2.7

(1)

用快速排序排序数组 A，若 A 中存在主元素，则主元素必为 A 数组中位数 (n 为奇数) 或者两个最邻近中位数的平均值，令该值为 y。

然后遍历数组 A，查看数组 A 中是否存在大于 $\lfloor \frac{n}{2} \rfloor$ 次的 y 值。

时间复杂度为 $T(n) = O(n \log n) + O(n) = O(n \log n)$

(2)

同 (1) 找中位数的算法可以达到 $O(n)$

时间复杂度为 $T(n) = O(n) + O(n) = O(n)$

(3)

芯片测试算法

1. 如果 n 为偶数，将数组 A 两两分组。如果 n 为奇数，取出某个数 a 后，在当前数组上检验该元素是否为主元素，如果是，算法结束，如果不是再将，舍弃该数，再将数组 A 两两分组。

2. 将每组数组的两个数组比较, 若两个数相同，则随机将该组某个数放入数组 B，否则则将其这两个数舍弃。

3、将数组 B 复制给数组 A, 如果数组 A 元素的个数小于等于 1，则将剩下的唯一元素与原数组比较，看该元素是否为主元素。否则则继续转入 1.

该算法下有两个基本性质

1. 若主元素存在，每一次迭代数组之后，主元素个数占总元素个数的比例仍然大雨 $\frac{1}{2}$

2. 每次迭代之后，如果算法需要继续，规模一定小于等于 $\lfloor \frac{n}{2} \rfloor$ 迭代之前的数组规模为 n

证明:

性质 2 显然成立。下面证明性质 1。

设数组 A 两两分组之后，情况如下:

(1) 两个元素都为主元素，数量为 a

(2) 两个元素一个为主元素，一个不为主元素，数量为 b

(3) 两个元素均不为主元素，数量为 c

由主元素性质可以知道 $2a+b>2c+b$，得到 $a>c$，迭代一次之后，主元素的个数为 a, 总元素个数的最大值为 $a+c$, 所以性质 2 成立。

算法时间复杂度：

考虑最坏情况，每次迭代之后，子问题的规模最大为 $\frac{n}{2}$, 并且每次迭代都要扫描当前数组

$$W(n) = W(\frac{n}{2}) + O(n)$$

由主定理可得 $W(n) = O(n)$

最后还要扫描一遍原数组 $T(n) = W(n) + O(n) = O(n)$

总的时间复杂度为 $O(n)$

1 2.12

1. 将集合 B 排序，生成数组 L 2. 对 A 中每一个元素 a，在 L 中用 BinarySearch 查找是否在 L 中存在和 a 一样的元素。

算法时间复杂度：集合 B 排序时间复杂度 $O(m\log m)$，查找时间复杂度 $O(n\log m)$ 总的时间复杂度 $T(n) = O(m\log m) + O(n\log m) = O(n\log m) = O(n\log\log n)$

2 2.15

(1)

算法 A

$T(n) = \sum_1^i n-i = \frac{(n-1+n-i)*i}{2} = \frac{(2n-1-i)*i}{2} = O(n^{\frac{3}{2}})$

算法 B

$T(n) = O(n\log n)$

(2)

1. 令 $d = n-i+1$

2. 采用 Select 算法，在数组 S 中采用，Select(S,k)

3. 此时 S[n-i,n-1] 为数组 S 中最大的 i 个数（并无有序）

4、将 S[n-i，n-1] 采用快速排序，然后倒序输出。

时间复杂度: $T(n) = O(n) + O(i\log i) = O(n)$

2.18

1. 遍历 n 个点，先算出每个点到原点 $(0,0)$ 的距离，得到一个数组 A
2. 用 Select 算法，运行 Select(A,$\lfloor\sqrt{n}\rfloor$)
3. 此时 A[0，$\lfloor\sqrt{n}\rfloor - 1$] 为距离原点 $(0,0)$ 最近的 $\lfloor\sqrt{n}\rfloor$ 个点。
4. 对该区间进行快排，然后倒序输出。

时间复杂度: $T(n) = O(n) + O(\lfloor\sqrt{n}\rfloor \log \lfloor\sqrt{n}\rfloor) = O(n)$

for $i \leftarrow 1$ to n

$d_i \leftarrow \sqrt{x_i^2 + y_i^2}$, 同时将每个点的下标与距离组成一个结构体, 得到数组 A

$k \leftarrow \lfloor\sqrt{n}\rfloor$

Select([$d_0, d_1, \cdots, d_{n-1}$],k)

Quicksort(A[0,$\lfloor\sqrt{n}\rfloor$-1])

for $i \leftarrow \lfloor\sqrt{n}\rfloor - 1$ to 0

return A[i]

2.23

用 Select 算法找第 $\lceil \frac{n}{4} \rceil$ 个小的数 a 和第 $\lfloor \frac{3n}{4} \rfloor$ 个小的数 b

if a == b:

return None

else:

将数组分成 A,B,C,D,E, 其中 A 中元素小于 a,B 中元素等于 a,C 中元素大于 a 且小于 b,D 中元素等于 b,E 中元素大于 b, 若 C 非空，C 中任意一个数都可以作为近似中值。

如果 C 为空，再特定判断 a，b 是否满足近似中值的条件。

时间复杂度

$T(n) = O(n) + O(n) = O(n)$

2.27

对于 n 个商店，其坐标分别为 $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$，我们需要找到一个 (x, y)

使得 $min|x - x_1| + |y - y_1| + |x - x_2| + |y - y_2| + \cdots + |x - x_n| + |y - y_n|$

上式可化简为 $min|x - x_1| + \cdots| + |x - x_n|$+$min|y - y_1| + \cdots + |y - y_n|$

对于$min|x - x_1| + \cdots + |x - x_n|$, 我们只需找到序列 $[x_1, x_2, \cdots, x_n]$ 的中位数即可

算法复杂度

$T(n) = O(n)$