

Divide and Conquer II

Michael Tsai

2012/09/20

矩陣相乘

- 問題:
- Input: A, B 都是 $n \times n$ 的 Square Matrix
- Output: C , $n \times n$ 的 square matrix, $C = A \cdot B$

矩陣相乘 - 基本法

n個乘法, n-1個加法,
產生了一個entry,
共有 n^2 個entries

$$\begin{array}{c} \leftarrow n \text{ 個} \\ \left[\begin{array}{ccc} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{array} \right] \left[\begin{array}{ccc} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{array} \right] \begin{array}{c} n \text{ 個} \\ \updownarrow \end{array} = \left[\begin{array}{ccc} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{array} \right] \end{array}$$

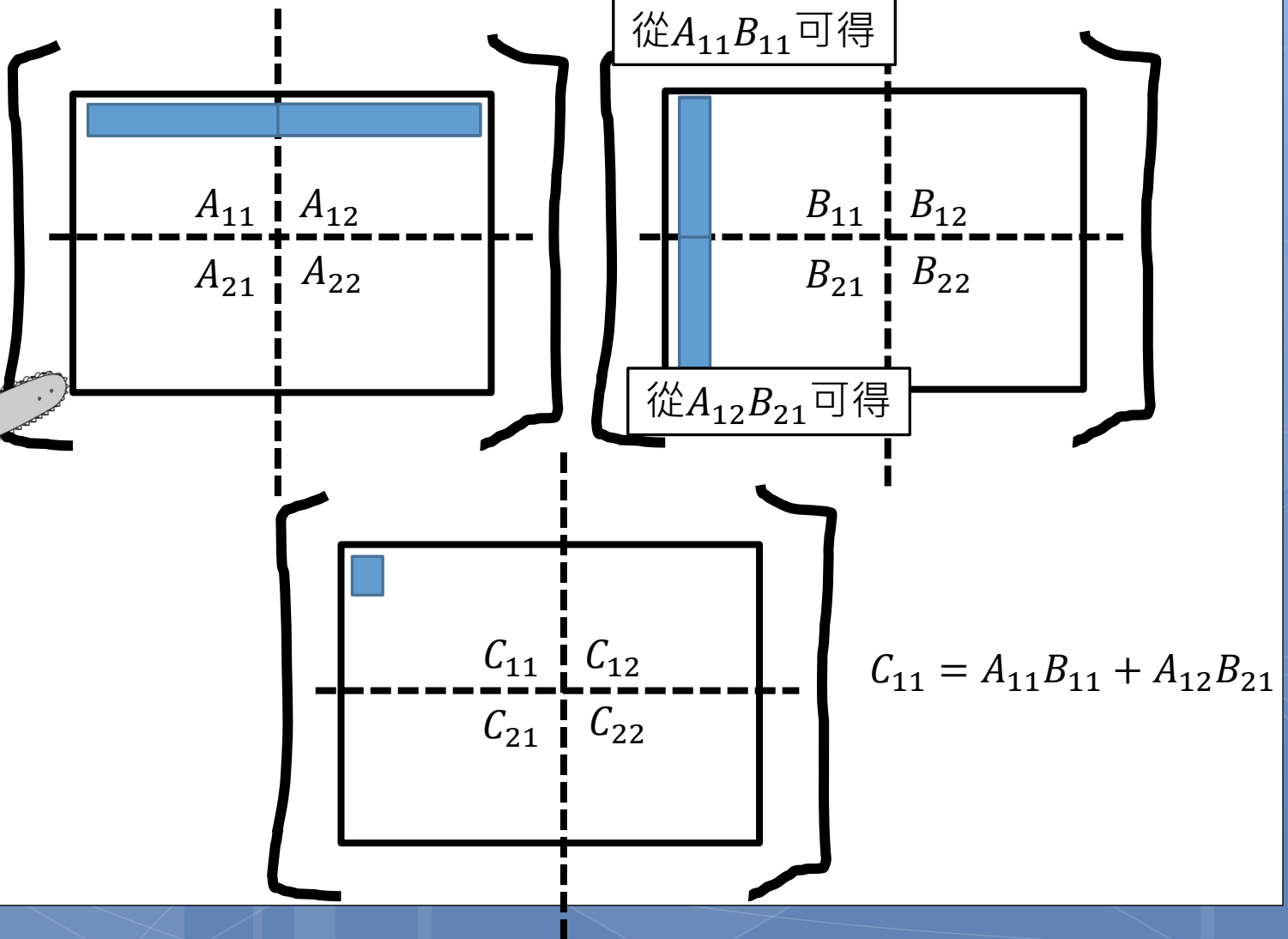
$$c_{ij} = \sum_{k=1}^n a_{ik} \cdot b_{kj}$$

