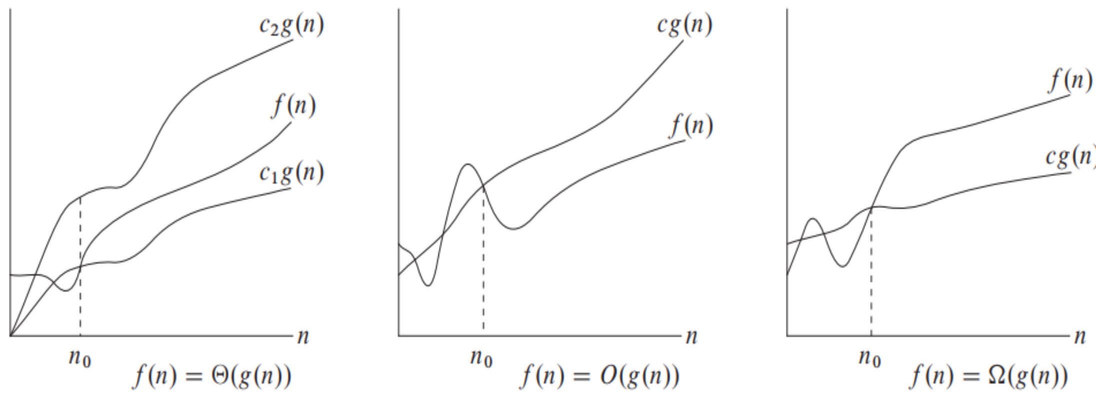
劉錫, Liu Xi, Algorithms, c3, 4

66



Θ bounds a function from above and below, 确界

$$\Theta(g(n)) = \{f(n) : \exists (c_1, c_2, n_0) \in \mathbb{R}^3_{>0}, 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n), \forall n \geq n_0\}$$

O give an upper bound on a function, 渐进上界

$$O(g(n)) = \{f(n) : \exists (c, n_0) \in \mathbb{R}^2_{>0}, 0 \leq f(n) \leq cg(n), \forall n \geq n_0\}$$

Ω give an lower bound on a function, 渐进下界

$$\Omega(g(n)) = \{f(n) : \exists (c, n_0) \in \mathbb{R}^2_{>0}, 0 \leq cg(n) \leq f(n), \forall n \geq n_0\}$$

71

o = not tight upper bound

$$o(g(n)) = \{f(n) : \forall c \in \mathbb{R}_{>0}, \exists n_0 \in \mathbb{R}_{>0}, 0 \leq f(n) < cg(n), \forall n \geq n_0\}$$

ω = not tight lower bound

$$\omega(g(n)) = \{f(n) : \forall c \in \mathbb{R}_{>0}, \exists n_0 \in \mathbb{R}_{>0}, 0 \leq cg(n) < f(n), \forall n \geq n_0\}$$

88
technicality 技术性 in recurrence
if we call MERGE-SORT on n elements when n is odd, we end up with subproblems of size $\lceil n/2 \rceil$ /*$left\_len = mid - left + 1$*/ and $\lfloor n/2 \rfloor$ /*$right\_len = right - mid$*/. both not equal to $n/2$, because $n/2$ is not an integer when n is odd

$$T(n) = \begin{cases} \Theta(1) & \text{if n = 1} \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) & \text{if n > 1} \end{cases}$$

104
substitution

$$\text{recurrence 递推: } T(n) = 2T(\lfloor n/2 \rfloor) + n$$

guess solution is $T(n) = O(n \lg n)$, substitution method requires us to prove

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

assume $T(m)$ is true for all positive $m < n$, in particular for $m = \lfloor n/2 \rfloor$, or equivalently $T(\lfloor n/2 \rfloor) \le c\lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor)$ IH
substitute IH into recurrence

$$\begin{aligned} T(n) &= 2(T(\lfloor n/2 \rfloor)) + n \\ &\le 2(c\lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor)) + n \\ &\le cn \lg(n/2) + n \qquad /* \lfloor n/2 \rfloor \le n/2; \quad 2\lfloor n/2 \rfloor \le n */ \\ &= cn(\lg n - \lg 2) + n \qquad /* \lg\left(\frac{2^a}{2^b}\right) = \lg(2^{a-b}) = a - b = \lg 2^a - \lg 2^b */ \\ & /* \lg(2^a 2^b) = \lg(2^{a+b}) = a + b = \lg 2^a + \lg 2^b */ \\ &= cn \lg n - cn \lg 2 + n \\ &= cn \lg n - cn + n \\ &\le cn \lg n \end{aligned}$$

problematic boundary $T(1)$

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

$$T(1) = 2T(0) + 1 = 1 \not\le c1 \lg 1 = 0$$

so use $T(2)$ and $T(3)$ as base cases in inductive proof

$$T(2) = 2T(\lfloor 2/2 \rfloor) + 2 = 2T(1) + 2 = 2(1) + 2 = 4$$

$$T(3) = 2T(\lfloor 3/2 \rfloor) + 3 = 2T(1) + 3 = 2(1) + 3 = 5$$

choose c so that $T(2) \le c\,2 \lg 2$ and $T(3) \le c\,3 \lg 3$

$$T(2) = 4 \le 2c \qquad 2 \le c$$

$$T(3) = 5 \le c\,3 \lg 3 \qquad 1.05154958929 \approx \frac{5}{3 \lg 3} \le c$$

so let $c \ge 2$

subtract a lower term

try to prove $T(n) \le cn$

$$\begin{aligned} T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 1 \\ &\le c\lceil n/2 \rceil + c\lfloor n/2 \rfloor + 1 \\ &= cn + 1 \end{aligned}$$

