算法导论习题选集

作业 6-2

节选自《算法导论》教材第三版

课程网站：https://algorithm.cuijiacai.com

Problem 1

（快速排序的栈深度）第 6 讲 PPT 第 5 页中的 QUICKSORT 算法包含了两个对其自身的递归调用。在调用 PARTITION 后，QUICKSORT 分别递归调用了左边的子数组和右边的子数组。QUICKSORT 中的第二个递归调用并不是必须的。我们可以用一个循环控制结构来代替它。这一技术称为 **尾递归**（tail recursion），好的编译器都提供这一功能。考虑下面这个版本的快速排序，它模拟了尾递归情况：

```
TAIL-RECURSIVE-QUICKSORT(A, p, r)
while p < r
    // Partition and sort left subarray.
    q = PARTITION(A, p, r)
    TAIL-RECURSIVE-QUICKSORT(A, p, q − 1)
    p = q + 1
```

1. 证明：TAIL-RECURSIVE-QUICKSORT($A, 1, A.length$) 能正确地对数组 A 进行排序。

编译器通常使用 **栈**来存储递归执行过程中的相关信息，包括每一次递归调用的参数等。最新调用的信息存在栈的顶部，而第一次调用的信息存在栈的底部。当一个过程被调用时，其相关信息被 **压入**栈中；当它结束时，其信息则被 **弹出**。因为我们假设数组参数是用指针来表示的，所以每次过程调用只需要 $O(1)$ 的栈空间。**栈深度**是在一次计算中会用到的栈空间的最大值。

2. 请描述一种场景，使得针对一个包含 n 个元素数组的 TAIL-RECURSIVE-QUICKSORT 的栈深度是 $\Theta(n)$ 。

3. 修改 TAIL-RECURSIVE-QUICKSORT 的代码，使其最坏情况下的栈深度是 $\Theta(\log n)$ ，并且能够保持 $O(n \log n)$ 的期望时间复杂度。

Problem 1 continued on next page…

(续页)

(续页)

Problem 2

(三数取中划分) 一种改进 RANDOMIZED-QUICKSORT 的方法是在划分时，要从子数组中更细致地选择作为主元的元素（而不是简单地随机选择）。常用的做法是三数取中法：从子数组中随机挑选出三个元素，取其中位数作为主元（见练习 6 问题 5）。对于这个问题的分析，我们不妨假设数组 $A[1..n]$ 的元素是互异的且有 $n \geq 3$ 。我们用 $A'[1..n]$ 来表示已排好序的数组。用三数取中法选择主元 x ，并定义 $p_i = Pr\{x = A'[i]\}$ 。

1. 对于 $i = 2, 3, \cdots, n-1$ ，请给出以 n 和 i 表示的 p_i 的准确表达式（注意 $p_1 = p_n = 0$）。

2. 与平凡实现相比，在这种实现中，选择 $x = A'[\lfloor (n+1)/2 \rfloor]$ （即 $A[1..n]$ 的中位数）的值作为主元的概率增加了多少？假设 $n \to \infty$ ，请给出两种实现下主元去到中位数的概率比值的极限值。

3. 如果我们定义一个“好”划分意味着主元选择 $x = A'[i]$ ，其中 $n/3 \leq i \leq 2n/3$ 。与平凡实现相比，这种实现中得到一个好划分的概率增加了多少？（提示：用积分来近似累加和。）

4. 证明：对快速排序而言，三数取中法只影响其时间复杂度 $\Omega(n \log n)$ 的常数项因子。

Problem 2 continued on next page…

(续页)

Problem 2 continued on next page…

(续页)

Problem 3

（对区间的模糊排序）考虑这样的一种排序问题：我们无法准确知道待排序的数字是什么。但对于每一个数，我们知道它属于实数轴上的某个区间。也就是说，我们得到了 n 个形如 $[a_i, b_i]$ 的闭区间，其中 $a_i \le b_i$ 。我们的目标是实现这些区间的**模糊排序**，即对 $j = 1, 2, \cdots, n$ ，生成一个区间的排列 $\langle i_1, i_2, \cdots, i_n \rangle$ ，且存在 $c_j \in [a_{i_j}, b_{i_j}]$ ，满足 $c_1 \le c_2 \le \cdots \le c_n$ 。

1. 为 n 个区间的模糊排序设计一个随机算法。你的算法应该具有算法的一般结构，它可以对左端点（即 a_i 的值）进行快速排序，同时它也能利用区间的重叠性质来改善时间性能。（当区间重叠越来越多的时候，区间的模糊排序问题会变得越来越容易。你的算法应能充分利用这一重叠性质。）

2. 证明：在一般情况下，你的算法的期望运行时间为 $\Theta(n \log n)$ 。但是，当所有的区间都有重叠的时候，算法的期望运行时间为 $\Theta(n)$ （也就是说，存在一个值 x ，对所有的 i ，都有 $x \in [a_i, b_i]$ 。）你的算法不必显式地检查这种情况，而是随着重叠情况的增加，算法能自然地提高。

Problem 3 continued on next page…

(续页)

Problem 3 continued on next page…

(续页)