Running time = ?

$\Theta(n^3)$

矩陣相乘 - D&C嘗試一

Recursive case:



矩陣相乘 – D&C嘗試一

Base case:

- $n=1$
- 則 $C = A \cdot B$ 直接算 (A, B, C 各自為一個數)

矩陣相乘 – D&C嘗試一

- 以下面的等式可求得C的四個小分塊的解:
- $C_{11} = A_{11} \cdot B_{11} + A_{12} \cdot B_{21}$
- $C_{12} = A_{11} \cdot B_{12} + A_{12} \cdot B_{22}$
- $C_{21} = A_{21} \cdot B_{11} + A_{22} \cdot B_{21}$
- $C_{22} = A_{21} \cdot B_{12} + A_{22} \cdot B_{22}$

矩陣相乘 – D&C嘗試一

Pseudo code:

Square-Matrix-Multiply-Recursive (A, B)

$n = A.rows$

let C be a new $n \times n$ matrix

if $n == 1$

$c_{11} = a_{11} \cdot b_{11}$ **Base case** $T(1) = \Theta(1)$

else partition the matrix into 4 $n/2 \times n/2$ matrices $\Theta(1)$

Recursive case

$C_{11} = \text{Square-Matrix-Multiply-Recursive}(A_{11}, B_{11})$

$+ \text{Square-Matrix-Multiply-Recursive}(A_{12}, B_{21})$

$C_{12} = \text{Square-Matrix-Multiply-Recursive}(A_{11}, B_{12})$

$+ \text{Square-Matrix-Multiply-Recursive}(A_{12}, B_{22})$

$C_{21} = \text{Square-Matrix-Multiply-Recursive}(A_{21}, B_{11})$

$+ \text{Square-Matrix-Multiply-Recursive}(A_{22}, B_{21})$

$\Theta(n^2)$ $C_{22} = \text{Square-Matrix-Multiply-Recursive}(A_{21}, B_{12})$

$+ \text{Square-Matrix-Multiply-Recursive}(A_{22}, B_{22})$

return C

$$T(n) = \Theta(1) + 8T\left(\frac{n}{2}\right) + \Theta(n^2) = 8T\left(\frac{n}{2}\right) + \Theta(n^2)$$

$$T(n) = \begin{cases} \Theta(1) & , \text{if } n = 1 \\ 8T\left(\frac{n}{2}\right) + \Theta(n^2) & , \text{if } n > 1 \end{cases}$$

