论对算法的选择

上海市复旦附中　　张云亮

一、引言

- 计算机竞赛是一项对选手计算机知识、编程能力的综合测试，在平时的训练之中，我们一般都会精益求精，设法想出最好的算法，但是在竞赛的时候，时间和心理状态都是不一样的，如何在竞赛中编出能得分尽量多的程序，对各种算法的灵活运用是很重要的。

二、算法的复杂度

- 我在这里所说的复杂度，包括编程复杂度，时间复杂度，空间复杂度，因为在竞赛几个小时的时间里，编程复杂度就不容忽视，知道算法但是程序没完成，这是最不理想的情况，对源程序的最基本的质量要求是正确性和可靠性，只要保证在这两点的情况下，各种算法都是好算法。所以在这三个复杂度都考虑的情况下，找到最适宜的算法是必要的。

三、对于一些题目的思路

- 在我们做题的时候，我们先想出的算法不一定是最好的算法，但不是说明它是不好的算法，所以，在我们想出一种算法的时候，要考虑一下它的可行性，如果可行的话，不如自己先把该算法编一编，可以锻炼自己编那些非标准算法的能力，这样对自己面对新颖题型的时候是会有好处的。

例 1

[问题描述]

- 输入：

  一个正整数 n ， 以及整数数列 $A_1, A_2, A_3 \ldots\ldots, A_n$ 。

  一个正整数 m ， 以及整数数列 $B_1, B_2, B_3 \ldots\ldots, B_m$ 。

  其中

（ $1<=n<=10^6$, $1<=A_i<=10^6$ ， $1<=m<=1000$, $1<=B_i<=10^6$ ）

- 输出：

  一共 m 行，每行一个整数，第 i 行所输出的数表示数列 {A} 中小于等于 B_i 的数的数目 。

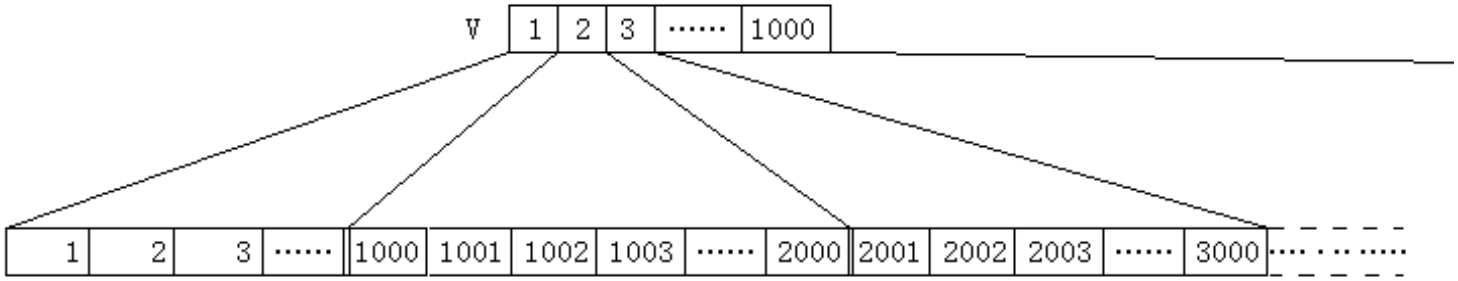
例 1

[算法分析]

- 首先，我们便注意到了 n 与 m 的范围的差距。
- 本算法运用了两个数组
  - L[1..1000000]

    (记录数列 {A} 中取值为 k 的数有 L[k] 个)
  - V[1..1000]

    (记录数列 {A} 中取值在 [(k-1)*1000+1 ， k*1000] 的数有 V[k] 个)



例 1

[算法分析]

程序流程：

- 预处理：对数组于 L， V 进行清零。

- 读入 n，依次读入 A_i。对于每个 A_i，
  - 得出 k （k 为整数），使 A_i 在区间 [(k-1)*1000+1， k*1000] 内
  - L[A_i]++,V[k]++

- 读入 m，依次读入 B_i。对每个 B_i，
  - 得出 k （k 为整数），使 B_i 在区间 [(k-1)*1000+1， k*1000] 内
  - 设 {A} 中小于 B_i 的数有 S 个，易知
  - $S = \sum_{i=1}^{i<k} V[i] + \sum_{i=(k-1)*1000+1}^{i<=B[i]} L[i]$
  - 输出 S

例 1

[复杂度分析]

- 本算法
  - 计算每个 S 的值最多需要运算操作 1000+1000=2000 次。
  - 每次将一个 A_i 记录的操作需要 2 次
- 线段树
  - 计算每个 S 的值最多需要运算操作 $(n+m)*\log_2 G$
  - 每次将一个 A_i 记录的操作需要 $(n+m)*\log_2 G$
- 两种算法的复杂度比较
  - 其中线段数对于计算 S 的值与记录一个 A_i 都只要在 $(n+m)*\log_2 G$ 的运算次数下完成
  - 线段树　　本算法　　（最坏情况）
  - $(n+m)*\log_2 G$　　$n*2+m*2000$　　（其中 $G=10^6$）

