CH1、Time Complexity

時間複雜度

考題重點(目錄)

Asymptotic Notation

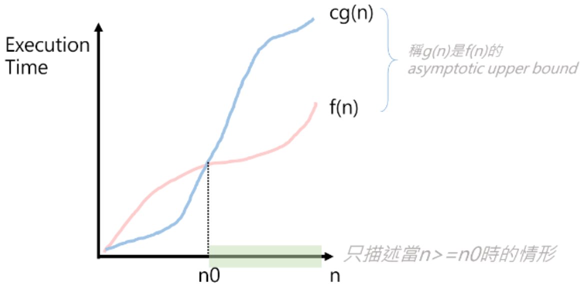
一、目的

當 input size 變大時，執行時間以何種趨勢成長

二、Asymptotic Notation

1. f(n) = O(g(n))：f(n)的 order <= g(n)的 order

   存在 c, n0 > 0，使得當 n>=n0*(當 input size 夠大時)*時，f(n) <= cg(n)



2. f(n)=Omega(g(n))：f(n)的 order >= g(n)的 order、或稱：g(n)為 f(n)的 asymptotic lower bound

   存當 c, n0 > 0，使得當 n>=n0*(當 input size 夠大時)*時，f(n) >= cg(n)

3. f(n)=Theta(g(n))：f(n)的 order = g(n)的 order、或稱 g(n)為 f(n)的 asymptotic tight bound

   存在 c1, c2, n0 > 0，使得當 n>=n0*(當 input size 夠大時)*時，c1g(n) <= f(n) <= c2g(n)*(前一不等式表 Omega、後一不等式表 O)*

4. $f(n)=o(g(n))$：$f(n)$的 order < $g(n)$的 order

   (def：可用 lim 法)

   例(94 成大)： True/False

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

   1. False
   2. True

   比較：$n=O(2n)$ => True；$n=O(n^2)$ => True

5. $f(n)=w(g(n))$：$f(n)$的 order > $g(n)$的 order

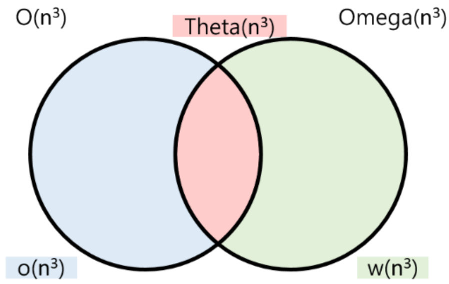
   (def：可用 lim 法)

   例：True/False

   1. $2n = w(n)$
   2. $n^2 = w(n)$

   1. False
   2. True

三、若固定 $g(n)$，則可將所有函數做以下分類：例：$g(n)=n^3$



例(99 政大)：寫出 2 個在 $O(n^3)$中，但不在 $o(n^3)$的函數

n^3、$2n^3$

例(100 交大)：

NCTU=Theta(n)

CS=Omega(n)

1. NCTU 總是比 CS 快
2. 當 n>=100000..0 時，NCTU 比 CS 快
3. 兩者執行時間相同
4. 稱 CS 的時間複雜度為 Theat(n)
5. 以上皆非

1. False
2. False
3. False
4. False
5. True

四、特性

1. 若 $f(n) = O(g(n))$，則 $f(n) + g(n) = O(g(n))$

   (Note：函數相加後的 order，決定於大者 => 多項式函數的 order 為最高次項)

例(98 交大)：

$p(n)=\text{Sum } a_i n^i$ 是 d 次多項式，填 True/False 在以下表格：

		O	o	Omega	w	Theta
p(n)	n^k, k>d					
p(n)	n^k, k<d					
p(n)	n^k, k=d					

p(n)的 order 為 n^d

		O	o	Omega	w	Theta
p(n)	n^k, k>d	O	O			
p(n)	n^k, k<d			O	O	
p(n)	n^k, k=d	O		O		O

