算法导论习题选集

作业 3

节选自《算法导论》教材第三版
课程网站：https://algorithm.cuijiacai.com

Problem 1

（更多的递归式例子） 对下列每个递归式，给出 $T(n)$ 的渐近上界和下界。假定对足够小的 n，$T(n)$ 是常数。给出尽量紧确的界，并验证其正确性。

1. $T(n) = 4T(n/3) + n\log n$
2. $T(n) = 3T(n/3) + n/\log n$
3. $T(n) = 4T(n/2) + n^2\sqrt{n}$
4. $T(n) = 3T(n/3 - 1) + n/2$
5. $T(n) = 2T(n/2) + n/\log n$
6. $T(n) = T(n/2) + T(n/4) + T(n/8) + n$
7. $T(n) = T(n-1) + 1/n$
8. $T(n) = T(n-1) + \log n$
9. $T(n) = T(n-2) + 1/\log n$
10. $T(n) = \sqrt{n}T(\sqrt{n}) + n$

Problem 1 continued on next page…

(续页)

Problem 2

（佩波那契数） 本题讨论递归式 $F_0=0, F_1=1, F_i=F_{i-1}+F_{i-2}, i\geq 2$ 定义的佩波那契数的性质。我们将使用生成函数技术来求解佩波那契递归式。**生成函数**（又称为 **形式幂级数**）$\mathfrak{F}$ 定义为

$$\mathfrak{F}(z)=\sum_{i=0}^{\infty}F_iz^i=0+z+z^2+2z^3+3z^4+5z^5+8z^6+13z^7+21z^8+\ldots$$

其中 F_i 为第 i 个佩波那契数。

1. 证明：$\mathfrak{F}(z)=z+z\mathfrak{F}(z)+z^2\mathfrak{F}(z)$ 。

2. 证明：

$$\mathfrak{F}(z)=\frac{z}{1-z-z^2}=\frac{z}{(1-\phi z)(1-\hat{\phi}z)}=\frac{1}{\sqrt{5}}(\frac{1}{1-\phi z}-\frac{1}{1-\hat{\phi}z})$$

其中

$$\phi=\frac{1+\sqrt{5}}{2}=1.61803\cdots, \hat{\phi}=\frac{1-\sqrt{5}}{2}=-0.61803\cdots$$

3. 证明：

$$\mathfrak{F}(z)=\sum_{i=0}^{\infty}\frac{1}{\sqrt{5}}(\phi^i-\hat{\phi}^i)z^i$$

4. 利用上述结果证明：对 $i>0, F_i=\frac{\phi^i}{\sqrt{5}}$ ，结果舍入到最接近的整数。（提示：观察到 $|\hat{\phi}|<1$ 。）

(续页)

Problem 3

（芯片检测） 熊教授有 n 片可能完全一样的集成电路芯片，原理上可以用来相互检测。教授的测试夹具同时只能容纳两块芯片。当夹具装载上时，每块芯片都检测另一块，并报告它是好是坏。一块好的芯片总能准确报告另一块芯片的好坏，但教授不能信任坏芯片报告的结果。因此，4 种可能的测试结果如下：

芯片 A 的结果	芯片 B 的结果	结论
B 是好的	A 是好的	两片都是好的，或都是坏的
B 是好的	A 是坏的	至少一块是坏的
B 是坏的	A 是好的	至少一块是坏的
B 是坏的	A 是坏的	至少一块是坏的

1. 证明：如果超过 $n/2$ 块芯片是坏的，使用任何基于这种逐对检测操作的策略，教授都不能确定哪些芯片是好的。假定坏芯片可以合谋欺骗教授。
2. 考虑从 n 块芯片中寻找一块好芯片的问题，假定超过 $n/2$ 块芯片是好的。证明：进行 $\lfloor n/2 \rfloor$ 次逐对检测足以将问题规模减半。
3. 假定超过 $n/2$ 块芯片是好的，证明：可以用 $\Theta(n)$ 次逐对检测找出好的芯片。给出描述检测次数的递归式，并求解它。

Problem 3 continued on next page…

(续页)

Problem 4

（**Monge 阵列**）对一个 $m \times n$ 的实数阵列 A ，若对所有满足 $1 \le i < k \le m$ 和 $1 \le j < l \le n$ 的 i, j, k, l 有

$$A[i, j] + A[k, l] \le A[i, l] + A[k, j]$$

则称 A 是 **Monge 阵列**（Monge Array）。换句话说，无论何时选出 $Monge$ 阵列的两行和两列，对于交叉点上的 4 个元素，左上和右下两个元素之和总是小于等于左下和右上元素之和。例如，下面就是一个 Monge 阵列：

$$\begin{array}{rrrrr}
10 & 17 & 13 & 28 & 23 \\
17 & 22 & 16 & 29 & 23 \\
24 & 28 & 22 & 34 & 24 \\
11 & 13 & 6 & 17 & 7 \\
45 & 44 & 32 & 37 & 23 \\
36 & 33 & 19 & 21 & 6 \\
75 & 66 & 51 & 53 & 34
\end{array}$$

1. 证明：一个数组是 Monge 阵列当且仅当对所有 $i = 1, 2, ..., m-1$ 和 $j = 1, 2, ..., n-1$ ，有

$$A[i, j] + A[i+1, j+1] \le A[i, j+1] + A[i+1, j]$$

（提示：对于“当”的部分，分别对行和列使用归纳法。）

2. 下面数组不是 Monge 阵列。改变一个元素使其变成 Monge 阵列。（提示：利用第 1 问的结果。）

$$\begin{array}{rrrr}
37 & 23 & 22 & 32 \\
21 & 6 & 7 & 10 \\
53 & 34 & 30 & 31 \\
32 & 13 & 9 & 6 \\
43 & 21 & 15 & 8
\end{array}$$

3. 令 $f(i)$ 表示第 i 行的最左最小元素的列下标。证明：对任意 $m \times n$ 的 Monge 阵列，$f(1) \le f(2) \le ... \le f(m)$ 。

4. 下面是一个计算 $m \times n$ 的 Monge 阵列 A 每一行最左最小元素的分治算法的描述：提取 A 的偶数行构造其子矩阵 A' 。递归地确定 A' 每行的最左最小元素。然后计算 A 的奇数行的最左最小元素。

解释如何在 $O(m+n)$ 时间内计算 A 的奇数行的最左最小元素（在偶数行的最左最小元素已知的情况下）。

5. 给出第 4 问中描述的算法的运行时间的递归式。证明其解为 $O(m + n \log m)$ 。

Problem 4 continued on next page…

(续页)

(续页)