

## 中等硬度解题报告

**[摘要]**中等硬度是 IOI2000 第一试的最后一道题目。这道题主要考察选手的创造性，自创算法正确高效的解决问的能力。本文主要讲述我做这到题目的过程和方法。

**[关键字]**二分法 随机数

**[问题描述]**见附件

**[问题分析]**

算法 1-1: 由于每次比较可以得出最大的数和最小的数（虽然不知道那个数最大的），所以可以先求出 1, 2, 3 号中的最大与最小者，再用它们与 4 号比较得出 1~4 号中的最大最小者，再用这两个数与 5 号比……。以此类推，可求出 1~n 中的最大与最小者，显然它们不是中等硬度物体，所以将它们去掉，再用上述方法求出剩下 n-2 中的最大与最小者，再去掉，……，最后剩下的一个数就是中等硬度的物体编号。

这个算法比较的复杂度为  $O(n^2)$ ，不满足 1449 个物体用 7777 次比较出来的限制。究其原因是因为每次比较利用的信息不够。一次比较三个数，可以知道这三个数的顺序关系（虽然不知道是递增还是递减），若已知两个数的顺序关系，再与第三个数比较，则可知第三个数在前两个数的顺序关系下的位置。算法 1-1 中正是没有利用这个信息，导致复杂度高。所以利用这个排序的观点，得出算法 2-1。

算法 2-1: 已知前 m 个物体硬度的顺序关系。将下一个物体用二分法插入到适当的位置，得出前 m+1 个物体硬度的顺序关系。以此类推得出 n 个物体的顺序关系，从而知道中等硬度者。参看实例：

Label	1	2	3	4	5
Strength	2	5	4	3	1

插入物体编号	已知顺序	比较	返回结果	得出顺序
		1 2 3	3	1 3 2
4	1 3 2	1 3 4	4	1 ( ) 3 2
5	1 4 3 2	4 3 5	4	(1) 4 3 2
5	1 4 3 2	1 4 5	1	( ) 1 4 3 2
	5 1 4 3 2			

注：（）表示改物体可能插入的区域。第二行得出结论 4 在 1 和 3 中间。第三行得出结论 5 的位置在 4 的左侧，所以还要比较一次以确定 5 与 1 的位置关系。

这个算法采用二分插入法。二分插入法复杂度为  $O(\log_2 m)$ 。要插入 n-3 个数。所以总的复杂度比  $n \cdot \log_2 n$  要小一些。实践证明，当 n=1449 时，平均用 11000 多次比较。（ $1449 \times \log_2 1449 \approx 15216$ ）究其原因是因为这个算法将所有物体的硬度都排了序，其中有些是一定不只中等硬度的，对于它们的排序浪费了比较次数。所以在算法 2-1 的基础上加上剪枝，得出算法 2-2。

算法 2-2: 设  $2m+1=n$ 。则可知对于一个已全部排好顺序的序列，中等硬度物体左边有 n 个物体，右边也有 n 个物体。也就是说若已知一个物体所在序列中位置的左边(右边)有 n 个以上物体时，则它一定不是所求。所以不用排出它的具体位置，将其插在最右边（左边）即可。利用这一原理，用先将前 m+1 个物体排序，因为这 m+1 个物体都有可能是所求。当第 m+2 号物体插入后，最两端的物体肯定不是所求。同理当现在已经有 m+3 个物体排好序，则这个序列两端的两个物体一定不是所求。也就是说第 k 个物体只有插在位置 (k-m-1) 到位置 (m+1) 这个子序列中，才有可能成为中等硬度者。所以在二分法插入中，若当前插入区域已不在上述区域中，则它的具体位置对结果没有影响，所以直接插入到最左端或最右端。举个简单的实例，当前已将 n-1 个物体排好序，在这 n-1 个物体中只有最中间的两个物

体才有可能为中等硬度。我们称其为 A,B (A 在 B 左侧)。按照上述方法, 第 n 个物体 C 只需与 A、B 比较一次。若 A 在中间, 则 C 的插入区域在 A 的左边, 已不再可能区域中, 于是将 C 插入到最左端, 最终结果是 A。若 B 在中间, 同理将 C 插入到最右端, 结果为 C。若 C 在中间, 则插入到 AB 之间, 结果为 C。而不需要象算法 2-1 那样, 比较若干次, 将 C 插入到正确的位置。第二个剪枝在第一个剪枝的基础上, 通过实践发现有相当一部分物体是通过上述剪枝, 插入到两端。而当判断出它不属于可能区域时, 已经做了 2~5 次判断。所以直接先用 1~2 次比较, 判断出插入物体是否在可能区域。若是, 则在用二分法插入; 若否, 则直接插入到两端。这样可以提高一些效率。

这个改进可使 1449 个物体的比较次数平均为 8200, 只距 7777 的上限一步之遥。但改进幅度很小, 原因是这种排序的方法对前 725 个物体的排序无法剪枝, 要用去 5000 次左右的比较。所以要在规定次数内解出此题, 只能改变算法, 进入 3-X 算法系列。

算法 3-1: 虽然前两系列算法没有成功, 但我总结了一下, 有以下 2 点经验值得借鉴。1.要有位置概念。虽然不知大小, 但要有顺序。2.要有区域概念。先判断出可能区域, 再逐步细化。(算法 2-2 就是细化得太早) 所以算法是: 用 a[1]和 a[2] (a[i]表示可能区域中第 i 个物体的号码是 a[i]。不妨设 a[1]在 a[2]左边) 依次与 a[3]~a[n]比较, 将所有物体分为三类: 一类在 a[1]左边 (比 a[1]小或大), 一类在 a[1]和 a[2]之间 (硬度介于两者之间), 一类在 a[2]右边 (比 a[2]大或小)。统计个数可知中等硬度物体是 a[1]或 a[2]或三类之一。若是三类之一, 再对这一类物体采用相似的方法分解, 直到求出结果。之所以称相似的方法, 是因为有两点不同: 1.不能假设 a[1]在 a[2]左边 (这时 a[1],a[2]的值已经改变, a[1]~a[u]分别记录这一类物体的编号)。a[1]与 a[2]的位置关系, 要引入上一层分类中的一个基数 (上一层的 a[1]或 a[2]), 做一次比较, 使得 a[1]a[2]的顺序与以前保持一致。2.中等硬度者位置已不是 m+1。具体位置, 要根据上一层的中等硬度物体位置及分类结果得出。这个位置用一个参量在递归中传递。最后做一点改进, 为防止特殊输入数据。每次所选的基数不再是 a[1]和 a[2], 而是 a[x]、a[y], x、y 为随机数。

这个算法可以完全解决此问题。  
[总结]三个算法的效率比较表, 结果为运行 10 次随即产生的输入数据的平均值。:

	2-1	2-2	3-1
N=1449 所用比较次数	11176	8189	3442

通过这道题, 我认识到改进一个程序要充分了解到该程序的不足之处, 抓住主要矛盾。并且当看到算法没有太大的改进地步时, 也要善于总结经验与其成功之处, 为新算法打基础。

[附录]

中等硬度结题报告.doc	-----本文
median1.pas	-----算法 2-1 程序
median2.pas	-----算法 2-2 程序
median3.pas	-----算法 3-1 程序
medmake.pas	-----输入数据生成程序
device.pas	-----交互单元