$$T(n) = \Theta(n^3)$$

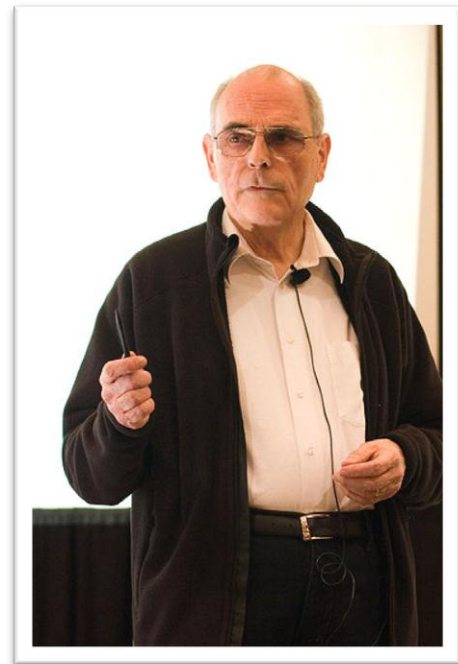
Combine

$8T\left(\frac{n}{2}\right)$



矩陣相乘 – Strassen's method

- 課本說: “Not at all obvious”
- 可達到 $\Theta(n^{\log 7}) = \Theta(n^{2.8074})$
- Overview:
 1. 將A, B及C都切成四塊 $n/2$ 大小的matrix
 2. 做出10個matrices, S_1, S_2, \dots, S_{10} . 這些matrix都是以步驟1中 $n/2$ 大小的matrices加減後得到的結果.
 3. 使用步驟1&2中得到的 $n/2$ 大小的matrices做乘法後得到 P_1, P_2, \dots, P_7 .
 4. 以 P_1, P_2, \dots, P_7 相加減後得到的結果產生 $C_{11}, C_{12}, C_{21}, C_{22}$ 四個matrix
- 細節先略過, 但我們可以計算running time了...



德國數學家
Volker Strassen
攝於2009年

矩陣相乘 – Strassen's method

- $\Theta(1)$ 1. 將A, B及C都切成四塊 $n/2$ 大小的matrix
- $\Theta(n^2)$ 2. 做出10個matrices, S_1, S_2, \dots, S_{10} . 這些matrix都是以步驟1中 $n/2$ 大小的matrices加減後得到的結果.
- $7T\left(\frac{n}{2}\right)$ 3. 使用步驟1&2中得到的 $n/2$ 大小的matrices做乘法後得到 P_1, P_2, \dots, P_7 .
- $\Theta(n^2)$ 4. 以 P_1, P_2, \dots, P_7 相加減後得到的結果產生 $C_{11}, C_{12}, C_{21}, C_{22}$ 四個matrix

$$T(n) = \Theta(1) + \Theta(n^2) + 7T\left(\frac{n}{2}\right) + \Theta(n^2) = 7T\left(\frac{n}{2}\right) + \Theta(n^2)$$



$$T(n) = \Theta(n^{\log 7})$$

矩陣相乘 – Strassen's method

- $S_1 = B_{12} - B_{22}$
- $S_2 = A_{11} + A_{12}$
- $S_3 = A_{21} + A_{22}$
- $S_4 = B_{21} - B_{11}$
- $S_5 = A_{11} + A_{22}$
- $S_6 = B_{11} + B_{22}$
- $S_7 = A_{12} - A_{22}$
- $S_8 = B_{21} + B_{22}$
- $S_9 = A_{11} - A_{21}$
- $S_{10} = B_{11} + B_{12}$
- $P_1 = A_{11} \cdot S_1$
- $P_2 = S_2 \cdot B_{22}$
- $P_3 = S_3 \cdot B_{11}$
- $P_4 = A_{22} \cdot S_4$
- $P_5 = S_5 \cdot S_6$
- $P_6 = S_7 \cdot S_8$
- $P_7 = S_9 \cdot S_{10}$

Not at all obvious

矩陣相乘 – Strassen's method

- $P_5 = A_{11}B_{11} + A_{11}B_{22} + A_{22}B_{11} + A_{22}B_{22}$
- $P_4 = \quad \quad \quad -A_{22}B_{11} \quad \quad \quad +A_{22}B_{21}$
- $-P_2 = \quad \quad \quad -A_{11}B_{22} \quad \quad \quad -A_{12}B_{22}$
- $P_6 = A_{12}B_{21} \quad \quad \quad -A_{22}B_{22} - A_{22}B_{21} + A_{12}B_{22}$
- $C_{11} = A_{11}B_{11} + A_{12}B_{21} = P_5 + P_4 - P_2 + P_6$
- C_{12}, C_{21}, C_{22} 用類似的方法
- <Reading assignment> Textbook 4.2
- <Homework for yourself> Exercise 4.2-1

