算法基础 2022 春 Homework 1

任课老师：陈雪 due: Mar 3, 15:30

问题 1 (15 分). 给出辗转相除法的时间复杂度上界 (5 分)，写出详细分析过程 (10 分)。辗转相除法的伪代码如下：

Algorithm 1 Euclid's algorithm

```
function EUCLID(a, b)
    if b = 0 then
        return a
    else
        return EUCLID(b, a mod b)
    end if
end function
```

解答:

不失一般性，假设 $a \geq b$(如果 $b > a$ 只会多一层函数调用，不影响分析结果)。令 $t_1 = a$，$t_2 = b$。对 $i > 2$, 让 $t_i = t_{i-2} \mod t_{i-1}$,。由 t_i 的构造方式可知，t_{i+1}，t_{i+2} 就是第 i 次调用函数时的输入。辗转相除法的正确性保证，数列会一直下降到 $t_{n+2} = 0$。因此一共调用了 n 次函数。

由 $t_{i-1} = t_{i-3} \mod t_{i-2}$ 知，$t_{i-1} \leq t_{i-2}$。进而 $t_i \leq t_{i-2} - t_{i-1}$，所以 $t_i + t_{i-1} \leq t_{i-2}$。又 $t_n \geq 1, t_{n+1} \geq 1$，归纳得 $t_i \geq F_{n-i+2}$（其中 F_i 是斐波那契数列）。得到 $t_1 \geq F_{n+1} \geq \frac{1}{\sqrt{5}}((\frac{1+\sqrt{5}}{2})^{n+1} + (\frac{1-\sqrt{5}}{2})^{n+1}))$，故 $n = O(\log t_1) = O(\log a + \log b)$。

每层函数活动时间为 $O(1)$，最终得时间复杂度为 $O(\log a + \log b)$。

问题 2 (15 分). 根据增长的阶来从小到大排序下面的函数 (5 分)，给出分析过程 (10 分)：(log 以 2 为底数)

$$2^{\Theta(\frac{\log n}{\log\log n})}, \quad \log(n!), \quad (\log n)^{\Theta(\log n)}, \quad n^{\Theta((\log\log\log n))}, \quad \sqrt{n}, \quad n^2。$$

提示: 斯特林公式：$n! \approx \sqrt{2\pi n} \cdot (\frac{n}{e})^n$

解答:

对所有函数取对数得,$\Theta(\frac{\log n}{\log\log n})$, $\log n + \log\log n$, $(\log\log n)\cdot\Theta(\log n)$, $\Theta(\log\log\log n)\cdot\log n$, $\frac{1}{2}\log n$, $2\log n$。

从小到大排序为 $2^{\Theta(\frac{\log n}{\log\log n})}$, $\sqrt{n}$, $\log(n!)$, n^2, $n^{\Theta(\log\log\log n)}$, $(\log n)^{\Theta(\log n)}$。

问题 3 (40 分)**.** 在课上我们观察到了 Shell Sort 的一个重要性质：
一旦对数组 A 调用了步长为 h 的选择排序，A 数组在 Shell Sort 的未来时刻总是有 (注：此处的未来时刻指调用 $\textsc{InsertionSort}(h')$ 之后，而不包括 $\textsc{InsertionSort}(h')$ 的进行过程中):

$$1 \le i \le n-i, A[i] \le A[i+h] \tag{1}$$

使用上述观察到的性质，我们在此题完成课上结果的证明：
假设 A 的元素各不相同，给定步长 h_1, h_2, h_3，其中 h_3 和 h_2 互质，考虑在数组 A 上首先调用了 $\textsc{InsertionSort}(h_3)$ 和 $\textsc{InsertionSort}(h_2)$, 然后调用 $\textsc{InsertionSort}(h_1)$。

(a) 证明在调用 $\textsc{InsertionSort}(h_1)$ 之前，数组 A 满足如下性质：对任意的 j 和 ℓ 满足 $j > \ell + 2h_2h_3$ 时，总会有 $A[j] > A[\ell]$。

提示：将 h_2, h_3 代入性质(1)，可得 $A[i] \le A[i+h_2]$，$A[i] \le A[i+h_3]$。

附加题：证明条件加强到 $j > \ell + h_2h_3$ 时，也满足上述关系。(10 分)

(b) 在 (a) 的基础上，证明最后调用 $\textsc{InsertionSort}(h_1)$ 所需时间为 $O(n \cdot \frac{h_3h_2}{h_1})$。

解答:

(a) 对 $A[j]$ 反复使用性质(1)，得到对 $\forall a \ge 0, b \ge 0$，$A[j] \ge A[j - ah_2 - bh_3]$,。所以要证对任意的 j 和 ℓ 满足 $j > \ell + 2h_2h_3$ 时，总会有 $A[j] > A[\ell]$，

可证 $\forall \ell < j - 2h_2h_3$, $\exists a \ge 0, b \ge 0$, $j - \ell = ah_2 + bh_3$。

首先，因为 h_2, h_3 互质，可知 $\{ah_2 + bh_3 \mod h_2h_3 | 0 \le a \le h_3, 0 \le b \le h_2,\} = \{0, 1, 2, 3, ..., h_2h_3 - 1\}$。

