Chapter 3
Growth of Functions

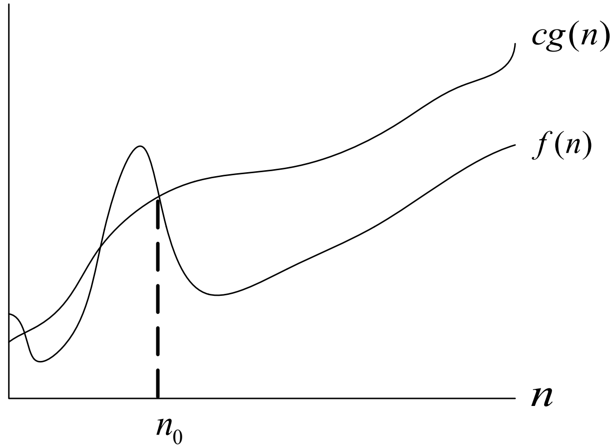
Chi-Yeh Chen
陳奇業
成功大學資訊工程學系

藏行顯光
成就共好
Achieve Securely
Prosper Mutually

國立成功大學 九十週年
90th Anniversary of NCKU

Asymptotic notation

$$O(g(n)) = \{ f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ s.t. } 0 \le f(n) \le cg(n) \text{ for all } n \ge n_0 \}$$



$g(n)$ is an asymptotic upper bound for $f(n)$

If $f(n) \in O(g(n))$, we write $f(n) = O(g(n))$
(will precisely explain this soon)

藏行顯光 成就共好
國立成功大學 九十週年
90th Anniversary of NCKU
Achieve Securely · Prosper Mutually

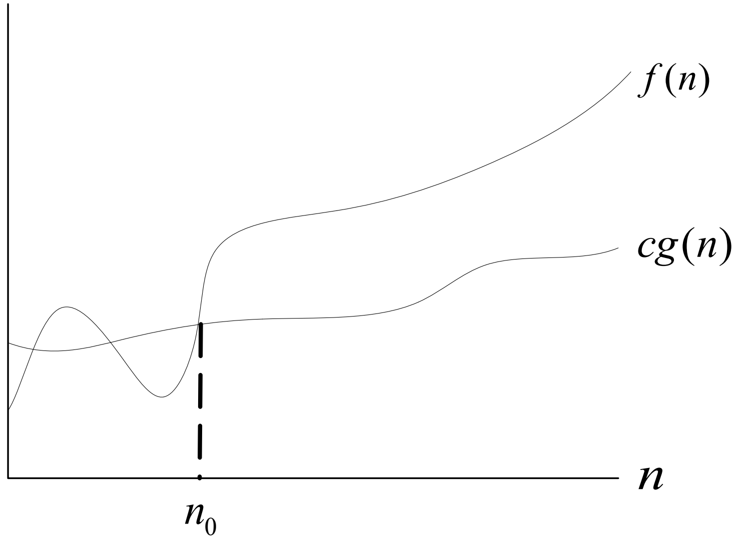
Asymptotic notation

- **O-notation**
- **Example:** $2n^2 = O(n^3)$, with $c = 1$ and $n_0 = 2$
- Examples of the functions in: $O(n^2)$

n^2	n
$n^2 + n$	$n / 1000$
$n^2 + 1000n$	$n^{1.99999}$
$1000n^2 + 1000n$	$n^2 / \lg\lg\lg n$

Asymptotic notation

$$\Omega\big(g(n)\big) = \{\, f(n) : \text{there exist positive constants } c \text{ and } n_0 \\ \text{s.t. } 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0 \}$$



