递推方程

李琦煜
杨思祺
张远鹏

算法设计与分析 2020 小班 17

2020 年 2 月 22 日

递归方程的求解方法

- 迭代法
  - 直接迭代
  - 换元迭代
  - 差消迭代
  - 递归树
- 主定理（Master Theorem）
- 尝试法
- 递推过程的归纳证明

迭代法

- 直接迭代：插入排序最坏情况下时间分析

$$W(n) = W(n-1) + n - 1, W(1) = 0 \Rightarrow W(n) = \frac{n(n-1)}{2}$$

迭代法

- 直接迭代：插入排序最坏情况下时间分析

$$W(n) = W(n-1) + n - 1, W(1) = 0 \Rightarrow W(n) = \frac{n(n-1)}{2}$$

- 换元迭代：二分归并排序最坏情况下时间分析

$$W(n) = 2W(n/2) + n - 1, W(1) = 0$$

设 $n = 2^k$，则有

$$\begin{aligned} W(n) &= 2^k W(1) + k2^k - (1 + 2 + \ldots + 2^{k-1}) \\ &= k2^k - 2^k + 1 = n \log n - n + 1 \end{aligned}$$

迭代法

- 直接迭代：插入排序最坏情况下时间分析

$$W(n) = W(n-1) + n - 1, W(1) = 0 \Rightarrow W(n) = \frac{n(n-1)}{2}$$

- 换元迭代：二分归并排序最坏情况下时间分析

$$W(n) = 2W(n/2) + n - 1, W(1) = 0$$

设 $n = 2^k$，则有

$$\begin{aligned} W(n) &= 2^k W(1) + k2^k - (1 + 2 + ... + 2^{k-1}) \\ &= k2^k - 2^k + 1 = n\log n - n + 1 \end{aligned}$$

- 使用迭代法得到的解（可能）需要通过数学归纳法验证

- 差消迭代：快速排序平均时间分析

$T(n) = \frac{2}{n}\sum_{i=1}^{n-1} T(i) + cn, n \geq 2, T(1) = 0$

$$nT(n) = 2\sum_{i=1}^{n-1} T(i) + cn^2$$

$$(n-1)T(n-1) = 2\sum_{i=1}^{n-2} T(i) + c(n-1)^2$$

两式相减得$nT(n) = (n+1)T(n-1) + 2cn - c$

$$故\frac{T(n)}{n+1} = \frac{T(n-1)}{n} + \frac{2cn-c}{n(n+1)}$$

$$= 2c[\frac{1}{n+1} + ... + \frac{1}{3}] + \frac{T(1)}{2} - O(\frac{1}{n}) = \Theta(\log n)$$

故 $T(n) = \Theta(n \log n)$

练习：求解递推方程

- 求解递推方程 $T(n) = 2T(\sqrt{n}) + \log n$

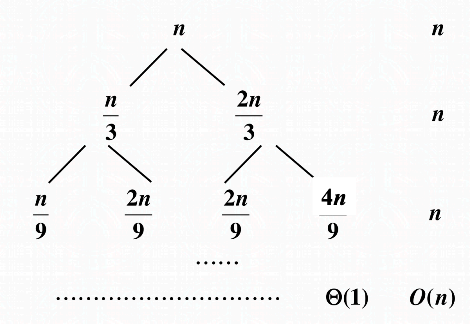
练习：求解递推方程

- 求解递推方程 $T(n) = 2T(\sqrt{n}) + \log n$
- 令 $m = \log n$, $T(2^m) = 2T(2^{m/2}) + m$
- 令 $S(m) = T(2^m)$, 则有 $S(m) = 2S(m/2) + m = \Theta(m \log m)$
- $T(n) = T(2^m) = S(m) = \Theta(m \log m) = \Theta(\log n \log \log n)$

