寻找发帖“水王”

★

Tango 是微软亚洲研究院的一个试验项目。研究院的员工和实习生们都很喜欢在 Tango 上面交流灌水。传说，Tango 有一大“水王”，他不但喜欢发贴，还会回复其他 ID 发的每个帖子。坊间风闻该“水王”发帖数目超过了帖子总数的一半。如果你有一个当前论坛上所有帖子（包括回帖）的列表，其中帖子作者的 ID 也在表中，你能快速找出这个传说中的 Tango 水王吗？

分析与解法

首先想到的是一个最直接的方法，我们可以对所有 ID 进行排序。然后再扫描一遍排好序的 ID 列表，统计各个 ID 出现的次数。如果某个 ID 出现的次数超过总数的一半，那么就输出这个 ID。这个算法的时间复杂度为 $O(N * \log_2 N + N)$。

如果 ID 列表已经是有序的，还需要扫描一遍整个列表来统计各个 ID 出现的次数吗？

如果一个 ID 出现的次数超过总数 N 的一半。那么，无论水王的 ID 是什么，这个有序的 ID 列表中的第 $N/2$ 项（从 0 开始编号）一定会是这个 ID（读者可以试着证明一下）。省去重新扫描一遍列表，可以节省一点算法耗费的时间。如果能够迅速定位到列表的某一项（比如使用数组来存储列表），除去排序的时间复杂度，后处理需要的时间为 $O(1)$。

但上面两种方法都需要先对 ID 列表进行排序，时间复杂度方面没有本质的改进。能否避免排序呢？

如果每次删除两个不同的 ID（不管是否包含“水王”的 ID），那么，在剩下的 ID 列表中，“水王”ID 出现的次数仍然超过总数的一半。看到这一点之后，就可以通过不断重复这个过程，把 ID 列表中的 ID 总数降低（转化为更小的问题），从而得到问题的答案。新的思路，避免了排序这个耗时的步骤，总的时间复杂度只有 $O(N)$，且只需要常数的额外内存。伪代码如下：

代码清单 2-8

```
Type Find(Type* ID, int N)
{
    Type candidate;
    int nTimes, i;
    for(i = nTimes = 0; i < N; i++)
    {
        if(nTimes == 0)
        {
            candidate = ID[i], nTimes = 1;
        }
        else
        {
            if(candidate == ID[i])
                nTimes++;
            else
                nTimes--;
        }
    }
    return candidate;
}
```

在这个题目中，有一个计算机科学中很普遍的思想，就是如何把一个问题转化为规模较小的若干个问题。分治、递推和贪心等都是基于这样的思路。在转化过程中，小的问题跟原问题本质上一致。这样，我们可以通过同样的方式将小问题转化为更小的问题。因此，转化过程是很重要的。像上面这个题目，我们保证了问题的解在小问题中仍然具有与原问题相同

的性质：水王的 ID 在 ID 列表中的次数超过一半。转化本身计算的效率越高，转化之后问题规模缩小得越快，则整体算法的时间复杂度越低。

扩展问题

随着 Tango 的发展，管理员发现，“超级水王”没有了。统计结果表明，有 3 个发帖很多的 ID，他们的发帖数目都超过了帖子总数目 *N* 的 1/4。你能从发帖 ID 列表中快速找出他们的 ID 吗？