Not at all obvious

矩陣相乘 – Strassen's method

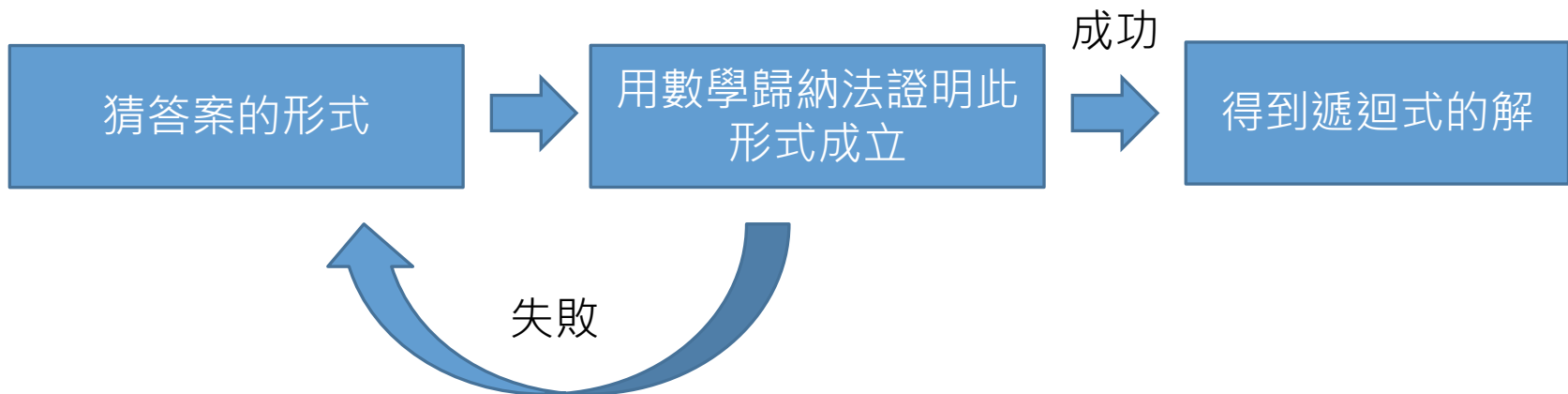
- 實用嗎? 好用嗎? 讓我們來挑毛病:
 1. $\Theta(n^{lg7})$ 的constant比 $\Theta(n^3)$ 大
 2. 那些sub-matrices要花額外的空間
 3. 當是Sparse Matrix時, 特別為Sparse Matrix設計的方法比較快
 4. Strassen's method is not as numerically stable as 基本法.
- 1990年代的部分研究減輕了2&4的壞處
- 所以, Strassen's method有什麼用呢?
- Key: find the crossover point and combine the two algorithms.
- 目前所知, the most asymptotically efficient algorithm has a running time of $O(n^{2.376})$. (Coppersmith and Winograd)

接下來...

- 問題: 我們要怎麼解遞迴式?
- 取代法
- 遞迴樹法
- 大師定理法



取代法



取代法 - 例子



- 問題: $T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n$
- 猜測: $T(n) = O(n \log n)$
- 用歸納法證明 $T(n) \leq c n \log n$, for a constant $c > 0$
- 假設上面的bound在 $m < n$ 時成立
- 則 $m = \left\lfloor \frac{n}{2} \right\rfloor$ 時亦成立.
- 也就是說 $T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) \leq c \left\lfloor \frac{n}{2} \right\rfloor \log\left(\left\lfloor \frac{n}{2} \right\rfloor\right)$

取代法 - 例子

$$\begin{aligned} \bullet T(n) &= 2T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + n \\ &\leq 2\left(c\left\lfloor \frac{n}{2} \right\rfloor \log\left(\left\lfloor \frac{n}{2} \right\rfloor\right)\right) + n \\ &\leq cn \log\left(\frac{n}{2}\right) + n \\ &= cn \log n - cn \log 2 + n \\ &= cn \log n - cn + n \\ &\leq cn \log n \end{aligned}$$

as long as $c \geq 1$

取代法 - 例子

- 接著必須也證明邊界條件成立.
- 有時候需要多一點努力....
- 假設 $T(1) = 1$.
- 我們必須證明 $T(n) \leq cn \log n, n = 1$
- 天不從人願: $n = 1$ 時, $cn \log n = 0$
- $T(1) = 1 > 0 = cn \log n$
- 娃~