迭代模型: 递归树

- 可以处理子问题规模不同的情况

$$T(n) = T(n/3) + T(2n/3) + n$$



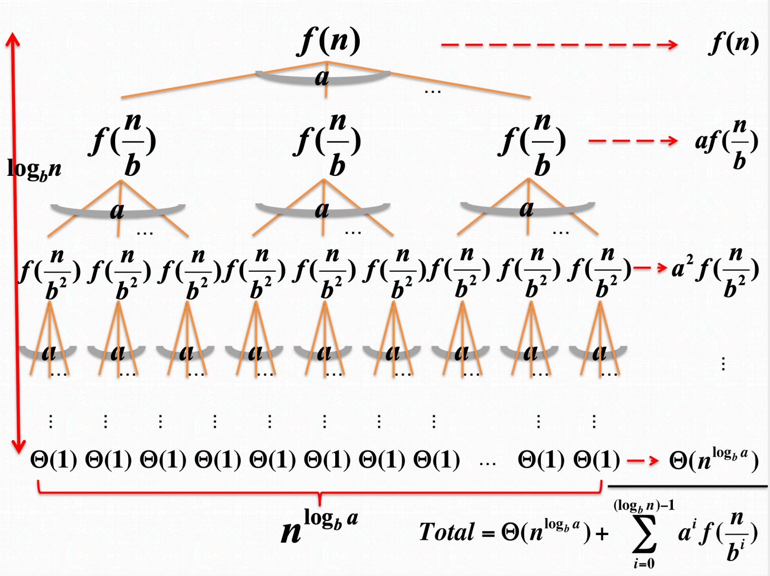
层数 $k \le \log_{3/2} n$, 故 $T(n) = O(n \log n)$

- 递归树不平衡时可选取最长（短）路径求和求得一个上（下）界

主定理 (Master Theorem)

- 处理一类子问题规模相同的递推方程
- $T(n) = aT(n/b) + f(n)(a \geq 1, b > 1, T(n) \geq 0)$
  - $f(n) = O(n^{\log_b a - \varepsilon}), \varepsilon > 0 \Rightarrow T(n) = \Theta(n^{\log_b a})$
  - $f(n) = \Theta(n^{\log_b a} \log^k n), k \geq 0 \Rightarrow T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$ (扩展的第二类情况)
  - $f(n) = \Omega(n^{\log_b a + \varepsilon}), \varepsilon > 0, \exists c < 1, \forall n > n_0, af(n/b) \leq cf(n) \Rightarrow T(n) = \Theta(f(n))$
- $f(n) = \frac{n^{\log_b a}}{\log n}$ 等不能归入以上任何一类
- $T(n) = 27T(n/3) + n^3$ 等不满足第三类 c 条件，需要用递归树

主定理 (Master Theorem) 的证明



主定理 (Master Theorem) 的证明

- $T(n) = c_1 n^{\log_b a} + \sum_{i=0}^{\log_b n - 1} a^i f(\frac{n}{b^i})$

主定理 (Master Theorem) 的证明

- $T(n) = c_1 n^{\log_b a} + \sum_{i=0}^{\log_b n - 1} a^i f(\frac{n}{b^i})$
- Case 1: $f(n) = O(n^{\log_b a - \varepsilon})$

$$T(n) = c_1 n^{\log_b a} + O(n^{\log_b a - \varepsilon} \sum_{i=0}^{\log_b n - 1} (b^\varepsilon)^i)$$

$$= c_1 n^{\log_b a} + O(n^{\log_b a - \varepsilon} \frac{b^{\varepsilon \log_b n} - 1}{b^\varepsilon - 1}) = \Theta(n^{\log_b a})$$

主定理 (Master Theorem) 的证明

- $T(n) = c_1 n^{\log_b a} + \sum_{i=0}^{\log_b n-1} a^i f(\frac{n}{b^i})$
- Case 1: $f(n) = O(n^{\log_b a-\varepsilon})$

$$T(n) = c_1 n^{\log_b a} + O(n^{\log_b a-\varepsilon} \sum_{i=0}^{\log_b n-1} (b^\varepsilon)^i)$$

$$= c_1 n^{\log_b a} + O(n^{\log_b a-\varepsilon} \frac{b^{\varepsilon \log_b n} - 1}{b^\varepsilon - 1}) = \Theta(n^{\log_b a})$$

- Case 2: $f(n) = \Theta(n^{\log_b a} \log^k n)$

$$T(n) = c_1 n^{\log_b a} + \Theta(n^{\log_b a} \sum_{i=0}^{\log_b n-1} \log^k(\frac{n}{b^i})) = \Theta(n \log_b a \log^{k+1} n)$$