prove $T(n) \le cn - d$ instead

because $d \ge 0 \wedge T(n) \le cn - d \rightarrow T(n) \le cn$

assume $T(n) \le cn - d$ is true for all positive $m < n$, in particular for $m = \lceil n/2 \rceil$ and $m = \lfloor n/2 \rfloor$, or equivalently assume $T(\lceil n/2 \rceil) \le c\lceil n/2 \rceil - d$ and $T(\lfloor n/2 \rfloor) \le c\lfloor n/2 \rfloor - d$

$$\begin{aligned} T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + 1 \\ &\le (c\lceil n/2 \rceil - d) + (c\lfloor n/2 \rfloor - d) + 1 \\ &= cn - 2d + 1 \\ &\le cn - d \qquad if\ d \ge 1 \end{aligned}$$

change variables

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

rename $m = \lg n,\ \ 2^m = n$

$$T(2^m) = 2T(\lfloor 2^{m/2}\rfloor) + \lg n$$

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

$$S(m) = 2S(\lfloor m/2\rfloor) + m$$

which is very much like recurrence in p.104
so

$$S(m) = O(m \lg m)$$

$$T(n) = T(2^m) = S(m) = O(m \lg m) = O(\lg n \lg(\lg n))$$

108
4.3-2
show $T(n) = T(\lceil n/2\rceil) + 1$ is $O(\lg n)$:
need to show $T(n) \leq c \lg n$ for all $n \geq n_0$, where c and n_0 are positive constants
assume $T(n) \leq c \lg n$ is true for all positive $m < n$, in particular for $m = \lceil n/2\rceil$, or equivalently assume $T(\lceil n/2\rceil) \leq c \lg(\lceil n/2\rceil)$

$$\begin{aligned} T(n) &= T(\lceil n/2\rceil) + 1 \\ &\leq c \lg(\lceil n/2\rceil) + 1 \\ &< c \lg(n/2 + 1) + 1 \\ &= c \lg(n/2 + 2/2) + 1 \\ &= c \lg((n+2)/2) + 1 \\ &= c \lg(n+2) - c \lg 2 + 1 \\ &= c \lg(n+2) - c + 1 \qquad \textit{inconclusive} \end{aligned}$$

assume $T(n) \leq c\lg(n-2)$ is true for all positive $m < n$, in particular for $m = \lceil n/2 \rceil$, or equivalently assume $T(\lceil n/2 \rceil) \leq c\lg(\lceil n/2 \rceil - 2)$

$$\begin{aligned}
T(n) &= T(\lceil n/2 \rceil) + 1 \\
&\leq c\lg(\lceil n/2 \rceil - 2) + 1 \\
&< c\lg(n/2 + 1 - 2) + 1 \\
&= c\lg(n/2 - 1) + 1 \\
&= c\lg((n-2)/2) + 1 \\
&= c(\lg(n-2) - \lg 2) + 1 \\
&= c(\lg(n-2) - 1) + 1 \\
&= c\lg(n-2) - c + 1 \\
&\leq c\lg(n-2) \qquad if\ c \geq 1
\end{aligned}$$

4.3-3
$T(n) = 2T(\lfloor n/2 \rfloor) + n$ is $O(n\lg n)$ (p.104), show it is also $\Omega(n\lg n)$:

need to show $T(n) \geq cn\lg n$ for all $n \geq n_0$, where c and n_0 are positive constants

assume $T(n) \geq cn\lg n$ is true for all positive $m < n$, in particular for $m = \lfloor n/2 \rfloor$, or equivalently assume $T(\lfloor n/2 \rfloor) \geq c\lfloor n/2 \rfloor \lg \lfloor n/2 \rfloor$

$$\begin{aligned}
T(n) &= 2T(\lfloor n/2 \rfloor) + n \\
&\geq 2c\lfloor n/2 \rfloor \lg \lfloor n/2 \rfloor + n \\
&\geq 2c(n/2 - 1)\lg(n/2 - 1) + n \\
&= 2c((n-2)/2)\lg((n-2)/2) + n \\
&= c(n-2)(\lg(n-2) - \lg 2) + n \\
&= c(n-2)\lg(n-2) - c(n-2) + n \\
&= c(n-2)\lg(n-2) + n(1-c) + 2c \qquad \textit{inconclusive}
\end{aligned}$$

assume $T(n) \geq c(n+2)\lg(n+2)$ is true for all positive $m < n$, in particular for $m = \lfloor n/2 \rfloor$, or equivalently assume $T(\lfloor n/2 \rfloor) \geq c(\lfloor n/2 \rfloor + 2)\lg(\lfloor n/2 \rfloor + 2)$

$$\begin{aligned}
T(n) &= 2T(\lfloor n/2 \rfloor) + n \\
&\geq 2c(\lfloor n/2 \rfloor + 2)\lg(\lfloor n/2 \rfloor + 2) + n \\
&\geq 2c(n/2 - 1 + 2)\lg(n/2 - 1 + 2) + n \\
&= 2c(n/2 + 1)\lg(n/2 + 1) + n \\
&= 2c((n+2)/2)\lg((n+2)/2) + n \\
&= c(n+2)(\lg(n+2) - \lg 2) + n \\
&= c(n+2)\lg(n+2) - c(n+2) + n \\
&= c(n+2)\lg(n+2) + n(1-c) - 2c \\
&\geq c(n+2)\lg(n+2) \\
& if\ n(1-c) - 2c \geq 0, \quad i.e. \quad n \geq \frac{2c}{1-c}
\end{aligned}$$

