Homework 1

PB20000112　黄天一

1. 假定 $f(n)$ 和 $g(n)$ 都是渐进非负函数, 判断下列等式或陈述是否一定是正确的, 并简要解释你的答案.
   (a) $f(n) = O(f(n)^2)$.
   (b) $f(n) + g(n) = \Theta(\max(f(n), g(n)))$.
   (c) $f(n) + O(f(n)) = \Theta(f(n))$.
   (d) if $f(n) = \Omega(g(n))$, then $g(n) = o(f(n))$.

Solution.

(a) 错误. 令 $f(n) = \dfrac{1}{n}$, 则 $\lim\limits_{n\to+\infty} \dfrac{f(n)}{f(n)^2} = \lim\limits_{n\to+\infty} n = +\infty$, 即 $f(n) = \omega(f(n)^2)$, 与 $f(n) = O(f(n)^2)$ 相悖.

(b) 正确. 由于

$$(f(n) + g(n)) - \max(f(n), g(n)) = \min(f(n), g(n)) \geqslant 0;$$

$$\max(f(n), g(n)) - \frac{f(n) + g(n)}{2} = \frac{\max(f(n), g(n)) - \min(f(n), g(n))}{2} \geqslant 0,$$

因此 $\forall n \in \mathbb{N}$, $\dfrac{f(n)+g(n)}{2} \leqslant \max(f(n), g(n)) \leqslant f(n) + g(n)$, 即 $f(n) + g(n) = \Theta(\max(f(n), g(n)))$.

(c) 正确. 由定义得 $\exists n_0 \in \mathbb{N}$, $\forall n \geqslant n_0$, $\exists c \in \mathbb{R}^+$, s.t. $0 \leqslant O(f(n)) \leqslant cf(n)$. 因此 $f(n) \leqslant f(n) + O(f(n)) \leqslant (c+1)f(n)$, 即 $f(n) + O(f(n)) = \Theta(f(n))$.

(d) 错误. 设 $f(n) = 2n$, $g(n) = n$, 则 $\forall n \in \mathbb{N}, 0 < g(n) < f(n) \Rightarrow f(n) = \Omega(g(n))$, 但是 $\lim\limits_{n\to\infty} \dfrac{g(n)}{f(n)} = \dfrac{1}{2} \neq 0 \Rightarrow g(n) \neq o(f(n))$.

2. 时间复杂度.
   (a) 证明 $\lg(n!) = \Theta(n \lg n)$, 并证明 $n! = \omega(2^n)$ 且 $n! = o(n^n)$.
   (b) 使用代入法证明 $T(n) = T(\lceil n/2 \rceil) + 1$ 的解为 $O(\lg n)$.
   (c) 对递归式 $T(n) = T(n-a) + T(a) + cn$, 利用递归树给出一个渐进紧确解, 其中 $a \geqslant 1$ 和 $c > 0$ 为常数.
   (d) 主方法能应用于递归式 $T(n) = 4T(n/2) + n^2 \lg n$ 吗? 请说明为什么可以或者为什么不可以. 给出这个递归式的一个渐进上界.

Solution.

(a) 首先有 $\lg(n!) = \sum_{k=1}^{n} \lg k \leqslant \sum_{k=1}^{n} \lg n = n\lg n$, 故 $\lg(n!) = O(n\lg n)$. 注意到 $\lg(n!)$ 的部分和

$$\sum_{k=\lfloor n/2 \rfloor}^{n} \lg k \geqslant \sum_{k=\lfloor n/2 \rfloor}^{n} \lg \left\lfloor \frac{n}{2} \right\rfloor \geqslant \frac{n}{2} \lg \left\lfloor \frac{n}{2} \right\rfloor = \Theta(n\lg n),$$

因此 $\lg(n!) = \Omega(n\lg n)$. 综上所述, $\lg(n!) = \Theta(n\lg n)$.

应用 Stirling 公式: $n! = \sqrt{2\pi n}\left(\frac{n}{\mathrm{e}}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right)$, 可得

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

$$\lim_{n\to+\infty} \frac{n!}{n^n} = \lim_{n\to+\infty} \frac{\sqrt{2\pi n}}{\mathrm{e}^n} \left(1 + \Theta\left(\frac{1}{n}\right)\right) = 0,$$

因此 $n! = \omega(2^n)$, $n! = o(n^n)$.

