n! = \omega(2^n) && (38)\\
\Leftrightarrow & n! > c2^n \quad (\forall c > 0, \exists n_0 > 0, n > n_0) && (39)\\
\Leftrightarrow & \frac{n(n-1)\cdots 2}{2\cdot 2\cdots 2} > 2c \quad (n \geq 2) && (40)\\
\Leftarrow & \frac{n(n-1)\cdots 2}{2\cdot 2\cdots 2} > \frac{n}{2} > 2c && (41)\\
\Leftarrow & \frac{n}{2} > 2c && (42)\\
\Leftrightarrow & n > 4c \quad (n_0 = \max\{2, \lceil 4c \rceil\}) && (43)
\end{aligned}$$

□

$$\lg(n!) = \Theta(n \lg n) \tag{44}$$

9 The iterated logarithm function

9.1 Functional iteration

$$f^{(i)}(n) = \begin{cases} f(n) & \text{if } i = 0 \\ f(f^{(i-1)}(n)) & \text{if } i > 0 \end{cases} \qquad (i \text{ is nonnegative integer.})$$

Example 17.

$$f(n) = 2n \implies f^{(i)}(n) = 2^i n$$

9.2 Log-star function

$\lg^{(i)} n$ is defined only if $\lg^{(i-1)} n > 0$, different from $\lg^{i} n$.

The iterated logarithm or log-star function is defined as

$$\lg^* n = \min\{i \geq 0 : \lg^{(i)} n \leq 1\}$$

Another definition:

$$\lg^* n = \begin{cases} 0 & \text{if } n \leq 1 \\ 1 + \lg^*(\lg n) & \text{if } n > 1 \end{cases} \tag{45}$$

Example 18.

$$\begin{aligned} \lg^* 2 &= 1 \\ \lg^* 3 &= 2 \\ \lg^* 4 &= 2 \\ \lg^* 10 &= 3 \\ \lg^* 16 &= 3 \\ \lg^* 1000 &= 4 \\ \lg^* 65536 &= 4 \\ \lg^* 2^{65536} &= 5 \end{aligned}$$

It is rare to have an input size n such that $\lg^* n > 5$.

Some references on practicality of a large number:

$\boxed{\text{The age of the Universe is 13 billion years or } 4 \times 10^{17} \text{ seconds} \ll 2^{65536}.}$

$\boxed{\text{The number of atoms in the universe is } 10^{80} \ll 2^{65536}.}$

Use of log star function. From Wikipedia: The iterated logarithm appears in the time and space complexity bounds of some algorithms such as:

- Finding the Delaunay triangulation of a set of points knowing the Euclidean minimum spanning tree: randomized $O(n \log^* n)$ time
- Fürer's algorithm for integer multiplication: $O(n \log n 2^{O(\lg^* n)})$
- Finding an approximate maximum (element at least as large as the median): $\lg^* n - 4$ to $\lg^* n + 2$ parallel operations
- Richard Cole and Uzi Vishkin's distributed algorithm for 3-coloring an n-cycle: $O(\log^* n)$ synchronous communication rounds
- Performing weighted quick-union with path compression