对 $\forall n \geq 2h_1h_2$，$\exists a_0 \geq 0, b_0 \geq 0,\ a_0h_2 + b_0h_3 \mod h_2h_3 = n \mod h_2h_3$。同时，当 $0 \leq a \leq h_3, 0 \leq b \leq h_2$ 时，$ah_2 + bh_3 \leq 2h_2h_3$，因此 $\dfrac{n - a_0h_2 - b_0h_3}{h_2}$ 是正整数。$n = (\dfrac{n - a_0h_2 + b_0h_3}{h_2} + a_0)h_2 + b_0h_3$。

所以可以保证任何大于 $2h_2h_3$ 的数都能被写成 $ah_2 + bh_3, a \geq 0, b \geq 0$，当然也包括 $j - \ell$。

附加题：

由上面的证明过程可知，命题即证当 $(h_2, h_3) = 1$ 时，$\forall n > h_2h_3, \exists a \geq 0, b \geq 0, n = ah_2 + bh_3$。

考虑余数集合 $\{(n - kh_3)\ mod\ h_2 \mid 0 \leq k \leq h_2 - 1\}$。假设有 $1 \leq k_1 \neq k_2 \leq h_2$，使 $(n - k_1h_3)\ mod\ h_2 = (n - k_2h_3)\ mod\ h_2$，可得到 $h_2 \mid (k_1 - k_2)h_3$。但 $(h_2, h_3) = 1, 0 < |k_1 - k_2| < h_2$，会推出 h_2 不能整除 $(k_1 - k_2)h_3$，产生矛盾。因此余数各不相同。

所以余数集合中有 h_2 个元素，说明 0 属于这个集合，即 $\exists 0 \leq b \leq h_2$ 使 $n - bh_3$ 被 h_2 整除，且 $a = (n - bh_3)/h_2$ 是非负整数。这也就证明了满足条件的 a, b 是存在的。

(b) 既然 (a) 已经证明了数组 A 在调用了 InsertionSort(h_3) 和 InsertionSort(h_2) 后满足如下性质：对任意的 j 和 ℓ 满足 j> $\ell + 2h_2h_3$ 时，总会有 $A[j] > A[\ell]$。在执行 InsertionSort(h_1) 前，$A[j]$ 前最多有 $\dfrac{2h_2h_3}{h_1}$ 个大于它且模 h_1 同余的元素。观察 InsertionSort 的过程，它从前至后对每个元素进行插入，在插入 $A[j]$ 时后面的元素不会到前面来。也就是说插入 $A[j]$ 时，前面还是只有 $\dfrac{2h_2h_3}{h_1}$ 个大于它且模 h_1 同余的元素。那么每个元素在 InsertionSort(h_1) 只要比较至多 $\lfloor\dfrac{2h_2h_3}{h_1}\rfloor$ 次。一共 n 个元素，时间复杂度为 $O(n \cdot \dfrac{h_3h_2}{h_1})$。

问题 4 (10 分). 投掷一粒均匀的 k 面骰子，在骰子的面上分别标有数字 1 到 k，如果 X 是投出的点数，X 的期望大小是多少？(5 分) 如果想要反复投掷骰子来得到特定的点数 C，投掷次数的期望是多少 (5 分)？写出分析过程。

解答:

投出的点数 X 的期望大小 $\mathbb{E}\,X = \sum_{i=1}^{k} i \cdot \frac{1}{k} = \frac{k+1}{2}$。

反复投掷骰子时，第 n 次没有得到特定的点数 C 的概率为 $(1-\frac{1}{k})^n$，投掷次数 Y 的期望 $\mathbb{E}\,Y = \sum_{i=1}^{+\infty} i \cdot \frac{1}{k}(1-\frac{1}{k})^{n-1} = k$。

问题 5 (20 分)**.** 对下面两个程序，给出时间复杂度的最好估计。

(a) 分析下面的程序，给出算法的时间复杂度 (3 分)，并给出分析过程 (7 分):

提示： $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$, $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$

Algorithm 2 5.1

```
function A(n)
    t = 1
    for i = 1 to n do
        for j = i to n do
            for k = 1 to j do
                t = t+1
            end for
        end for
    end for
end function
```

(b) 分析下面的程序，给出算法的时间复杂度下界 (3 分)，并给出分析过程 (7 分):

解答:

(a) 总循环次数为

$$\sum_{i=1}^{n}\sum_{j=i}^{n}\sum_{k=1}^{j} 1 = \sum_{i=1}^{n}\sum_{j=i}^{n} j = \sum_{i=1}^{n}\sum_{m=1}^{n-i+1}(n+1-m) = \sum_{i=1}^{n}\frac{n^2+n-i^2+i}{2} = \frac{n^3}{3}+\frac{n^2}{2}+\frac{n}{6}$$

所以时间复杂度为 $\Theta(n^3)$

Algorithm 3 5.2

```
function B(n)
    if n == 1 then
        return 1
    else
        t = 0
        for i = 1 to n − 1 do
            t = t+B(i)
        end for
        return t
    end if
end function
```

(b) 设输入大小为 n 时，运行时间为 $T(n)$。
可知，$T(n) = \sum_{i=1}^{n-1} T(i)$，$T(1) = \Theta(1)$。
令 $S(n) = \sum_{i=1}^{n} T(i)$，有 $S(n) = 2S(n-1)$，$S(n) = \Theta(1)$，所以 $S(n) = 2^{n-1}$。
故 $T(n) = \Theta(2^n)$