例(98 交大)：寫出最適答案：$O(n^2)+\Theta(n^2)$

此為一常見的誤用

$O(n^2) + \text{Theta}(n^2) = \text{Theta}(n^2)$

自 $O(n^2)$取一函數 f(n)；自 $\text{Theta}(n^2)$取一函數 g(n)；f(n)+g(n)的 order 為何？

n(小) + n2(大) = $\text{Theta}(n^2)$

例(100 中央)：prove or disprove

2. f(n)+g(n) = Theta(max{f(n), g(n)})

True

用定義證：存在 c1=1 c2=1 n0=10 使得當 n>=10 時，滿足以下

c1 [f(n) + g(n)] <= Theta(max{f(n), g(n)}) <= c2 [f(n) + g(n)]

2. f(n)=O(g(n))且 f(n)=Omega(g(n)) ⇔ f(n)=Theta(g(n)) *//是一種常見求 tight bound 的方式*

例(95 台大)：Sum i^5 = Theta(n^a), a=?

1. *證 Sum $i^5 = O(n^6)$ by 1,2*
2. *證 Sum i^5 = Omega(n^6)*
3. *=> Sum i^5 = Theta(n^6)*

比較

一、定義法

適用時機：型簡單、分多

例(96 成大)：True/False(10%)

$n^2 + n \lg n + n/2 = O(n^8)$

True

存在 c=10 , n0= 100, 使得當 n>= 時，$n^2 + n \lg n + n/2 <= 10\, O(n^8)$

例(91 交大)：證明以下為錯誤

$n^2 / \log n = \Theta(n^2)$

定義法證明之想法：以下兩條件，破解其一即得證

$n^2 / \log n = O(n^2)$ *正確*

$n^2 / \log n = \Omega(n^2)$ *不正確，故：*

設 $n^2 / \log n = \Omega(n^2)$*成立，則存在 c, n0 >0，使得當 n>=n0 時，*$n^2 / \log n >= cn^2$

⇨ $1/\log n >= c$ *(不可能成立，因為 n 愈大，1/log n 愈接近 0)*

因不存在此 c，故等式錯誤

例(100 中央)：prove or disprove

1. f(n)=Theta(g(n))，則 h(f(n))=O(h(g(n)))，其中 h()為遞增函數

False

$f(n)=2n, g(n)=n => f(n)=O(g(n))$

$h(n)=2^n$

$h(f(n)) = 2^n != 2^{2n}=O(h(g(n)))$

二、lim 法

適用時機：型複雜，尚知如何微分，證 o, w 時可當 def

1. lim f(n)/g(n) = 0 ⇔ f(*n)=o(g(n))
2. lim f(n)/g(n) = 無限大 ⇔ f(*n)=w(g(n))
3. lim f(n)/g(n) = L ⇔ f(*n)=Theta(g(n))

(Note：通常搭配羅必達使用)

$\lim f(n)/g(n) = \lim f'(n)/g'(n)$

例：$f(n) = \log_3 4n$，問 f(n) = 何種(g(n))？

$\lim \log_3 4n/\log_4 3n = \lim (1/\lg 3 * 1/n)/(1/\lg 4 * 1/n) = \ln 4/\ln 3 > 0$

⇨ $f(n) = \Theta(g(n))$

(Note：log n 的底只要是常數，其 order 均相同)

Ex：$\lg n, \ln n, \log n, \log_{100} n$

例(96 成大)：True/False
$n^b = o(a^n)$, 其中 a>1, b 為任意實數

True
$\lim nb/an = \lim bn^{b-1}/a^n(\ln a) = \lim b(b-1)n^{b-2}/a^n(\ln a)^2 = 0$

例(98 台大電機)：True/False
對任何正數 a, b 而言：$n^b = o(a^n)$

False
反例：b=2, a=1

三、log 法

適用時機：分少的計算題(定理層次)

指數函數，n! (和 log(n!) = Theta(n lg n 搭配))

Log(f(n)) = o/w(log(g(n)))，則 f(n) = o/w(g(n))