$0 \leq c < 1$
if $c = 0,\ n \geq= 0 = n_0$
if $c = \frac{1}{2},\ n \geq \frac{2(1/2)}{1/2} = 2 = n_0$

108
4.3-5
show $\Theta(n \lg n)$ is the solution to merge sort:
recurrence is given by (p.88)

$$T(n) = \begin{cases} \Theta(1) & if\ n = 1 \\ T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + \Theta(n) & if\ n > 1 \end{cases}$$

need to prove O and Θ
prove $T(n) \leq c(n-2)\lg(n-2)$

assume $T(n) \geq c(n-2)\lg(n-2)$ is true for all positive $m < n$, in particular for $m = \lceil n/2 \rceil$ and $m = \lfloor n/2 \rfloor$, or equivalently assume $T(\lceil n/2 \rceil) \leq c(\lceil n/2 \rceil - 2)\lg(\lceil n/2 \rceil - 2)$ and $T(\lfloor n/2 \rfloor) \leq c(\lfloor n/2 \rfloor - 2)\lg(\lfloor n/2 \rfloor - 2)$
assume $\Theta(n) = kn$ for the recurrence stated above, where k is a positive

constant

$$
\begin{aligned}
T(n) &= T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + kn \\
&\le c(\lceil n/2 \rceil - 2)\lg(\lceil n/2 \rceil - 2) + c(\lfloor n/2 \rfloor - 2)\lg(\lfloor n/2 \rfloor - 2) + kn \\
&\le c(n/2 + 1 - 2)\lg(n/2 + 1 - 2) + c(n/2 + 1 - 2)\lg(n/2 + 1 - 2) + kn \\
&= 2c(n/2 + 1 - 2)\lg(n/2 + 1 - 2) + kn \\
&= 2c(n/2 - 1)\lg(n/2 - 1) + kn \\
&= 2c((n-2)/2)\lg((n-2)/2) + kn \\
&= c(n-2)\lg((n-2)/2) + kn \\
&= c(n-2)(\lg(n-2) - \lg 2) + kn \\
&= c(n-2)(\lg(n-2) - 1) + kn \\
&= c(n-2)\lg(n-2) - c(n-2) + kn \\
&= c(n-2)\lg(n-2) + n(k-c) + 2c \\
&= c(n-2)\lg(n-2) - (n(c-k) - 2c) \\
&\le c(n-2)\lg(n-2) \qquad if\ n(c-k) - 2c \ge 0
\end{aligned}
$$

$$n(c-k) - 2c \ge 0$$

$$n(c-k) \ge 2c$$

$$n \ge \frac{2c}{c-k}$$

$$\text{so, pick} \quad n \ge n_0 = \frac{2c}{c-k} \quad \text{and} \quad c > k$$

4.3-9
solve $T(n) = 3T(\sqrt{n}) + \lg n$:
let $2^m = n, \quad \log_2 n = m$

$$T(2^m) = 3T(\sqrt{2^m}) + \log_2(2^m)$$

$$T(2^m) = 3T(2^{m/2}) + m$$

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

$$S(m) = 3S(m/2) + m$$

if a recurrence is $T(n) = aT(n/b) + f(n)$, solution is $O(n^{\log_b a})$

prove $S(m) \le cm^{\log_2 3}$ for all $m \ge m_0$, where c and m_0 are positive constants

assume $S(m) \ge cm^{\log_2 3}$ is true for all positive $i < m$, in particular for $i = m/2$, or equivalently assume $S(m/2) \le c(m/2)^{\log_2 3}$

$$\begin{aligned}
S(m) &= 3S(m/2) + m \\
&\le 3c(m/2)^{\log_2 3} + m \\
&= 3c\left(\frac{m^{\lg 3}}{2^{\lg 3}}\right) + m \\
&= 3c\left(\frac{m^{\lg 3}}{3}\right) + m \\
&= cm^{\lg 3} + m \qquad \textit{inconclusive}
\end{aligned}$$

prove $S(m) \le cm^{\log_2 3} - bm$ for all $m \ge m_0$, where c and m_0 are positive constants

assume $S(m) \ge cm^{\log_2 3} - bm$ is true for all positive $i < m$, in particular for $i = m/2$, or equivalently assume $S(m/2) \le c(m/2)^{\log_2 3} - b(m/2)$

$$\begin{aligned}
S(m) &= 3S(m/2) + m \\
&\le 3(c(m/2)^{\log_2 3} - b(m/2)) + m \\
&= 3c\frac{m^{\lg 3}}{2^{\lg 3}} - 3bm/2 + m \\
&= 3c\frac{m^{\lg 3}}{3} - 3bm/2 + m \\
&= cm^{\lg 3} - 3bm/2 + m \\
&= cm^{\lg 3} - (2+1)bm/2 + m \\
&= cm^{\lg 3} - 2bm/2 + bm/2 + m \\
&= cm^{\lg 3} - bm + bm/2 + m \\
&= cm^{\lg 3} - bm - m(b/2 - 1) \\
&\le cm^{\lg 3} - bm \qquad if\ m(b/2 - 1) \ge 0
\end{aligned}$$