(b) 设 $T(n) \leqslant a\lg n + b$, $a > 0, b \in \mathbb{R}$ 为待定参数. 应用归纳法, 可得

$$T(n) = T(\lceil n/2 \rceil) + 1 \leqslant a\lg(\lceil n/2 \rceil) + b + 1 \leqslant a\lg n + b - (a\lg\frac{3}{2} - 1),$$

取 $a = \frac{1}{2}$, 则 $a\lg\frac{3}{2} - 1 = \frac{1}{2}\lg\frac{3}{8} < 0$, 此时 $T(n) < a\lg n + b$, 归纳成立. 因此 $T(n) \leqslant \frac{1}{2}\lg n + O(1)$, 即 $T(n) = O(\lg n)$.

(c) 题设递归式的决策树如 Figure 1所示, 其中 $r = n (\mathrm{mod}\ a)$. 决策树的高度为 $h = \lceil n/a \rceil$, 叶子结点为一个 $T(r)$ 和 h 个 $T(a)$. 因此

$$T(n) = hT(a) + T(r) + \sum_{k=0}^{h-2} c(n - ka) = cn\lceil n/a \rceil + \Theta(n) = \Theta(n^2).$$

(d) 不能应用 Master Theorem. 注意到 $n^2\lg n = \Omega(n^2)$, 但是对于 $\forall \varepsilon > 0$, $\lg n = o(n^{\varepsilon})$, 因此不存在 $\epsilon > 0$, s.t. $n^2 \lg n = O(n^{2-\epsilon})$ 或 $n^2\lg n = \Omega(n^{2+\epsilon})$, 不满足 Master Theorem 的条件.

猜测: 题设递推式的一个渐进上界为 $n^2\lg^2 n$, 设 $T(n) \leqslant an^2\lg^2 n$, $a > 0$. 应用归纳法, 可得当 $n \geqslant 2$ 时,

$$\begin{aligned} T(n) &= 4T(n/2) + n^2\lg n \\ &\leqslant a^2n^2\lg^2(n/2) + n^2\lg n \\ &= a^2n^2\lg^2 n + a^2n^2 + (1 - 2a^2)n^2\lg n \\ &\leqslant a^2n^2\lg^2 n + (1 - a^2)n^2\lg n. \end{aligned}$$

令 $a = 1$, 则 $T(n) \leqslant n^2\lg^2 n$, 归纳成立. 因此 $T(n) = O(n^2\lg^2 n)$.

3. 对下列递归式, 使用主方法求出渐进紧确解.

(a) $T(n) = 2T(n/4) + \sqrt{n}$.

(b) $T(n) = 2T(n/4) + n^2$.

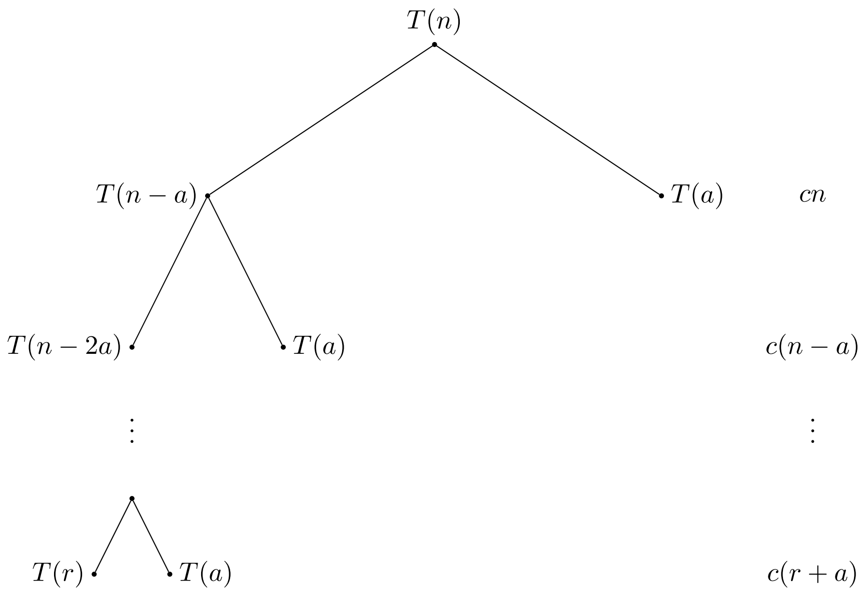

Figure 1: $T(n) = T(n-a) + T(a) + cn$ 的决策树.