主定理 (Master Theorem) 的证明

- $T(n) = c_1 n^{\log_b a} + \sum_{i=0}^{\log_b n-1} a^i f(\frac{n}{b^i})$
- Case 1: $f(n) = O(n^{\log_b a-\varepsilon})$

$$T(n) = c_1 n^{\log_b a} + O(n^{\log_b a-\varepsilon} \sum_{i=0}^{\log_b n-1} (b^\varepsilon)^i)$$

$$= c_1 n^{\log_b a} + O(n^{\log_b a-\varepsilon} \frac{b^{\varepsilon \log_b n} - 1}{b^\varepsilon - 1}) = \Theta(n^{\log_b a})$$

- Case 2: $f(n) = \Theta(n^{\log_b a} \log^k n)$

$$T(n) = c_1 n^{\log_b a} + \Theta(n^{\log_b a} \sum_{i=0}^{\log_b n-1} \log^k(\frac{n}{b^i})) = \Theta(n \log_b a \log^{k+1} n)$$

- Case 3: $f(n) = \Omega(n^{\log_b a+\varepsilon}), \exists 0 < c < 1,$ 使得$af(\frac{n}{b}) \le cf(n)$

$$T(n) \le c_1 n^{\log_b a} + \sum_{i=0}^{\log_b n-1} c^i f(n) = c_1 n^{\log_b a} + f(c)\frac{c^{\log_b n} - 1}{c - 1}$$

$$= c_1 n^{\log_b a} + \Theta(f(n)) = \Theta(f(n))$$

主定理的证明

- 下面引入 $\lceil\rceil$ 和 $\lfloor\rfloor$ 将 n 推广到非 b 的幂的情形，即 $T(n) = aT(\lceil n/b\rceil) + f(n)$ 和 $T(n) = aT(\lfloor n/b\rfloor) + f(n)$
- 设 $n_j = \begin{cases} n, & if\ j = 0 \\ \lceil n_{j-1}/b\rceil, else \end{cases}$，由于 $\lceil x\rceil \le x + 1$,

  $n_j \le \frac{n}{b^j} + \sum_{i=0}^{j-1} \frac{1}{b^i} < \frac{n}{b^j} + \sum_{i=0}^{\infty} \frac{1}{b^i} = \frac{n}{b^j} + \frac{b}{b-1}$

  $n_{\lfloor \log_b n\rfloor} < \frac{n}{b^{\lfloor \log_b n\rfloor}} + \frac{b}{b-1} < \frac{n}{b^{n-1}} + \frac{b}{b-1} = b + \frac{b}{b-1} = O(1)$

  故 $T(n) = \Theta(n^{\log_b a}) + \sum_{j=0}^{\lfloor \log_b n\rfloor} a^j f(n_j)$
- Case 3 与上述证明类似。对于 Case 2, $j \le \lfloor \log_b n\rfloor \Rightarrow b^j/n \le 1$
  故 $f(n_j) < c(\frac{n}{b^j} + \frac{b}{b-1})^{\log_b a}) = c(\frac{n}{b^j}(1 + \frac{b^j}{n} \cdot \frac{b}{b-1}))^{\log_b a} =$
  $c(\frac{n^{\log_b a}}{a^j})(1 + \frac{b^j}{n} \cdot \frac{b}{b-1})^{\log_b a} \le c(\frac{n^{\log_b a}}{a^j})(1 + \frac{b}{b-1})^{\log_b a} = O(\frac{n^{\log_b a}}{a^j})$
  类似的对于 Case 1 可证 $f(n_j) = O(n^{\log_b a-\varepsilon})$ ■

通用定理 (Akra-Bazzi Theorem)

对形如 $T(n) = \sum_{i=0}^{k} a_i T(\frac{n}{b_i}) + f(n)(a_i \geq 1, b_i > 1)$ 的递推公式，若

$\exists! p > 0$ 使得 $\sum_{i=0}^{k} \frac{a_i}{b_i^p} = 1$ 则有

- $f(n) = O(n^{p-\varepsilon}) \Rightarrow T(n) = \Theta(n^p)$
- $f(n) = \Theta(n^p \log^k n) \Rightarrow T(n) = \Theta(n^p \log^{k+1} n)$
- $f(n) = \Omega(n^{p+\varepsilon}) \Rightarrow T(n) = \Theta(f(n))$

推论 1: $\sum_{i=0}^{k} \frac{a_i}{b_i} = 1, f(n) = \Theta(n \log^k n) \Rightarrow T(n) = \Theta(n \log^{k+1} n)$