if $m_0 = 1$, then $b \ge 2$

$$T(n) = O(m^{\lg 3}) = O((\lg n)^{\lg 3}) = O(\lg^{\lg 3} n)$$

109

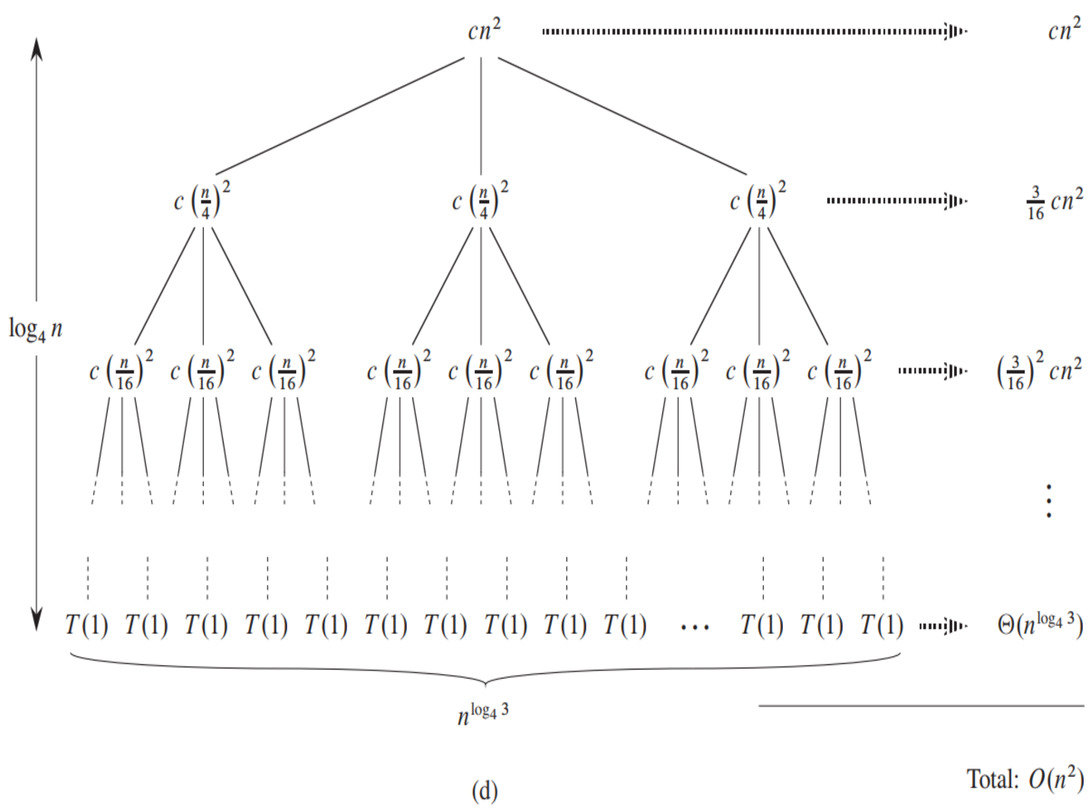
recursion tree

each node represents the cost of a single subproblem

sum the costs within each level

tolerate “sloppiness 敷衍”；容忍不良量

recursion tree for $T(n) = 3T(n/4) + cn^2$



(d)

cn^2 at the root represents the cost at the top level of recursion

cost for each of the three children of the root is $c(n/4)^2$

for a node at at depth i, $\text{subproblem\_size} = n/4^i$

$\text{subproblem\_size} = 1 = n/4^i \quad \leftrightarrow \quad 4^i = n \quad \leftrightarrow \quad \log_4 n = i$

so tree has $\log_4 n + 1$ levels $\quad$ $depth \in \{0, 1, 2, ..., \log_4 n\}$

at depth $i \in \{0,\ 1,\ 2,\ ...,\ \log_4 n - 1\}$, $\text{node\_cost} = c(\text{subproblem\_size})^2 = c(n/4^i)^2$
total cost at depth $i = \sum \text{node\_cost} = (num\_node\_per\_level)(node\_cost) = (3^i)c(n/4^i)^2 = (3/16)^i cn^2$

bottom level, at depth $i = \log_4 n$, num_nodes $= 3^i = 3^{\log_4 n} = n^{\log_4 3}$
因为 $\log_4(3^{\log_4 n}) = (\log_4 n)(\log_4 3) = \log_4(n^{\log_4 3})$

/* 对数

`https://mathinsight.org/logarithm_basics`

$\log_2(2^i) = i$
$2^{\log_2 i} =?$ $\quad$ $\log_2(2^{\log_2 i}) = \log_2 ?$ $\quad$ $2^{\log_2 ?} = 2^{\log_2 i}$ $\quad$ $? = i$
$2^{\log_2(ab)} = ab = 2^{\log_2 a} 2^{\log_2 b} = 2^{\log_2 a + \log_2 b}$ 积
$2^{\log_2(a/b)} = a/b = 2^{\log_2 a}/2^{\log_2 b} = 2^{\log_2 a - \log_2 b}$ 商
$2^{\log_2(a^b)} = a^b = (2^{\log_2 a})^b = 2^{b \log_2 a}$ 幂
let $\log_2 a = i$, then $2^i = a$, take log_c of both sides
$\log_c(2^i) = \log_c a$
$i \log_c 2 = \log_c a$
$\log_2 a \log_c 2 = \log_c a$

$$\log_2 a = \frac{\log_c a}{\log_c 2} \qquad \text{换底}$$

*/
each node contribute cost $T(1)$
$\text{bottom\_level\_total\_cost} = n^{\log_4 3} T(1) = \Theta(n^{\log_4 3})$