Note：若為 Theta 則不能使用

例：$f(n)=1.1^{0.01n}$, $g(n)=n^{100}$

$\log(f(n)) = \log(1.1^{0.01n}) = 0.01n(\log 1.1) = cn$
$\log(g(n)) = \log(n^{100}) = 100(\log n) = c \log n$
因為 log(f(n)) = o(log(g(n)))
所以 f(n) = o(g(n))
⇨ *f(n)的 order 較大*

例(98 交大)：證(log n)!不是 polynomial-bounded

若 f(n)是 polynomial-bounded，則 $f(n) = O(n^k)$ for some k
$\log f(n) = O(\log n^k) = O(\log n)$
⇨ *若為 polynomial-bounded，則 order 小於 log n*
log((logn)!) = Theta(log n log(log n))
其 order 大於 log n，即 log((log n)!) = w(log n)
因此不是 polynomial-bounded

計算複雜度

一、如何求 tight bound(Theta)

[法一]：求出 closed form，最高次項即為其 tight bound

例：T(n) = 1+2+…+n = n(n-1)/2 = n2/2 + n/2 = Theta(n2)

[法二]：求 upper bound + 求 lower bound，兩者一樣即為 tight bound

例：$T(n) = \text{Sum } i^5 = \text{Theta}(n^a)$，求 a= ?

證 $T(n) = O(n^6)$、證 $T(n) = Omega(n^6)$ => $T(n) = Theta(n^6)$

二、Harmonic Series：Hn = 1 + 1/2 + 1/3 + … + 1/n = Theta(log n)

1. *證：Hn = O(log n) ⇔ 證：存在 c, n0>0，使得當 n>=n0 時，Hn<=c lg n*
   Hn -1 = 1/2 + 1/3 + … + 1/n <= 積分 1/x dx = ln x|(n 1) = ln n – ln 1 = ln n
   => Hn <= (ln n)+1 <= 2(ln n) (當 c=2, n>=3 成立) => Hn = O(log n)
2. *證：Hn = Omega(log n) ⇔ 證：存在 c, n0>0，使得當 n>=n0 時，Hn>=c lg n*
   Hn = 1 + 1/2 + … + 1/n >= 積分 1/x|(n 1) = ln n => Hn = Omega(ln n)

Note：$T(n) = 1^a + 1/2^a + 1/3^a + \ldots + 1/n^a = Theta(1)$, if a = 2, 3, …　　　//收練

三、log(n!) = Theta(n lg n)

1. *證：log(n!) = O(n lg n)*
   *log(n!) = log(1*2*…*n) = log1 + log2 + … +log n <= log n + log n + … + log n = n log n =>*
   log(n!) = O(n lg n)
2. *證：log(n!) = Omega(n lg n)*
   *log(n!) = log(1*2*…*n) = log1 + log2 + … +log n >= log n/2 + log (n/2+1) + … + log n >= log n/2 + log n/2 + … + log n/2 = n/2 log n/2 => log(n!) = Omega(n lg n)*

例(96 台大)：$T(n)=Sum\ k^2(\log k)^3 = Theta(n^d(\log n)^e)$，求 d, e=?

$1^2(\log 1)^3 + 2^2(\log 2)^3 + \ldots + n^2(\log n)^3 <= n * n^2(\log n)^3 => Theta(n^3(\log n)^3)$

四、$(\log_a n)^b = o(n^k)$

例：$(\log n)^{100} = o(n^{0.0001})$

例(96 輔大)：$(\log n)^3= O(n^{1/16})$

$\lim (\log n)^3/n^{1/16}= \lim (1/\ln 10)^3*(\ln n)^2*1/n \quad / \quad 1/16\ n^{-15/16} = \lim a\ (\ln n)2/n^{1/16}$
$= \lim a*b*\ldots / n^{1/16} = 0 => (\log n)^3 = o(n^{1/16}) = O(n^{1/16})$

例：問 n^{1+e} 和 $n^2/\log n$ 誰的 order 較大，其中 0<e<1

$n^{1+e} = n^{2-c}$, 0<c<1，因為 $\log n = o(n^c)$，因此 $n^{1+e} = o(n^2/\log n)$