Solution. 将题设递归式均视为 $T(n) = aT(n/b) + f(n)$ 的形式.

(a) $f(n) = \sqrt{n}$, $a = 2$, $b = 4$. 因此 $f(n) = \Theta(n^{\log_b a})$. 应用 Master Theorem 可得 $T(n) = \Theta(n^{\log_b a} \lg n) = \Theta(\sqrt{n} \lg n)$.

(b) $f(n) = n^2$, $a = 2$, $b = 4$. 因此对于 $\varepsilon = 1$ 有 $f(n) = \Omega(n^{\log_b a + 1})$. 又因为 $af(n/b) = n^2/8 < f(n)$, 满足条件. 应用 Master Theorem 可得 $T(n) = \Theta(n^2)$.

4. 考虑以下查找问题:
输入: n 个数的一个序列 $A = a_1, a_2, \cdots, a_n$ 和一个值 v.
输出: 下标 i 使得 $v = A[i]$ 或者当 v 不在 A 中出现时, v 为特殊值 NIL.

(a) 写出**线性查找**的伪代码, 它扫描整个序列来查找 v. 使用一个 Loop Invariant(循环不变式) 来证明你的算法是正确的.

(b) 假定 v 等可能地为数组的任意元素, 平均需要检查序列的多少元素? 最坏情况又如何呢? 用 Θ 记号给出线性查找的平均情况和最坏运行时间.

Algorithm 1: Linear Search.

Input: a sequence of n numbers $A = a_1, a_2, \cdots, a_n$ and a value v.
Output: subscript i such that $v = A[i]$ or when v does not occur in A, v is the special value NIL.

```
for i = 1 to A.length do
    if A[i] == v then
        return i;
return NIL;
```

Solution.

(a) 线性查找算法如 Algorithm 1所示. 给定一个 Loop Invariant: 在 Algorithm 1的第 i 次迭代开始前, 算法返回 $i-1$ 并终止或子列 $A[1..i-1]$ 不包含元素 v. 证明如下:

初始化: 在迭代开始之前, 子列 $A[1..0]$ 为空, 显然不包含元素 v;
保持: 在第 i 此迭代开始前, 设循环不变式为真. 若算法终止并返回 $i-1$, 则迭代结束; 若 $A[1..i-1]$ 不包含元素 v, 则进行第 i 次迭代. 若 $A[i]=v$, 则算法返回下标 i 并终止代; 若 $A[i]\neq v$, 则此时 $A[1..i]$ 不包含元素 v, 循环不变式成立, 迭代继续.
终止: 算法终止时, 若 $i\leqslant n$, 则算法返回下标 i, 此时 v 在序列 A 中; 若 $i=n+1$, 则根据循环不变式, $A[1..i-1]=A[1..n]$ 不包含元素 v, 即序列 A 不包含 v, 查找失败, 返回特殊值 NIL.
综上所述, 线性查找算法 Algorithm 1是正确的.

(b) 设 $1\leqslant i\leqslant n$, 若 v 在序列 A 中, 则 $A[i]=v$ 的概率为 $p_i=\dfrac{1}{n}$, 此时需检查元素 $A[1..i]$, 共 i 个元素. 设检查序列元素的个数为 X, 则期望为

$$\mathbb{E}[X]=\sum_{i=1}^{n} ip_i=\frac{1}{n}\sum_{i=1}^{n} i=\frac{n+1}{2}=\Theta(n/2).$$

最坏情况下, $A[n]=v$ 或 v 不在序列 A 中, 此时需要检查序列中的全部元素, 即需要检查 $n=\Theta(n)$ 次.

5. 堆排序:
对于一个按升序排列的包含 n 个元素的有序数组 A 来说, HEAPSORT 的时间复杂度是多少? 如果 A 是降序的呢? 请简要分析并给出结果.