$$
\begin{aligned}
T(n) &= cost(depth \in \{0,\ 1,\ 2,\ ...,\ \log_4 n - 1\}) + cost(depth = \log_4 n) \\
&= \sum_{i=0}^{\log_4 n-1} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3}) \\
&< \sum_{i=0}^{\infty} \left(\frac{3}{16}\right)^i cn^2 + \Theta(n^{\log_4 3}) \\
&= \frac{1}{1-3/16} cn^2 + \Theta(n^{\log_4 3}) \\
&= \frac{1}{13/16} cn^2 + \Theta(n^{\log_4 3}) \\
&= \frac{16}{13} cn^2 + \Theta(n^{\log_4 3}) \\
&= O(n^2)
\end{aligned}
$$

$$
\begin{aligned}
s = a \ + \ ar \ + \ ar^2 \ + \ ar^3 \ + \ \cdots \ + \ ar^n&, \\
rs = \ ar \ + \ ar^2 \ + \ ar^3 \ + \ \cdots \ + \ ar^n \ + \ ar^{n+1}&, \\
s - rs = \ a \ - \ ar^{n+1}&, \\
s(1-r) = \ a(1 - r^{n+1})&, \\
s = \ a\left(\frac{1 - r^{n+1}}{1-r}\right) \quad (\text{if } r \neq 1)&.
\end{aligned}
$$

$$
if\ |r| < 1, \qquad \sum_{i=0}^{\infty} ar^i = \frac{a}{1-r}
$$

use substitution method to verify $T(n) = O(n)$ for

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

prove $T(n) \leq dn^2$
assume $T(n) \leq dn^2$ is true for all positive $m < n$, in particular for $m = \lfloor n/4 \rfloor$,
or equivalently assume $T(\lfloor n/4 \rfloor) \leq d\lfloor n/4 \rfloor^2$

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

$$if\ \frac{3}{16}d + c \leq d$$

$$c \leq \frac{16-3}{16}d = \frac{13}{16}d$$

$$\frac{16}{13}c \leq d$$

114
4.4-5
determine upper bound on recurrence $T(n) = T(n-1) + T(n/2) + n$

$$T(n) = T(n-1) + T(n/2) + n \leq T(n-1) + T(n-1) + n = 2T(n-1) + n$$

$$if\ n \geq 2, \quad then\ n/2 \leq n-1$$

for $T(n) \leq 2T(n-1) + n$:
at $depth = i \in \{1,\ 2,\ ,...,\ n\}$, $\quad$ num_node $= 2^i$, $\quad$ cost_per_node $= n - i$

$$\begin{aligned}
T(n) &= \sum_{i=0}^{n} 2^i(n-i) \\
&= n\sum_{i=0}^{n} 2^i - \sum_{i=0}^{n} i \cdot 2^i
\end{aligned}$$

/* colored sum S

$$\begin{aligned}
S &= 1\cdot 2^1 + \quad 2\cdot 2^2 + ... + (n-1)\cdot 2^{n-1} + n\cdot 2^n \\
2S &= \qquad\qquad 1\cdot 2^2 + ... + (n-2)\cdot 2^{n-1} + (n-1)\cdot 2^n \quad + n\cdot 2^{n+1} \\
S - 2S = -S &= 1\cdot 2^1 + \quad 1\cdot 2^2 + ... + 1\cdot 2^{n-1} \qquad + 1\cdot 2^n \qquad - n\cdot 2^{n+1}
\end{aligned}$$

$$-S = \sum_{i=1}^{n} 2^i - n\cdot 2^{n+1}$$

$$S = n\cdot 2^{n+1} - \sum_{i=1}^{n} 2^i$$

又因 $$\sum_{i=0}^{n} 2^i = 2^0 + \sum_{i=1}^{n} 2^i; \qquad \sum_{i=1}^{n} 2^i = \sum_{i=0}^{n} 2^i - 1$$

$$S = n\cdot 2^{n+1} - \left(\sum_{i=1}^{n} 2^i - 1\right) = n\cdot 2^{n+1} - \sum_{i=1}^{n} 2^i + 1$$

*/

$$
\begin{aligned}
T(n) &= n\sum_{i=0}^{n} 2^i - \sum_{i=0}^{n} i\cdot 2^i \\
&= n\sum_{i=0}^{n} 2^i - \left(n\cdot 2^{n+1} - \sum_{i=1}^{n} 2^i + 1\right) \\
&= (n+1)\sum_{i=0}^{n} 2^i - n\cdot 2^{n+1} - 1 \\
&= (n+1)\frac{1-2^{n+1}}{1-2} - n\cdot 2^{n+1} - 1 \\
&= (n+1)\frac{2^{n+1}-1}{2-1} - n\cdot 2^{n+1} - 1 \\
&= (n+1)(2^{n+1}-1) - n\cdot 2^{n+1} - 1 \\
&= n\cdot 2^{n+1} - n + 2^{n+1} - 1 - n\cdot 2^{n+1} - 1 \\
&= 2^{n+1} - n - 2 \\
&\le 2^{n+1} \\
&\le c2^n \qquad if\ c \ge 2,\ n \ge 0
\end{aligned}
$$