$g(n)$ is an asymptotic upper bound for $f(n)$

Asymptotic notation

- **Ω-notation**
- **Example:** $\sqrt{n} = \Omega(\lg n)$, with $c = 1$ and $n_0 = 16$
- Examples of the functions in: $\Omega(n^2)$

n^2

$n^2 + n$

$n^2 - n$

$n^2 + 1000n$

$n^2 - 1000n$

n^3

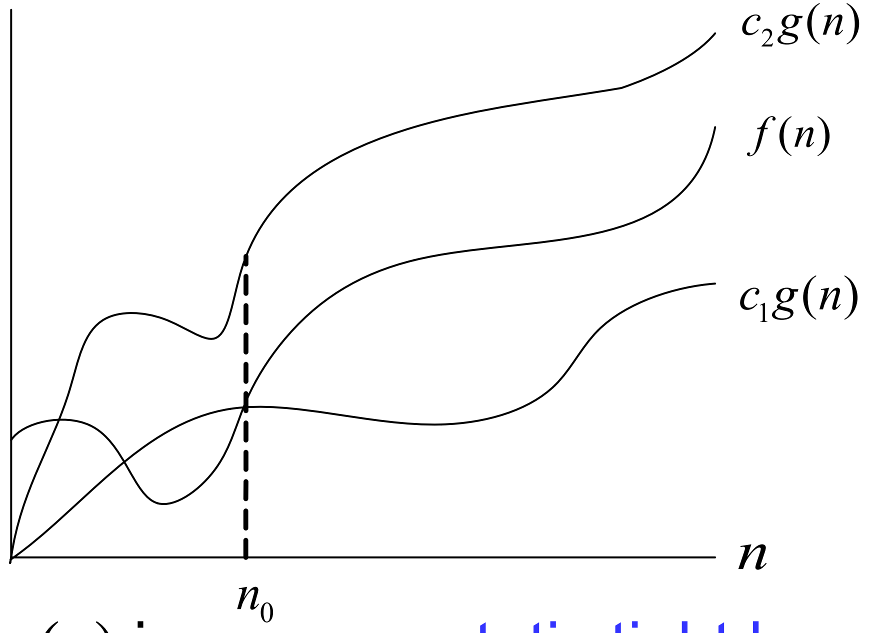
$n^{2.00001}$

$n^2 \lg\lg\lg n$

2^{2^n}

Asymptotic notation

$$\Theta\big(g(n)\big) = \{ f(n)\text{: there exist positive constants } c_1, c_2 \text{ and } n_0 \text{ s.t. } 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \text{ for all } n \ge n_0 \}$$



$g(n)$ is an asymptotic tight bound for $f(n)$

Asymptotic notation

- **Θ-notation**
- **Example:** $\dfrac{n^2}{2} - 3n = \Theta(n^2)$, with $c_1 = \dfrac{1}{14}$, $c_2 = \dfrac{1}{2}$, and $n_0 = 7$
- **Theorem**
  $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $\Omega(g(n))$
- Leading constants and low-order terms don't matter.

藏行顯光 成就 共好
國立成功大學 九十週年
90th Anniversary of NCKU
Achieve Securely · Prosper Mutually

$T(n) = T(n-2) + n\log\frac{n}{2}$ (Assume n is even.)

(1) Proof O:

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

$$= T(n-4) + (n-2)\log\frac{n-2}{2} + n\log\frac{n}{2}$$

$$= c + 2\log\frac{2}{2} + 4\log\frac{4}{2} + \ldots + (n-2)\log\frac{n-2}{2} + n\log\frac{n}{2}$$

$$\leq c + n\log\frac{n}{2} + n\log\frac{n}{2} + \ldots + n\log\frac{n}{2} + n\log\frac{n}{2}$$

$$= c + \frac{n}{2} * n\log\frac{n}{2} = c + \frac{n^2}{2}\log\frac{n}{2}$$

$$Therefore, T(n) = O(n^2\log n)$$

藏行顯光 成就共好 Achieve Securely Prosper Mutually

$T(n) = T(n-2) + n\log\frac{n}{2}$ (Assume n is even.)

