5. 贪心算法

张桄玮(gwzhang@cug.edu.cn)

郑州一中(Legacy)

2024-08-01

类似背包的问题

一个特殊的凑硬币问题

问题: 某人带着 3 种面值的硬币, 分别是 1, 2, 5 元. 数量不限. 需要支付M元, 问怎样支付需要的硬币最少?

- 第一节讲过任意的面值的递归做法

策略: 首先拿出面额最大的数凑(5 元); 然后是面额中等的(3 元); 最后是面额最小的(1 元).

为什么? 大的可以用小的 “表示”

- $1 + 1 = 2$;
- $2 + 2 + 1 = 5$.

如果要用小的, 肯定能用大的替换出来. (交换论证)

带来的潜在的麻烦: 不能推广

一个特殊的凑硬币问题

- 不能得到最优解的情况: $1, 2, 4, 5, 6$; 凑出 9 元;
  - $9 = 5 + 2 + 4 = 6 + 1$;
- 不能得到解的: 上述算法说凑不出! 但是$9 = 5 + 2 + 2$

思考题: 什么样的硬币组合能够用这个算法求解?(有点难)

- 硬币的面额是$c^0, c^1, c^2, ..., c^k$ 贪心算法总是可以找到最优解
- 还有别的情况吗? (Hard)

背包装载问题

问题：给出n个物品，第i个物品的质量为w_i，选择尽可能多的物品，使得总质量不超过C.

- 观察：装重的物品没有装轻的物品划算.
  - 如何形式化地考虑这个问题？
- 策略：把物品从小到大排序，依次选择每个物体，直到装不下为止.
- 如何证明这个内容是正确的？

证明：假设不是按照这种方法选取的，而是用的别的方法.

- 找到不是按照这个排序的第一个东西
- 把它换为新的东西就得到了更优的解答！

■

部分背包问题

问题：有n个物体，第i个的物体重量为x_i，价值为v_i，在总重量不超过C的情形下让总价值尽可能高.

- 每一个物品可以拿走一部分
- 价值和重量按照比例计算

Attempts:

- 拿重的？不一定(占满空间)；
- 拿轻的？不一定(价值太小)；
- 考虑价值 ÷ 质量？(看起来不错)

策略：按照价值 ÷ 质量排序，从大往小选择

证明：

部分背包问题

- 假设物品集合$S = \{W_1, W_2, ..., W_n\}$已经按价值 ÷ 质量从小到大排好序了
- 全局最优解是: $S(i) = \left\{W_{i_1}, W_{i_2}, ..., W_{i_n}\right\}$. W_{i_1}, $W_{i_2}, ..., W_{i_n}$ 是有序的. 对于贪心选择而言, 总是会优先选择 W_n 的物品, 当W_n没有后, 再选择W_{n-1}
  - 如果$W_{i_n} = W_n$ 问题已经得证. 因为, 我们的最优解$S(i)$中, 已经包含了贪心选择. 只要继续归纳下去, $W_{i_{n-1}}$ 就是 W_{n-1}
  - 假设存在一个最优解，在这个解中我们没有尽可能多地拿取物品 i，并且假设我们的背包已经满了（如果没有满，只需添加更多的物品 i 即可）。由于物品 i 具有最高的价值与重量比，那么在我们的背包中一定存在一个物品 j，使得 $\frac{v_j}{w_j} < \frac{v_i}{w_i}$。我们可以从背包中取出重量为 x 的物品 j，并且可以将重量为 x 的物品 i 放入

部分背包问题

背包中（因为我们取出了 x 重量，同时放入 x 重量，因此仍然在容量范围内）。背包的价值变化为 $x\frac{v_i}{w_i} - x\frac{v_j}{w_j} = x\left(\frac{v_i}{w_i} - \frac{v_j}{w_j}\right) > 0$，因为 $\frac{v_j}{w_j} < \frac{v_i}{w_i}$。

因此，我们得出矛盾，因为我们最初假设的“所谓”最优解实际上可以通过取出一些物品 j 并添加更多的物品 i 来改进。因此，这不是最优的。 ■

另一个问题: 你为什么能够分阶段考虑问题? (i.e. 为什么最优的解能够组合出最优的解?)

- 不那么优的解肯定无法生成一个最优解

问题：那么如果限定了每一个物品要么全拿走, 要么全不拿, 有没有贪心算法?

部分背包问题

- 没有了, 似乎任何策略都没有办法完成
- 必须 exhaustive 地遍历所有情况

乘船问题

问题: 有n个人, 第i个人的重量为w_i. 每艘船最大载重量为C. 每船最多只能承载 2 个人. 用最少的船装载所有人.

考虑最坏的情况

- 最轻的两个人也坐不了一艘船 $\rightarrow$ 每个人占据一条船
- 如果能坐一艘船, 让两个人一起坐这艘?
  - 仔细考虑: 两个人一艘船, 可能还有 “容量的浪费”
  - 最小化这个?