115
master theorem 主定理

$$a \ge 1, \quad b > 1 \quad T(n) = aT(n/b) + f(n)$$

一. 若 $\exists\epsilon > 0,\ f(n) = O(n^{\log_b a-\epsilon}),\quad$ 则 $T(n) = \Theta(n^{\log_b a})$
二. 若 $f(n) = \Theta(n^{\log_b a}),\quad$ 则 $T(n) = \Theta(n^{\log_b a}\lg n)$
三. 若 $\exists\epsilon > 0,\ f(n) = \Omega(n^{\log_b a+\epsilon}),\ \exists c < 1$ 和所有足够大的 n, 有$af(n/b) \le cf(n)$, 则 $T(n) = \Theta(f(n))$

对比 $f(n)$ 和 $n^{\log_b a}$, 解由两个函数中较大的决定:
1. $n^{\log_b a} > f(n)$, so $T(n) = \Theta(n^{\log_b a})$

2. $f(n)$ and $n^{\log_b a}$ are the same size, multiply by a logarithmic factor, $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(f(n) \lg n)$
3. $f(n) > n^{\log_b a}$, so $T(n) = f(n)$

technicality 技术问题, extra conditions

$$1. \quad \frac{n^{\log_b a}}{f(n)} > n^\epsilon, \quad \epsilon > 0$$

$$3. \quad \frac{f(n)}{n^{\log_b a}} > n^\epsilon, \quad \epsilon > 0$$

116
use master

$$T(n) = 9T(n/3) + n$$
$$a = 9$$
$$b = 3$$
$$n^{\log_b a} = n^{\log_3 9} = n^2 = \Theta(n^2)$$
$$f(n) = n = \Theta(n)$$
$$n^{\log_3 9} = n^2 > n = f(n)$$
$$f(n) = \Theta(n) = \Theta(n^{2-1}) = \Theta(n^{\log_3 9 - \epsilon}), \quad \epsilon = 1$$
$$T(n) = \Theta(n^{\log_3 9}) = \Theta(n^2) \qquad \textit{case } 1$$

$$\text{merge sort: } T(n) = 2T(n/2) + \Theta(n)$$
$$a = 2$$
$$b = 2$$
$$n^{\log_b a} = n^{\log_2 2} = n^1 = \Theta(n)$$
$$f(n) = \Theta(n) = \Theta(n^{\log_2 2})$$

$$T(n) = \Theta(n^{\log_2 2} \lg n) = \Theta(f(n) \lg n) = \Theta(n \lg n) \qquad \textit{case } 2$$

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

$$a = 3$$

$$b = 4$$

$$n^{\log_b a} = n^{\log_4 3} = n^{0.79248...} = O(n^{0.793})$$

$$f(n) = n \lg n = \Omega(n) \quad 因为 f(n) = n \lg n \geq cn = cg(n)$$

$$取\ c = 1,\ n_0 \geq 2,\ 则 \log_2 n \geq 1$$

$$f(n) = \Omega(n^1) = \Omega(n^{\log_4 3 + \epsilon}), \qquad \epsilon \approx 0.2$$

$$因为 1 = \log_4 3 + \epsilon \approx 0.79248... + 0.2$$

$$af(n/b) = 3f(n/4) = 3(n/4) \lg(n/4) \leq (3/4) n \lg n = cf(n), \qquad c = 3/4$$