(1) Proof Ω:

$$
\begin{aligned}
T(n) &= T(n-2) + n\log\frac{n}{2} \\
&= T(n-4) + (n-2)\log\frac{n-2}{2} + n\log\frac{n}{2} \\
&= c + 2\log\frac{2}{2} + \ldots + \left(\frac{n}{2}-2\right)\log\frac{(\frac{n}{2}-2)}{2} + \frac{n}{2}\log\frac{n}{4} + \ldots + n\log\frac{n}{2} \\
&\geq c + 0 + \ldots + 0 + \frac{n}{2}\log\frac{n}{4} + \ldots + n\log\frac{n}{2} \text{ (half of number)} \\
&= c + \frac{n}{4} * \frac{n}{2}\log\frac{n}{4} = c + \frac{n^2}{8}\log\frac{n}{4}
\end{aligned}
$$

$Therefore,\ T(n) = \Omega(n^2 \log n)$

By (1)(2), $T(n) = \theta(n^2 \log n)$

藏行顯光
成就共好
Achieve Securely
Prosper Mutually

Asymptotic notation in equations

- **When on the right-hand side:**

  $O(n^2)$ stands for some anonymous function in the set $O\left(n^2\right)$

  $2n^2 + 3n + 1 = 2n^2 + \Theta(n)$ means $2n^2 + 3n + 1 = 2n^2 + f(n)$ for $f(n) \in \Theta(n)$

  In particular, $f(n) = 3n + 1$

- We interpret # of anonymous functions as = # of times the asymptotic notation appears:

  $$\sum_{i=1}^{n} O(i)$$

  OK: 1 anonymous function

  $$O(1) + O(2) + \boxed{?} + O(n)$$

  not OK: n hidden constants → no clean interpretation

Asymptotic notation in equations

- **When on the left-hand side:**

No matter how the anonymous functions are chosen on the left-hand side, there is a way to choose the anonymous functions on the right-hand side to make the equation valid.

Interpret $2n^2 + \Theta(n) = \Theta(n^2)$ as meaning

for all functions $f(n) \in \Theta(n)$,

there exists a function $g(n) \in \Theta(n^2)$ such that $2n^2 + f(n) = g(n)$

Can chain together :

$$
\begin{aligned}
2n^2 + 3n + 1 &= 2n^2 + \Theta(n) \\
&= \Theta(n^2)
\end{aligned}
$$

Asymptotic notation in equations

- Interpretation

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

First equation : There exist $f(n) \in \Theta(n)$ such that

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

Second equation : For all $g(n) \in \Theta(n)$ (such as the $f(n)$ used to make the first equation hold), there exists $h(n) \in \Theta(n^2)$ such that $2n^2 + g(n) = h(n)$

藏行顯光 成就 共好 國立成功大學 九十週年 90th Anniversary of NCKU Achieve Securely · Prosper Mutually

Asymptotic notation in equations

o -notation

$$o(g(n)) = \{f(n): \text{for all constants } c > 0, \text{ there exists a constant } n_0 > 0 \text{ such that } 0 \le f(n) < cg(n) \text{ for all } n \ge n_0\}$$

$$\text{Another view, probably easier to use}: \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

$$n^{1.9999} = o(n^2)$$

$$n^2 / \lg n = o(n^2)$$

$$n^2 \ne o(n^2) \text{ (just like } 2 < 2)$$

$$n^2 / 1000 \ne o(n^2)$$

Asymptotic notation in equations

ω -notation $\omega(g(n)) = \{f(n)$: for all constants c > 0, there exists a constant $n_0 > 0$ such that $0 \leq cg(n) < f(n)$ for all $n \geq n_0\}$

$$\text{Another view, again, probably easier to use}: \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$$

$$n^{2.0001} = \omega(n^2)$$

$$n^2 \lg n = \omega(n^2)$$

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

藏行顯光 成就 共好 國立成功大學 九十週年 90th Anniversary of NCKU

Achieve Securely · Prosper Mutually

Comparisons of functions

- Relational properties:
  - **Transitivity:**

$f(n) = \Theta(g(n))$ and $g(n) = \Theta\big(h(n)\big) \Rightarrow f(n) = \Theta(h(n))$

Same at $O, \Omega, o,$ and ω.