从小到大, 体重最轻的人i先上船, 然后再剩余可以坐船的人中选一个最大的人j.

声称: 最小化每艘船中浪费的空间, 就会最小化船的数量.

证明: 假设这个方案不是最好的. 而是另有方案

乘船问题

- Case 1: i不合任何人同坐一艘船. 但是完全可以把j拉过来坐, 不会减少船的个数
- Case 2: i 和另一个人 k 同船, 完全可以把k和j交换, 我的船也不会超重, 所以解不一定变得更差

■

- 贪心策略并不丢失最优解
  - 并不会变得更差就可以了. 不一定要说明一定会变得更优.

P1094 纪念品分组

问题：他要把购来的纪念品根据价格进行分组，但每组最多只能包括两件纪念品，并且每组纪念品的价格之和不能超过一个给定的整数。分组的数目最少。

区间的例子

选择不相交的区间

“我们是在结局问题, 不是套公式套模板, 而是理解问题的结构!”

问题: 数轴上面有n个开区间(a_i, b_i). 选择尽量多的区间, 使得区间两两没有公共点.

最好画几个例子看一看...

- 画出 3 个区间的所有可能情况...?
- 怎么写代码?

观察 1: 如果区间 x 包含区间y, 那么肯定应该选 y.

- 区间数目不会减少
- 还能留更多的位置给别的区间选择

大家会给出怎样的贪心策略?

选择不相交的区间

- 最短的活动优先?

策略: 按照活动结束的时间排序, 一定要选第一个区间.

为什么?

证明: 现在的区间: $b_1 \leq b_2 \leq ... \leq b_n$

考虑a_1, a_2的大小关系:

- $a_1 > a_2$, 区间 2 包含区间 1, 无论如何都不会选择区间 2.
  - 只要任何一个i满足$a_1 > a_i$都不考虑(不明智)
- $a_1 \leq a_2 \leq a_3 \leq a_4 \leq ...$
  - 区间 2 和区间 1 完全不相交: 不会有任何影响;
  - 区间 2 与区间 1 有一部分重叠: 区间 1 的开头到区间 2 的开始这段是不受影响的;
    - 实际有用的区间: [区间 2 的开头, 区间 1 的结尾]

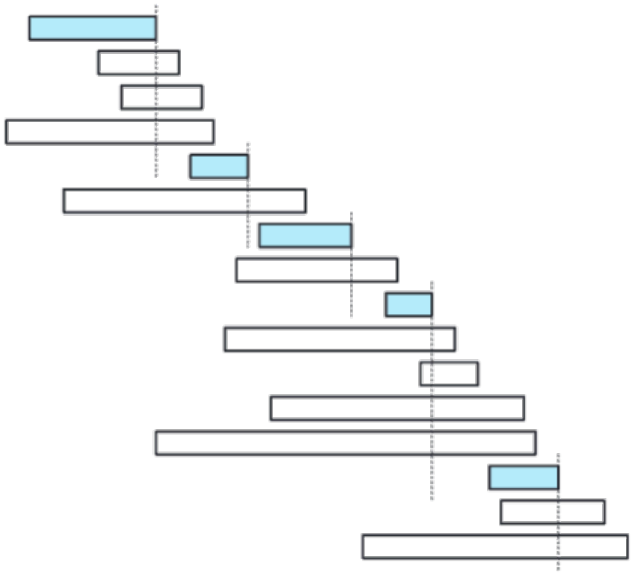
选择不相交的区间

- 这样一来若选择区间 2 就相当于选了一个被包含的大区间!!

综上所述, 一定要选择第一个区间. ■

策略(继续): 在选择完第一个区间之后, 把所有与第一个相交的内容排除在外, 记录上一个选择的区间编号, 再次进行选取.

问答: 试说出刚刚过程的子问题是什么.



区间选点问题

问题：数轴上面有n个闭区间$[a_i, b_i]$. 选择尽量少的区间, 使得每个区间内至少有 1 个点(不同区间包含的点可能是同一个).

定义: 一个区间已经被 “满足” := 这个区间里面已经选择了一个点.

观察: 小区间被满足 $\rightarrow$ 大区间一定被满足了!

- 首先按照b_i从小到大排序(和上题目一样)
  - 起点相同的话, 小的区间排在前面(a递增)
- 考虑第一个区间: 选择哪个点?
  - 注意已经按照b_i排好序了
  - 取得最后一个点(把中间的点挪到后面会让更多的区间满足)

区间覆盖问题

问题：数轴上面有n个闭区间$[a_i, b_i]$. 选择尽量少的区间, 覆盖一条指定的线段$[s, t]$.

观察: 每个区间在$[s, t]$以外的部分没有影响, 可以直接砍掉

还是观察: 和刚刚相反, 小区间的存在毫无意义!