$$T(n) = \Theta(f(n)) = \Theta(n \lg n) \qquad \textit{case } 3$$

https://www.youtube.com/watch?v=2H0GKdrIowU

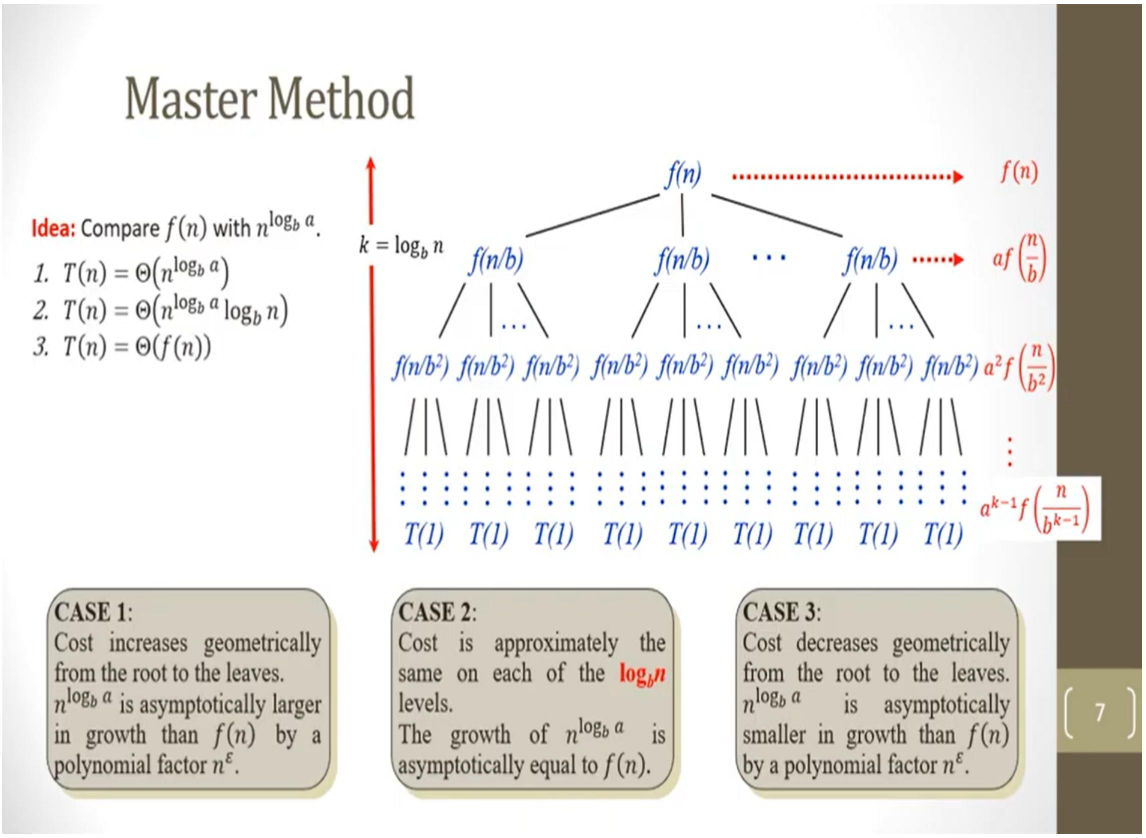
Master Theorem

Theorem

If $T(n) = aT\left(\left\lceil \frac{n}{b} \right\rceil\right) + O(n^d)$ (for constants $a > 0, b > 1, d \geq 0$), then:

$$T(n) = \begin{cases} O(n^d) & \text{if } d > \log_b a \\ O(n^d \log n) & \text{if } d = \log_b a \\ O(n^{\log_b a}) & \text{if } d < \log_b a \end{cases}$$

https://www.youtube.com/watch?v=zbf0llo3-YI

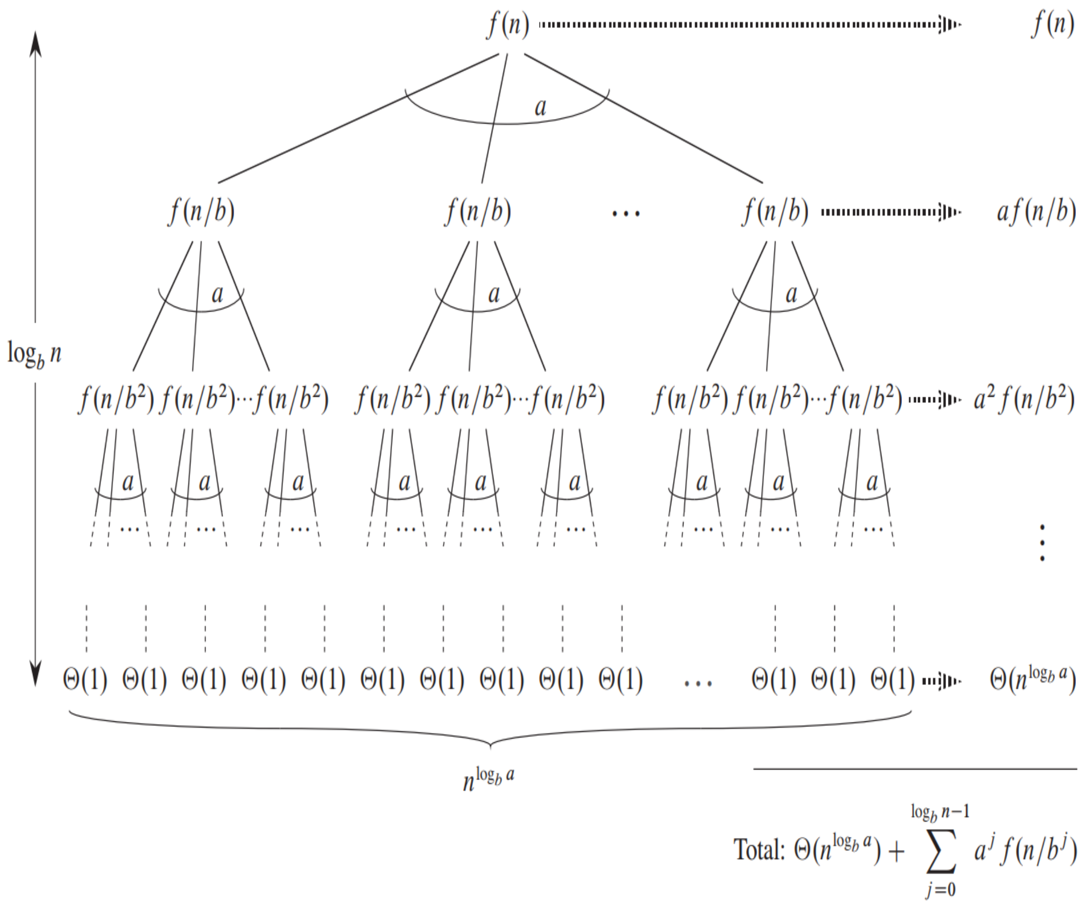


119
proof for exact powers

let $a \geq 1,\ \ b > 1,\ \ f(n) \geq 0,\, i \in \mathbb{Z}^+$
$T(n) = \begin{cases} \Theta(1) & if\ n = 1 \\ af(n/b) + f(n) & if\ n = b^i \end{cases}$

then

$$T(n) = \text{cost\_external} + \text{cost\_internal} = \Theta(n^{\log_b a}) + \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j)$$



root has cost $f(n)$, and root has a children, each child cost $f(n/b)$
at depth j, $\text{num\_node} = a^j$
$\text{cost\_per\_node} = f(n/b^j)$
at bottom level: $\text{subproblem\_size} = n/b^j = 1$, $\quad n = b^j$, $\quad \log_b n = j$
at bottom level: depth $= j = \log_b n$, $\quad \text{num\_node} = a^j = a^{\log_b n} = n^{\log_b a}$
/* 因 $\log_b(a^{\log_b n}) = (\log_b n)(\log_b a) = \log_b(n^{\log_b a})$ */