取代法 - 例子

- 事實上, 我們只須證明, 當 $n > n_0$ 時, $T(n) \leq cn \log n$ 即可. (把不聽話的 $n = 1$ 拔掉)
- 從原本的遞迴式, 我們可以得到 $T(2) = 4, T(3) = 5$.
- 接著設 $n_0 = 2$. 我們發現:
- $T(2) \leq c 2 \log 2$ & $T(3) \leq c 3 \log 3$
- (只要 $c \geq 2$) 至此可以使邊界條件成立.
- 喔耶.

取代法-怎麼猜?

○ 靠經驗.

○ 跟沒講一樣.

○ 一些小方法:

1. 根據以前看過類似的遞迴式來猜測

2. 使用等一下要介紹的遞迴樹

3. 證明比較鬆的upper bound或lower bound來慢慢接近tight bound



$$T(n) = 2T\left(\left\lfloor \frac{n}{2} \right\rfloor + 17\right) + n$$

取代法-小技巧1

- ◉ 題目: $T(n) = T\left(\left\lfloor \frac{n}{2} \right\rfloor\right) + T\left(\left\lceil \frac{n}{2} \right\rceil\right) + 1$
- ◉ 猜測: $T(n) = O(n)$
- ◉ 歸納法證明:
- ◉ $T(n) \leq c\left(\left\lfloor \frac{n}{2} \right\rfloor + \left\lceil \frac{n}{2} \right\rceil\right) + 1 = cn + 1 \not\leq cn$
- ◉ 爛掉了...

取代法-小技巧1

- 方法: 改使用 $T(n) \leq cn - d, d \geq 0$
- (減掉一個order較低的term)
- $$\begin{aligned} T(n) &\leq \left(c \left\lfloor \frac{n}{2} \right\rfloor - d \right) + \left(c \left\lfloor \frac{n}{2} \right\rfloor - d \right) + 1 \\ &= cn - 2d + 1 \\ &\leq cn - d \end{aligned}$$
- (as long as $d \geq 1$)
- (然後繼續選c, 使得boundary condition成立, 在此省略)

取代法-小技巧2

- 看起來挺嚇人的: $T(n) = 2T(\lfloor \sqrt{n} \rfloor) + \log n$
- 替換變數: $m = \log n$
- $T(2^m) = 2T\left(2^{\frac{m}{2}}\right) + m$ 暫時不管flooring
- 定義: $S(m) = T(2^m)$
- $S(m) = 2S\left(\frac{m}{2}\right) + m$
- $\rightarrow S(m) = O(m \log m)$
- $\rightarrow T(n) = T(2^m) = S(m) = O(m \log m) = O(\log n \log \log n)$