推论 2: $\sum_{i=0}^{k} \frac{a_i}{b_i} < 1, f(n) = \Omega(n) \Rightarrow T(n) = \Theta(f(n))$

更加一般的形式[1] [2]

$$T(n) = g(n) + \sum_{i=1}^{k} a_i T(b_i n + h_i(n))$$

$$\text{其中} a_i > 0, 0 < b_i < 1, |g(n)| \in O(n^c), |h_i(n)| \in O(\frac{n}{\log^2 n})$$

通解形式为

$$T(n) = \Theta(n^p(1 + \int_1^n \frac{g(u)}{u^{p+1}du}), \text{其中} p \text{满足} \sum_{i=1}^{k} a_i b_i^p = 1$$

[1]Proof: https://people.mpi-inf.mpg.de/~mehlhorn/DatAlg2008/NewMasterTheorem.pdf

[2]Example: https://www.blogcyberini.com/2017/07/metodo-de-akra-bazzi.html

例题

- $T(n) = T(n/3) + T(2n/3) + n$
  $\frac{a_1}{b_1} + \frac{a_2}{b_2} = 1, f(n) = n = \Theta(n)$, 由推论 1 得 $T(n) = \Theta(n \log n)$
- $T(n) = \begin{cases} n^2 + \frac{7}{4}T(\lfloor \frac{1}{2}n \rfloor) + T(\lceil \frac{3}{4}n \rceil), if\ n \geq 4, \\ 1, else \end{cases}$

$$T(n) \in \Theta(x^p(1 + \int_1^x \frac{g(u)}{u^{p+1}}du)) = \Theta(x^2(1 + \int_1^x \frac{u^2}{u^3}dx))$$
$$= \Theta(x^2(1 + \ln x)) = \Theta(x^2 \log x)$$

尝试法: 猜测解的形式

- "没有办法的办法"
- $T(n) = \frac{2}{n}\sum\limits_{i=1}^{n-1} T(i) + \Theta(n)$
  1. $T(n) = C, RHS = \frac{2}{n}C(n-1) + O(n) = 2C - \frac{2C}{n} + \Theta(n)$
  2. $T(n) = cn, RHS = \frac{2}{n}\sum\limits_{i=1}^{n-1} ci + \Theta(n) = cn - c + \Theta(n)$
  3. $T(n) = cn^2, RHS = \frac{2}{n}\sum\limits_{i=1}^{n-1} ci^2 + \Theta(n) = \frac{2}{n}[\frac{cn^3}{3} + \Theta(n^2)] + \Theta(n) = \frac{2}{3}n^2 + \Theta(n)$
  4. $T(n) = cn\log n, RHS = \frac{2c}{n}\sum\limits_{i=1}^{n-1} i\log i + \Theta(n) = cn\log n + \Theta(n)$

     注：$\sum\limits_{i=1}^{n-1} i\log i$ 可用积分逼近（上下界）
- 尝试 (猜测) 递推方程的解应用归纳法严格证明

递推方程的严格归纳证明

- 归纳法求解递推方程的一般步骤
  - 猜测解的形式
  - 用数学归纳法证明
    - 找出使解有效的常数
  - 确定常数使边界条件成立
- 常用技巧
  - 通过引入低阶项获得更紧的解的形式

递推方程的归纳证明

- $T(n) = 4T(n/2) + n$，证明 $T(n) = O(n^2)$
- 假设对于所有的 $k < n$，$T(k) \leq ck^2$

$$T(n) = 4T(n/2) + n \leq cn^2 + n = O(n^2)$$

- 伪证，必须证明完全相同的形式

更紧的上界

- 加强归纳假设：减去一个低阶项
- 假设对于所有的 $k < n$，$T(k) \le c_1 k^2 - c_2 k$

$$
\begin{aligned}
T(n) &= 4T(n/2) + n \\
&\le c_1 n^2 - 2c_2 n + n \\
&= c_1 n^2 - c_2 n - (c_2 n - n) \\
&\le c_1 n^2 - c_2 n \text{当} c_2 > 1 \text{时}
\end{aligned}
$$

- 取 c_1 足够大来处理初始情况
- 再证 $T(n) = \Omega(n^2)$，可得 $T(n) = \Theta(n^2)$