121

$$a \geq 1, \quad b > 1$$

$$\text{cost\_internal} = g(n) = \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j)$$

1. if $\exists \epsilon > 0,\ f(n) = O(n^{\log_b a - \epsilon}),$ then $g(n) = \Theta(n^{\log_b a})$
2. if $f(n) = \Theta(n^{\log_b a}),$ then $g(n) = \Theta(n^{\log_b a} \lg n)$
3. if $af(n/b) \leq cf(n)$ for some $c < 1$ and for all sufficiently large $n,$ then $g(n) = \Theta(f(n))$

case 1

$$f(n) = O(n^{\log_b a - \epsilon}) \quad \rightarrow \quad f(n/b^j) = O((n/b^j)^{\log_b a - \epsilon})$$

$$g(n) = \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j) = O\left(\sum_{j=0}^{\log_b n - 1} a^j \left(\frac{n}{b^j}\right)^{\log_b a - \epsilon}\right)$$

$$
\begin{aligned}
\sum_{j=0}^{\log_b n-1} a^j \left(\frac{n}{b^j}\right)^{\log_b a-\epsilon} &= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} a^j \left(\frac{1}{b^j}\right)^{\log_b a-\epsilon} \\
&= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} \left(\frac{a^j}{b^{j(\log_b a-\epsilon)}}\right) \\
&= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} \left(\frac{a}{b^{\log_b a-\epsilon}}\right)^j \\
&= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} \left(\frac{a}{(b^{\log_b a})(1/b^{\epsilon})}\right)^j \\
&= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} \left(\frac{ab^{\epsilon}}{a}\right)^j \\
&= n^{\log_b a-\epsilon} \sum_{j=0}^{\log_b n-1} (b^{\epsilon})^j \\
&= n^{\log_b a-\epsilon} \left(\frac{1-(b^{\epsilon})^{\log_b n}}{1-b^{\epsilon}}\right) \\
&= n^{\log_b a-\epsilon} \left(\frac{(b^{\log_b n})^{\epsilon}-1}{b^{\epsilon}-1}\right) \\
&= n^{\log_b a-\epsilon} \left(\frac{n^{\epsilon}-1}{b^{\epsilon}-1}\right) \\
&\text{since } b \text{ and } \epsilon \text{ are constants} \\
&= n^{\log_b a-\epsilon} O(n^{\epsilon}) \\
&= (n^{\log_b a})(1/n^{\epsilon}) O(n^{\epsilon}) \\
&= O(n^{\log_b a})
\end{aligned}
$$

$$
g(n) = O(n^{\log_b a})
$$

case 2

$$f(n) = \Theta(n^{\log_b a}) \quad \rightarrow \quad f(n/b^j) = \Theta((n/b^j)^{\log_b a})$$

$$g(n) = \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j) = \Theta\left(\sum_{j=0}^{\log_b n - 1} a^j \left(\frac{n}{b^j}\right)^{\log_b a}\right)$$

$$\begin{aligned}
\sum_{j=0}^{\log_b n - 1} a^j \left(\frac{n}{b^j}\right)^{\log_b a} &= n^{\log_b a} \sum_{j=0}^{\log_b n - 1} a^j \left(\frac{1}{b^j}\right)^{\log_b a} \\
&= n^{\log_b a} \sum_{j=0}^{\log_b n - 1} \left(\frac{a^j}{(b^{\log_b a})^j}\right) \\
&= n^{\log_b a} \sum_{j=0}^{\log_b n - 1} \left(\frac{a}{a}\right)^j \\
&= n^{\log_b a} \sum_{j=0}^{\log_b n - 1} 1 \\
&= n^{\log_b a} \log_b n
\end{aligned}$$

$$g(n) = \Theta(n^{\log_b a} \log_b n) = \Theta(n^{\log_b a} \lg n)$$

case 3
assume $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n
rewrite this assumption as $f(n/b) \leq (c/a)f(n)$

iterate j times, $f(n/b^j) \leq (c/a)^j f(n)$, or equivalently $a^j f(n/b^j) \leq c^j f(n)$

/* 可从假设$f(n/b) \leq (c/a)f(n)$推出以下内容:
因 $f(n/b)$ 中的参数比 $f(n)$ 中的参数多除了个 b
并且 $f(n)$ /* 少除的函数 */ 的前面比 $f(n/b)$ /* 多除的函数 */ 多乘了个系数 c/a
因此参数少除个 b 的函数前面会多乘一个系数 c/a

$$\begin{aligned} f(n/b^j) &\leq (c/a)^1 f(n/b^{j-1}) \\ &\leq (c/a)^2 f(n/b^{j-2}) \\ &\dots \\ &\leq (c/a)^j f(n/b^{j-j}) = (c/a)^j f(n) \end{aligned}$$

*/

$$\begin{aligned} g(n) &= \sum_{j=0}^{\log_b n - 1} a^j f(n/b^j) \\ &\leq \sum_{j=0}^{\log_b n - 1} c^j f(n) + O(1) \\ &\leq f(n) \sum_{j=0}^{\infty} c^j + O(1) \\ &= f(n) \left(\frac{1}{1-c} \right) + O(1) \\ &= O(f(n)) \end{aligned}$$

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