  - **Reflexivity:**

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

Same at O and Ω.

藏行顯光 成就 共好 國立成功大學 九十週年 90th Anniversary of NCKU Achieve Securely · Prosper Mutually

Comparisons of functions

- Relational properties:
  - **Symmetry:**

$$f(n) = \Theta\big(g(n)\big) \text{ if and only if } g(n) = \Theta(f(n))$$

  - **Transpose symmetry:**

$$f(n) = O\big(g(n)\big) \text{ if and only if } g(n) = \Omega(f(n))$$

$$f(n) = o\big(g(n)\big) \text{ if and only if } g(n) = \omega(f(n))$$

- Comparisons:

  $f(n)$ is asymptotically smaller than $g(n)$ if $f(n) = o(g(n))$

  $f(n)$ is asymptotically larger than $g(n)$ if $f(n) = \omega(g(n))$

  No trichotony. Although intuitively, we can liken O to ≤, Ω to ≥, etc., unlike real numbers, where a < b, a = b, or a > b we might not be able to compare functions.

  Example : $n^{1+\sin n}$ and n, since $1+\sin n$ oscillates between 0 and 2.

藏行顯光
成就共好
Achieve Securely
Prosper Mutually

Standard notations and common functions

- Monotonicity

$f(n)$ is monotonically increasing if $m \leq n \Rightarrow f(m) \leq f(n)$

$f(n)$ is monotonically decreasing if $m \leq n \Rightarrow f(m) \geq f(n)$

$f(n)$ is strictly increasing if $m < n \Rightarrow f(m) < f(n)$

$f(n)$ is strictly decreasing if $m < n \Rightarrow f(m) > f(n)$

- Exponentials

Userful identities :

$$a^{-1} = 1/a,$$

$$\left(a^m\right)^n = a^{mn}$$

$$a^m a^n = a^{m+n}$$

Can relate rates of growth of polynomials and expoonentials : for all real constants a and b such that $a > 1$,

$$\lim_{n\to\infty} \frac{n^b}{a^n} = 0,$$

which implies that $n^b = o\left(a^n\right)$

A suprosongly useful inequality : for all real x,

$$e^x \geq 1 + x.$$

As x gets colsers to 0, e^x gets colser to $1 + x$.

藏行顯光
成就共好
Achieve Securely
Prosper Mutually

- # Logarithms(1)

Notations :

$\lg n = \log_2 n$ (binary logarithm),

$\ln n = \log_e n$ (natural logarithm),

$\lg^k n = (\lg n)^k$ (exponentiation),

$\lg\lg n = \lg(\lg n)$ (composition),

Logarithm functions apply only to the next term in the formula, so the $\lg n + k$ means $(\lg n) + \mathrm{k}$, and *not* $\lg(n + k)$

In the expression $\log_b a$:

– If we hole b constant, then the expression is strictly increasing as a increases.

– If we hold a constant, then the expression is strictly decreasing as b increases.

- Logarithms(2)

Usegful identities for all real $a > 0, b > 0, c > 0$, and n, and where logarothm bases are not 1 :

$$a = b^{\log_b a},$$

$$\log_c(ab) = \log_c a + \log_c b,$$

$$\log_b a^n = n \log_b a,$$

$$\log_b a = \frac{\log_c a}{\log_c b},$$

$$\log_b(1/a) = -\log_b a,$$

$$\log_b a = \frac{1}{\log_a b},$$

$$a^{\log_b c} = c^{\log_b a}.$$

藏行顯光 成就共好 Achieve Securely Prosper Mutually

- Logarithms(3)

Changing the base of a logarithm from one constant to another only changes the value by a constant factor, so we usually don't worry about logarithm bases in asymptotic notation. Covention is to use lg within asymptotic notation, unless the base actually matters.