遞迴樹法

畫出遞迴樹



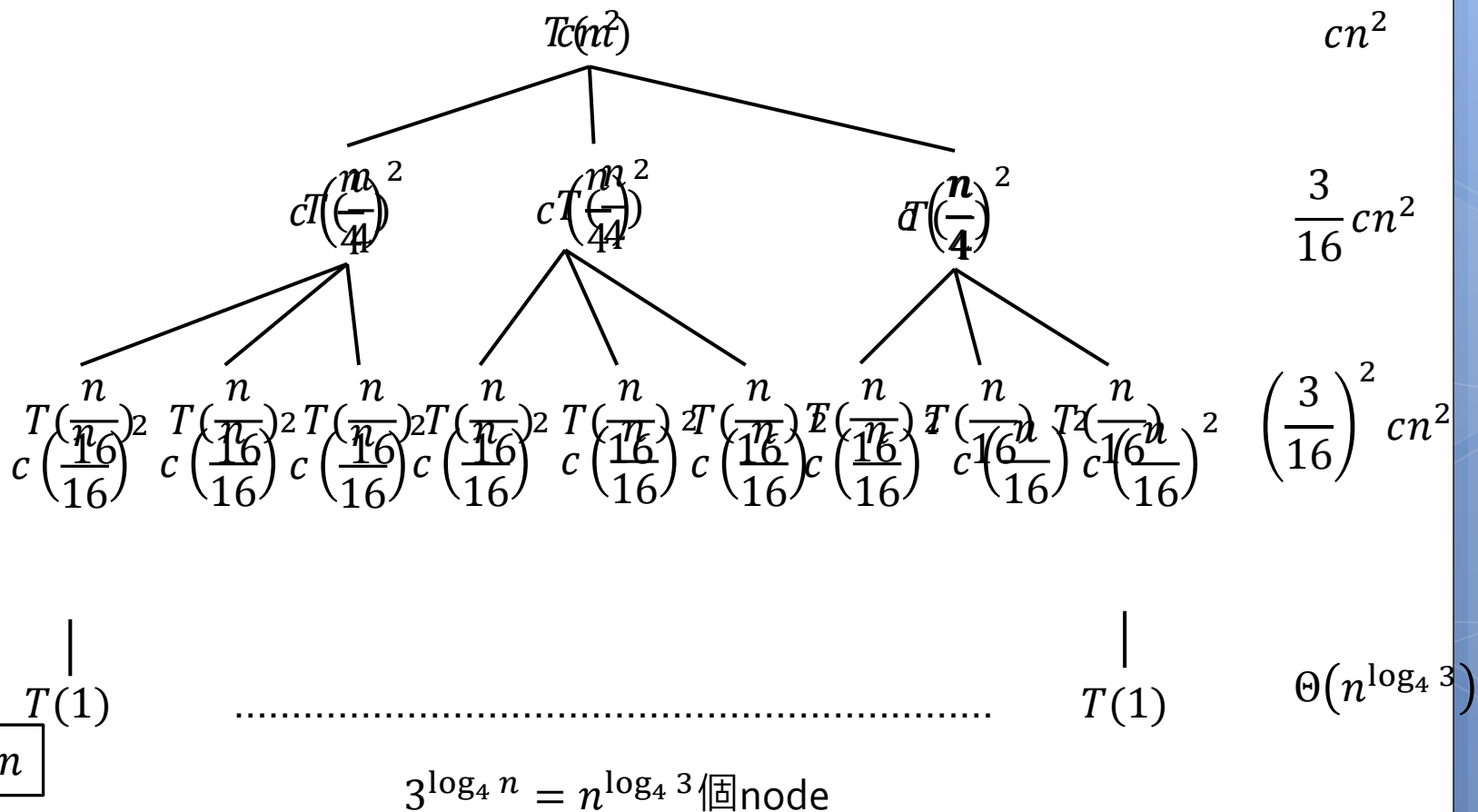
用比較不嚴謹的方法
加總得到解



用數學歸納法證明此
解成立

遞迴樹法-例子1

- 例子: $T(n) = 3T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + \Theta(n^2)$



遞迴樹法-例子1

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

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

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

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

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

$$= O(n^2)$$

遞迴樹法-例子1

- 用歸納法證明: $T(n) \leq dn^2$ for some $d > 0$.

- $T(n) \leq 3T\left(\left\lfloor \frac{n}{4} \right\rfloor\right) + cn^2$

$$\leq 3d \left(\left\lfloor \frac{n}{4} \right\rfloor\right)^2 + cn^2$$

$$\leq 3d \left(\frac{n}{4}\right)^2 + cn^2$$

$$= \frac{3}{16} dn^2 + cn^2$$

$$\leq dn^2$$

as long as $d \geq \left(\frac{16}{13}\right)c$

遞迴樹法-例子2

- 例子: $T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + O(n)$
- 請一位同學上來畫遞迴樹 (有點跛腳的遞迴樹)

