算法导论习题选集

作业 7-2

节选自《算法导论》教材第三版

课程网站：https://algorithm.cuijiacai.com

Problem 1

（平均排序）假设我们不是要完全排序一个数组，而只是要求数组中的元素在平均情况下是升序的。更准确地说，如果对所有的 $i = 1, 2, \cdots, n-k$ 有下式成立，我们就称一个包含 n 个元素的数组 A 为 **k-排序的**（k-sorted）：

$$\frac{\sum_{j=i}^{i+k-1} A[j]}{k} \leq \frac{\sum_{j=i+1}^{i+k} A[j]}{k}$$

1. 一个数组是 1 排序的，表示什么含义？

2. 给出对数字 $1, 2, \cdots, 10$ 的一个排列，它是 2 排序的，但不是完全有序的。

3. 证明：一个包含 n 个元素的数组是 k 排序的，当且仅当对所有的 $i = 1, 2, \cdots, n-k$ ，有 $A[i] \leq A[i+k]$ 。

4. 设计一个算法，它能在 $O(n \log(n/k))$ 时间内对一个包含 n 个元素的数组进行 k 排序。

当 k 是一个常数时，也可以给出 k 排序算法的下界。

5. 证明：我们可以在 $O(n \log k)$ 时间内对一个长度为 n 的 k 排序数组进行全排序。（提示：可以利用 [练习 5-2](/solution5/exe5-2/) 问题 6 的结果。）

6. 证明：当 k 是一个常数时，对包含 n 个元素的数组进行 k 排序需要 $\Omega(n \log n)$ 的时间。（提示：可以利用前面解决比较排序的下界的方法。）

Problem 1 continued on next page…

(续页)

Problem 2

（合并有序列表的下界）合并两个有序列表是我们经常会遇到的问题。作为 MERGE-SORT 的一个子过程，我们在第 1 讲 PPT 第 32 页已经遇到过这一问题。对这一问题，我们将证明在最坏情况下，合并两个都包含 n 个元素的有序列表所需的比较次数的下界是 $2n-1$ 。

首先，利用决策树来说明比较次数有一个下界 $2n - o(n)$ 。

1. 给定 $2n$ 个数，请算出共有多少种可能的方式将它们划分成两个有序的列表，其中每个列表都包含 n 个数。

2. 利用决策树和第 1 问的答案，证明：任何能够正确合并两个有序列表的算法都至少要进行 $2n - o(n)$ 次比较。

现在我们来给出一个更紧确的界 $2n - 1$ 。

3. 请说明：如果两个元素在有序序列中是连续的，且它们分别来自不同的列表，则它们必须进行比较。

4. 利用你对上一部分的回答，说明合并两个有序列表时的比较次数下界为 $2n - 1$ 。

(续页)

Problem 3

（0-1 排序引理和列排序）针对两个数组元素 $A[i]$ 和 $A[j]$ （$i < j$ ）的 **比较交换**操作的形式如下：

```
COMPARE-EXCHANGE(A, i, j)
1  if A[i] > A[j]
2       exchange A[i] with A[j]
```

经过比较交换操作之后，我们得到 $A[i] \le A[j]$ 。

遗忘比较交换算法是指算法只按照事先定义好的操作执行，即需要比较的位置下标必须事先确定好。虽然算法可能依靠待排序元素个数，但它不能依赖待排序元素的值，也不能依赖任何之前的比较交换操作的结果。例如，下面是一个基于遗忘比较交换算法的插入排序：