Just as polynomials grow more slowly than exponentials, logarithms grow more slowly than polynomials.

In $\lim_{n\to\infty} \frac{n^b}{a^n} = 0$, substitute $\lg n$ for n and 2^a for a:

$$\lim_{n\to\infty} \frac{\lg^b n}{(2^a)^{\lg n}} = \lim_{n\to\infty} \frac{\lg^b n}{n^a} = 0,$$

implying that $\lg^{\mathrm{b}} n = o(n^a)$.

- Factorials

$n! = 1.2.3\ldots n.$ Special case: $0! = 1$

Can use Stirling's approximation,

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

to derive that $\lg(n!) = \Theta(n \lg n)$

藏行顯光
成就共好
Achieve Securely
Prosper Mutually

- Functional iteration
  - $f^{(i)}(n)$: $f(n)$ iteratively applied i times to an initial value of n.

$$f^{(i)}(n) = \begin{cases} n & \text{if } i = 0 \\ f\left(f^{(i-1)}(n)\right) & \text{if } i > 0 \end{cases}$$

ex. If $f(n) = 2n$, then $f^{(i)}(n) = 2^i n$.

藏行顯光 成就共好 Achieve Securely Prosper Mutually

- The iterated logarithm function
  - $\lg^* n = \min\{i \geq 0 : \lg^{(i)} n \leq 1\}$

- ex. $\lg^* 2 = 1$,

  $\lg^* 4 = 2$,

  $\lg^* 16 = 3$,

  $\lg^* 65536 = 4$,

  $\lg^* (2^{65536}) = 5$.

- Fibonacci numbers

$$F_0 = 0,$$

$$F_1 = 1,$$

$$F_i = F_{i-1} + F_{i-2} \text{ for } i \geq 2.$$

golden ratio

$$\phi = \frac{1+\sqrt{5}}{2} = 1.61803\ldots$$

$$\hat{\phi} = \frac{1-\sqrt{5}}{2} = -.61803\ldots$$

$$\Rightarrow F_i = \frac{\phi^i - \hat{\phi}^i}{\sqrt{5}}$$

Function	Name	Value					
	constant						
	logarithm	0	1	2	3	4	5
	linear	1	2	4	8	16	32
		0	2	8	24	64	160
	square	1	4	16	64	256	1,024
	cube	1	8	64	512	4,096	32,768
	exponential	2	4	16	256	65,536	4,294,967,296
	factorial						

Table 2.1 The running times (rounded up) of different algorithms on inputs of increasing size, for a processor performing a million high-level instructions per second. In cases where the running time exceeds 10^{25} years, we simply record the algorithm as taking a very long time.

	n	$n \log_2 n$	n^2	n^3	1.5^n	2^n	$n!$
$n = 10$	< 1 sec	< 1 sec	< 1 sec	< 1 sec	< 1 sec	< 1 sec	4 sec
$n = 30$	< 1 sec	< 1 sec	< 1 sec	< 1 sec	< 1 sec	18 min	10^{25} years
$n = 50$	< 1 sec	< 1 sec	< 1 sec	< 1 sec	11 min	36 years	very long
$n = 100$	< 1 sec	< 1 sec	< 1 sec	1 sec	12,892 years	10^{17} years	very long
$n = 1{,}000$	< 1 sec	< 1 sec	1 sec	18 min	very long	very long	very long
$n = 10{,}000$	< 1 sec	< 1 sec	2 min	12 days	very long	very long	very long
$n = 100{,}000$	< 1 sec	2 sec	3 hours	32 years	very long	very long	very long
$n = 1{,}000{,}000$	1 sec	20 sec	12 days	31,710 years	very long	very long	very long

Ref: Algorithm Design

藏行顯光
成就共好

Achieve Securely
Prosper Mutually

國立成功大學 九十週年
90th Anniversary of NCKU

1931
國立成功大學
National Cheng Kung University