Solution.

(a) A 按升序排列. 则对于 $1\leqslant i\leqslant\lfloor n/2\rfloor$, 均有 $A[i]\leqslant A[j]$, $i<j$. 因此在建堆过程中的每次移动中, 元素 $A[i]$ 都需要从当前位置移动到叶子处 (因为此前只移动了下标大于 i 的元素), 移动次数为 $n_i=\lfloor\lg n\rfloor-\lfloor\lg i\rfloor$. 元素比较时间为 $\lfloor n/2\rfloor=\Theta(n)$, 因此

$$T(n)=\Theta(n)+\sum_{i=1}^{\lfloor n/2\rfloor} n_i=\sum_{i=1}^{\lfloor n/2\rfloor}(\lfloor\lg n\rfloor-\lfloor\lg i\rfloor)=\Theta\left(\sum_{i=1}^{\lfloor n/2\rfloor}\lg(n/i)\right)=\Theta(\lg(n!))=\Theta(n\lg n).$$

其中最后一个等式运用了 2(a) 的结论.

(b) A 按降序排列. 则对于 $1\leqslant i\leqslant\lfloor n/2\rfloor$, 均有 $A[i]\geqslant A[2i], A[2i+1]$, 此时 A 即为 MAXHEAP. 因此建堆过程中只需要比较元素, 不需要移动元素. 因此 $T(n)=\lfloor n/2\rfloor=\Theta(n)$.

6. 快速排序:

(a) 假设快速排序的每一层所做的划分比例都是 $1-\alpha:\alpha$, 其中 $0<\alpha\leqslant 1/2$ 且是一个常数. 试证明: 在相应的递归树中, 叶结点的最小深度大约是 $-\lg n/\lg\alpha$, 最大深度大约是 $-\lg n/\lg(1-\alpha)$(无需考虑舍入问题).

(b) 试证明: 在一个随机输入数组上, 对于任何常数 $0<\alpha\leqslant 1/2$, PARTITION 产生比 $1-\alpha:\alpha$ 更平衡的划分的概率约为 $1-2\alpha$.

Proof.

(a) PARTITION 算法的时间复杂度为 $\Theta(n)$. 设 QUICKSORT 的时间复杂度为 $T(n)$, 则有 $T(n)=T(\alpha n)+T((1-\alpha)n)+\Theta(n)$. 设递归树中 $T(n)$ 的 LEFTCHILD 结点为 $T(\alpha n)$, RIGHTCHILD 结点为 $T((1-\alpha)n)$. 由 $0<\alpha\leqslant 1/2$ 可得, 从 ROOT 开始始终沿 LEFT-CHILD 方向的路径上, CHILD 结点对应的时间复杂度的下降速率最快 $(1/\alpha)$, 因此该

路径上的叶结点具有最小深度 $h_{\min} = \log_{1/\alpha} n = -\lg n/\lg\alpha$; 从 ROOT 开始始终沿 RIGHTCHILD 方向的路径上, CHILD 结点对应的时间复杂度的下降速率最慢 $(1/(1-\alpha))$, 因此该路径上的叶结点具有最大深度 $h_{\max} = \log_{1/(1-\alpha)} n = -\lg n/\lg(1-\alpha)$.

(b) 递归树叶结点的深度的极差 $\Delta h = h_{\max} - h_{\min}$ 越小, 递归树对应的 PARTITION 的划分越平衡. 由于

$$\Delta h = \frac{\lg n}{\lg\alpha} - \frac{\lg n}{\lg(1-\alpha)} = \lg n\left(\frac{1}{\lg\alpha} - \frac{1}{\lg(1-\alpha)}\right),$$

因为 $\lg\alpha, -\dfrac{1}{\lg(1-\alpha)}$ 在区间 $(0, 1/2]$ 上均为关于 α 的递减函数, 因此 α 越大, PARTITION 越平衡. 设对于 $\alpha \in (0, 1/2]$, PARTITION 产生比 $1-\alpha : \alpha$ 更平衡的划分的概率为 p, 则 $p = \dfrac{1/2-\alpha}{1/2} = 1-2\alpha$.

$\square$