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2       for i = j − 1 downto 1
3            COMPARE-EXCHANGE(A, i, i + 1)
```

0-1 排序引理提供了有力的方法来证明一个遗忘比较交换算法可以产生正确的排序结果。该引理表明，如果一个遗忘比较交换算法能够对所有只包含 0 和 1 的输入序列排序，那么它也可以对包含任意值的输入序列排序。

你可以通过其逆否命题来证明 0-1 排序引理：如果一个遗忘比较交换算法不能对某个包含任意值的序列进行排序，那么它也不能对某个 0-1 序列进行排序。不妨假设一个遗忘比较交换算法 X 未能对数组 $A[1..n]$ 排序。设 $A[p]$ 是算法 X 未能将其放到正确位置的最小元素，而 $A[q]$ 是被算法 X 放在 $A[p]$ 原本应该在的位置上的元素。定义一个只包含 0 和 1 的数组 $B[1..n]$ 如下：

$$B[i] = \begin{cases} 0 & \text{if } A[i] \le A[p] \\ 1 & \text{if } A[i] > A[p] \end{cases}$$

1. 讨论：$A[q] > A[p]$ 时，从而 $B[p] = 0$ 且 $B[q] = 1$ 。

2. 为了完成 0-1 排序引理的证明，请先证明算法 X 不能对数组 B 正确地排序。

现在，需要用 0-1 排序引理来证明一个特别的排序算法的正确性。**列排序**算法是用于包含 n 个元素的矩形数组的排序。这一矩形数组有 r 行 s 列（因此 $n = rs$ ），满足下列三个限制条件：

- r 必须是偶数；
- s 必须是 r 的因子；
- $r \ge 2s^2$ ；

当列排序完成时，矩形数组是 **列优先有序**的：按照列从上到下，从左到右都是单调递增的。

如果不包括 n 的值的计算，列排序需要 8 步操作。所有奇数步都一样：对每一列单独进行排序。每一个偶数步是一个特定的排列。具体如下：

- 第 1 步：对每一列进行排序。
- 第 2 步：转置这个矩形数组，并重新规整化为 r 行 s 列的形式。也就是说，首先将最左边的一列放在前 r/s 行，然后将下一列放在第二个 r/s 行，依此类推。
- 第 3 步：对每一列进行排序。
- 第 4 步：执行第 2 步排列操作的逆操作。

- 第 5 步：对每一列进行排序。
- 第 6 步：将每一列的上半部分移动到同一列的下半部分位置，将每一列的下半部分移到下一列的上半部分，并将最左边一列的上半部分置为空。此时，最后一列的下半部分成为新的最右列的上半部分，新的最右列的下半部分为空。
- 第 7 步：对每一列进行排序。
- 第 8 步：执行第 6 步排列操作的逆操作。

下图展示了一个在 $r = 6$ 和 $s = 3$ 情况下的列排序步骤（即使这个例子违背了 $r \geq 2s^2$ 的条件，列排序仍然有效）。

10	14	5
8	7	17
12	1	6
16	9	11
4	15	2
18	3	13

(a)

4	1	2
8	3	5
10	7	6
12	9	11
16	14	13
18	15	17

(b)

4	8	10
12	16	18
1	3	7
9	14	15
2	5	6
11	13	17

(c)

1	3	6
2	5	7
4	8	10
9	13	15
11	14	17
12	16	18

(d)

1	4	11
3	8	14
6	10	17
2	9	12
5	13	16
7	15	18

(e)

1	4	11
2	8	12
3	9	14
5	10	16
6	13	17
7	15	18

(f)

5	10	16
6	13	17
7	15	18
1	4	11
2	8	12
3	9	14

(g)

4	10	16
5	11	17
6	12	18
1	7	13
2	8	14
3	9	15

(h)

1	7	13
2	8	14
3	9	15
4	10	16
5	11	17
6	12	18

(i)

3. 讨论：即使不知道奇数步采用了什么排序算法，我们也可以把列排序看做一种遗忘比较算法。

虽然似乎很难让人相信列排序也能实现排序，但是你可以利用 0-1 排序引理来证明这一点。因为列排序可以看做是一种遗忘比较交换算法，所以我们可以使用 0-1 排序引理。下面一些定义有助于你使用这一引理。如果数组中某个区域只包含全 0 或者全 1 ，我们定义

这个区域是 **干净的**。否则，如果这个区域包含的是 0 和 1 的混合，则称这个区域是 **脏的**。这里，假设输入数据只包含 0 和 1 ，且输入数据能够被转换为 r 行 s 列。

4. 证明：经过第 1 到 3 步，数组由三部分组成：顶部一些由全 0 组成的干净行，底部一些由全 1 组成的干净行，以及中间最多 s 行脏的行。

5. 证明：经过第 4 步之后，如果按照列优先原则读取数组，先读到的是全 0 的干净区域，最后是全 1 的干净区域，中间是由最多 s^2 个元素组成的脏的区域。

6. 证明：第 5 到 8 步产生一个全排序的 0-1 输出，并得到结论：列排序可以正确地对任意输入值排序。

(续页)

(续页)