遞迴樹法-例子2

- 歸納法證明: $T(n) \leq dn \log n$
- $T(n) \leq T\left(\frac{n}{3}\right) + T\left(\frac{2n}{3}\right) + cn$

$$\leq d\left(\frac{n}{3}\right) \log \frac{n}{3} + d\left(\frac{2n}{3}\right) \log \frac{2n}{3} + cn$$

$$= d\left(\frac{n}{3}\right) \log n + d\left(\frac{2n}{3}\right) \log n - d\left(\frac{n}{3}\right) \log 3 - d\left(\frac{2n}{3}\right) \log \left(\frac{3}{2}\right) + cn$$

$$= dn \log n - d\left(\left(\frac{n}{3}\right) \log 3 + \left(\frac{2n}{3}\right) \log 3 - \left(\frac{2n}{3}\right) \log 2\right) + cn$$

$$= dn \log n - dn \left(\log 3 - \frac{2}{3}\right) + cn$$

$$\leq dn \log n$$

as long as $d \geq \frac{c}{\log 3 - \frac{2}{3}}$

大師定理



Master Theorem:

Let $a \geq 1$ and $b \geq 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the nonnegative integers by the recurrence

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

where we interpret $\frac{n}{b}$ to mean either $\lfloor \frac{n}{b} \rfloor$ or $\lceil \frac{n}{b} \rceil$. Then $T(n)$ has the following asymptotic bounds:

1. if $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, 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 $\epsilon > 0$, and if $af\left(\frac{n}{b}\right) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n , then $T(n) = \Theta(f(n))$

大師定理

$n^{\log_b a}$ is larger

1. if $f(n) = O(n^{\log_b a - \epsilon})$ for some constant $\epsilon > 0$, then
 $T(n) = \Theta(n^{\log_b a})$
2. if $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$ The same order
3. if $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some constant $\epsilon > 0$, and if
 $af\left(\frac{n}{b}\right) \leq cf(n)$ for some constant $c < 1$ and all sufficiently
 large n , then $T(n) = \Theta(f(n))$ $f(n)$ is larger

Which one is polynomially larger/smaller?

$f(n)$

$n^{\log_b a}$

Note: not all possibilities for $f(n)$ can be covered by these 3 cases!

大師定理 - 例子1

- $T(n) = 9T\left(\frac{n}{3}\right) + n$
- $a=9, b=3, f(n)=n$
- $\log_b a = \log_3 9 = 2$
- satisfies case 1: $f(n) = O(n^{\log_b a - \epsilon}), \epsilon = 1$
- so $T(n) = \Theta(n^{\log_b a}) = \Theta(n^2)$

For your reference: $T(n) = aT\left(\frac{n}{b}\right) + f(n)$

$f(n)$

$n^{\log_b a}$

大師定理 - 例子2

- $T(n) = T\left(\frac{2n}{3}\right) + 1$
- $a=1, b=3/2, f(n)=1$
- $n^{\log_b a} = n^{\frac{\log_3 1}{2}} = n^0 = 1$
- satisfies case 2: $f(n) = \Theta(n^{\log_b a} \log n) = \Theta(\log n)$

For your reference: $T(n) = aT\left(\frac{n}{b}\right) + f(n)$

$f(n)$

$n^{\log_b a}$

大師定理 - 更多例子

上台解題時間...

- $T(n) = 3T\left(\frac{n}{4}\right) + n \log n$
- $T(n) = 2T\left(\frac{n}{2}\right) + n \log n$
- $T(n) = 2T\left(\frac{n}{2}\right) + \Theta(n)$
- $T(n) = 8T\left(\frac{n}{2}\right) + \Theta(n^2)$
- $T(n) = 7T\left(\frac{n}{2}\right) + \Theta(n^2)$



取數問題

- Selection Problem
- 問題:
- Input: n 個數字之集合
- Output: 取出此 n 個數字之中位數

- 中位數之定義: n 個數字中第 $k = \lceil \frac{n}{2} \rceil$ 小的數字

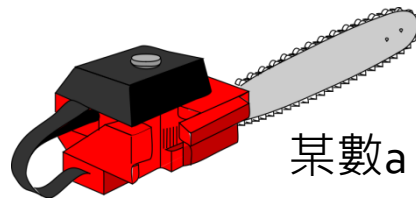
取數問題

- 菜瓜布解法:
- 先把 n 個數sort好
- 從最小的數過去算到第 k 小的數字即為答案
- running time=?
- $\Omega(n \log n)$
- Can we do a better job?