例 2 、营业额统计

[问题描述]

公司的账本上记录了公司成立以来每天的营业额。分析营业情况是项相当复杂的工作，营业额会出现一定的波动，当然一定的波动是能够接受的，但营业额突变得很高或是很低，这就证明公司此时的经营状况出现了问题。经济管理学上定义了一种最小波动值来衡量这种情况：

$$\text{该天的最小波动值} = \min\{\ |\text{该天以前某一天的营业额} - \text{该天营业额}|\ \}$$

当最小波动值越大时，就说明营业情况越不稳定。

而分析整个公司的从成立到现在营业情况是否稳定，只需要把每一天的最小波动值加起来就可以了。本程序任务就是编写一个程序来计算这一个值。第一天的最小波动值为第一天的营业额。

例 2 、营业额统计

[问题描述]

输入和输出：

- 输入：
  - 输入文件第一行为正整数，表示该公司从成立一直到现在的天数，接下来的 n 行每行有一个正整数 a_i $(|a_i| \le 1000000000)$，表示第 i 天公司的营业额。
- 输出：
  - 输出文件仅有一个正整数，即 $\sum$ 每一天的最小波动值，它小于 2^{31} 。
- 输入输出样例：

输入文件	输出文件
6 5 1 2 5 4 6	12

例 2 、营业额统计

[算法分析]

- 本题题意明了，关键是读入一个数，找到前面已经输入的与该数数相差最小的数。

- 本算法运用了两个数组

  - L[1..32767]
  - V[1..327]
  - 其中的定义与例 1 中的一样，只不过是对数的下标进行处理。

例2、营业额统计

[算法分析]

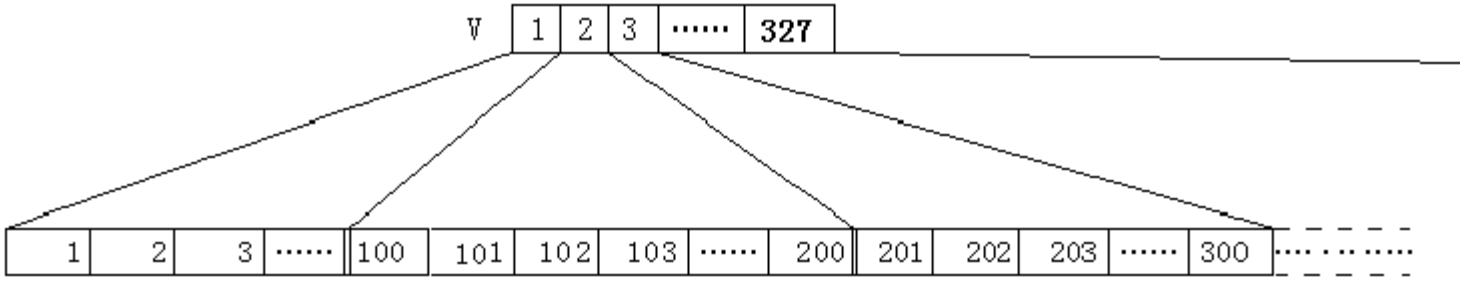
- 预处理：
  - 把全部数据读入，将 {A} 从小到大排序，这样得到一个序列 $B_1,B_2,\ldots\ldots,B_n$ 。
  - 累加器 S 赋值为 0
- 主过程：
  - 读入将要处理的数据，对每天的营业额进行处理，求出该天的最小波动值。
- 结尾阶段：输出 S

例 2 、营业额统计

主过程

- 读入当天的营业额 P 。
- 用二分查找法在数列 {B} 中找到 B_q=P
- 然后利用数组 L ， V ， 设法找到已经储存的下标之中小于等于 q 中最大的下标 ga ， 与大于等于 q 中最小的下标 gb ，

  且 L[ga]>0 ， L[gb]>0 ， 然后通过 ga 与 gb 计算出当前的最小波动值。



例 2 、营业额统计

[复杂度分析]

- 预处理的排序时间复杂度 $Nlog_2N$
- 对于计算每天的最小波动值。大约需要 32767*500 的运算次数。
- 总计时间复杂度为 $O(n^{1.5})$ 左右，可以在规定时间之内完成。

- 而平衡树的时间复杂度为 $O(nlog_2n)$ ，比较之下显然平衡树的时间复杂度小于本算法的时间复杂度。但本算法的测试与修改却比平衡树容易的多，因为它是分多步进行，可以分步进行测试而且每步的要求技术都不高。

四、总结

- 每个算法都是有它自己的优势，每种算法的效果在不同的场合是不一样的，在编程的过程中，在平常的练习之中，我们就需要对一道题目进行多方面的思考，不能抱有知道算法就完事的一种心态，对一到题目，要多考虑新的算法，这样便能发现每种算法使用的场合，这样面对形式未见过的题型的时候，就会有更多的思路。