取數問題

n個數



某數a (pivot)

L_2

小於等於a的

L_1

大於a的

if $k > |L_2|$, 找 L_1 中第 $k - |L_2|$ 小的

if $k \leq |L_2|$, 找 L_2 中第 k 小的

取數問題

- 下一個問題: 怎麼選a?
- 選不好的話... $L_1 = n, L_2 = 0$

- 最好是可以平均分成兩分.
- 那就是選中位數.
- 噢, 我們不是就要找中位數嗎?

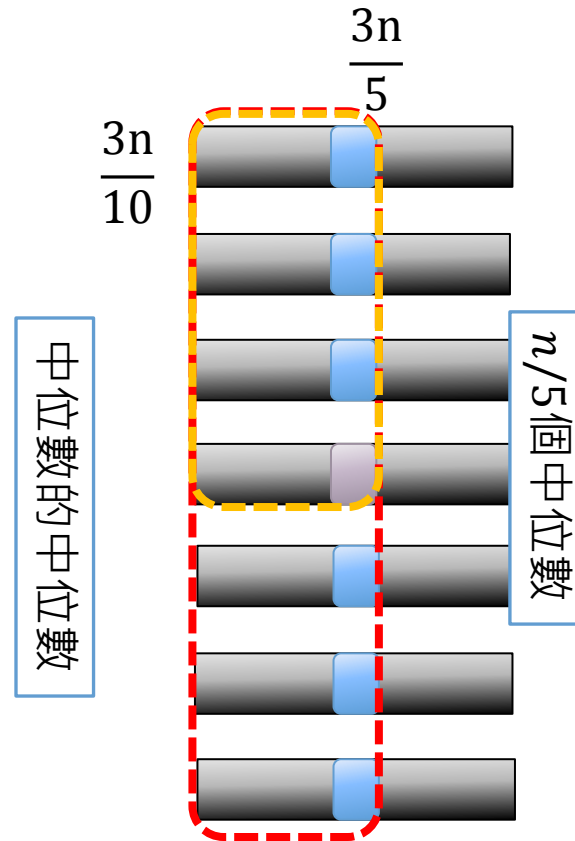
- 能不能花少一點時間, 找個“差不多”的中位數

取數問題

- 差不多的中位數:
 1. 把 n 個數分成很多大小為5個的sub list (大約共有 $n/5$ 個sub list)
 2. 這些sub list中各自找中位數
 3. 找出 $n/5$ 個中位數中的中位數
- 此為差不多的中位數

取數問題

- 有多差不多呢?
- 比“差不多中位數”小的至少有 $\frac{3n}{10}$
- $|L_2| \geq \frac{3n}{10}$
- $|L_1| \leq \frac{7n}{10}$



取數問題

Algorithm:

1. if $n \leq 5$ then 直接找出其中位數 $\Theta(1)$
2. else
3. 把數列拆成 $\frac{n}{5}$ 個大小為5的小數列 $\Theta(n)$
4. 每個小數列找出其中位數 $\Theta(n)$
5. 找出 $\frac{n}{5}$ 個中位數的中位數 m $T(\frac{n}{5})$
6. 用此中位數把原本的數列拆成兩部分: 比 m 大(L_1) $\Theta(n)$
及不比 m 大的(L_2)
7. if $k > |L_2|$, 找 L_1 中第 $k - |L_2|$ 小的
8. if $k \leq |L_2|$, 找 L_2 中第 k 小的

$$\text{Max: } T\left(\frac{7n}{10}\right)$$

$$\begin{aligned} T(n) &= \Theta(n) + \Theta(n) + T\left(\frac{n}{5}\right) + \Theta(n) + T\left(\frac{7n}{10}\right) \\ &= T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) + \Theta(n) \end{aligned}$$

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

Today's Reading Assignment

- Cormen ch 4.2 – 4.5