- 按照起点a_i排序, 如果 1 的起点不是s, 无解; (总有一段覆盖不上去)
- 否则选择起点为s的最长区间$[a_i, b_i]$.
- 选择后, 忽略所有的在b_i前的部分.

Huffman 树和贪心策略

需求: 压缩文件

```
THISSENTENCECONTAINSTHREEASTHREECSTWODSTWENTYSIXESFIVEFST
HREEGSEIGHTHSTHIRTEENISTWOLSSIXTEENNSNINEOSSIXRSTWENTYSEV
ENSSTWENTYTWOTSTWOUSFIVEVSEIGHTWSFOURXSFIVEYSANDONLYONEZ
```

- 做映射: 字母 $\rightarrow$ 0,1 编码的代码

要求

- 尽可能短: 出现的越频繁 $\rightarrow$ 对应的编码越短
- 可以恢复: 没有一个编码的是另一个编码的前缀
  - 例如, $a = 10, b = 101$不可以!

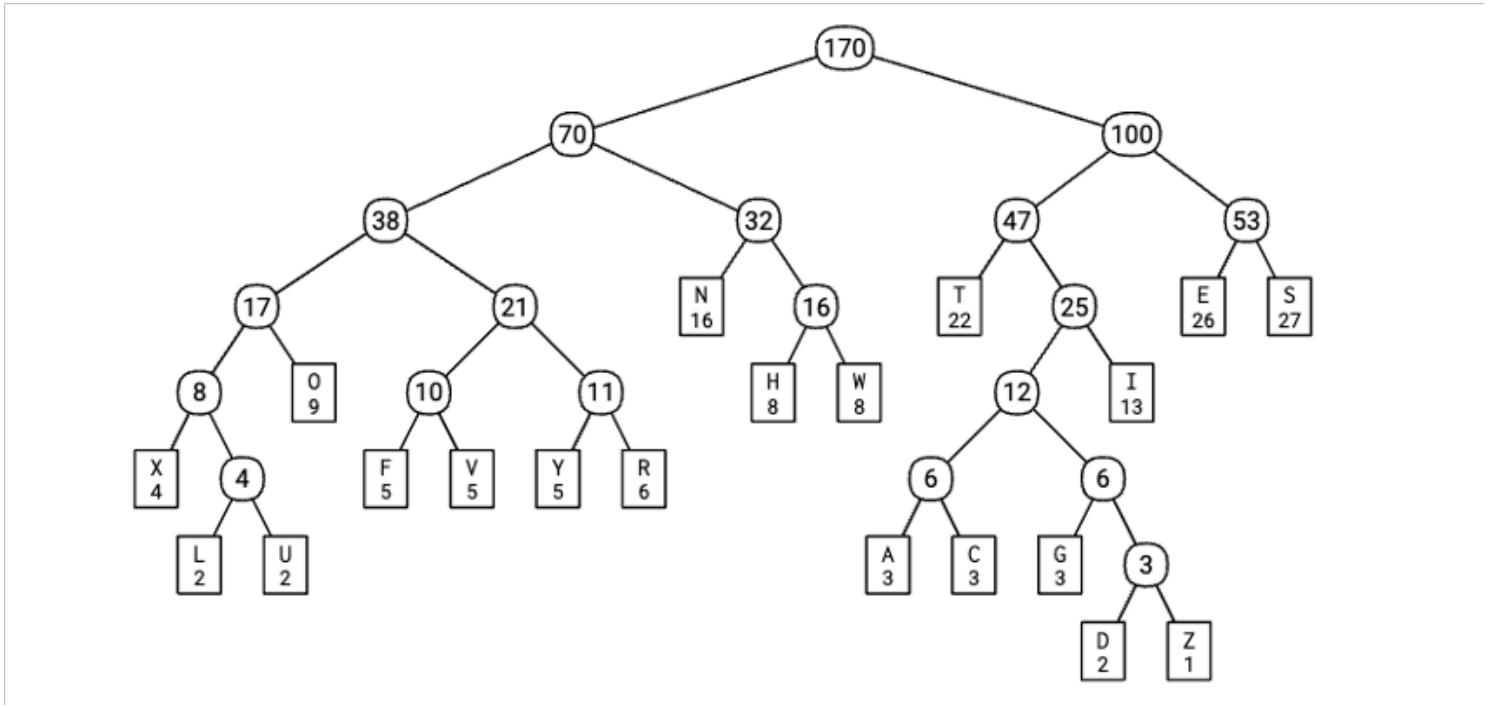
形式化的描述

- 要对一个写在n字符字母表中的消息进行编码，我们希望编码后的消息尽可能短。
- $f[1..n]$ 记录了每一个频率的出现次数
- 寻找一个前缀树, 最小化消息的长度.

例子

THISSENTENCECONTAINSTHREEASTHREECSTWODSTWENTYSIXESFIVEFST
HREEGSEIGHTHSTHIRTEENISTWOLSSIXTEENNSNINEOSSIXRSTWENTYSEV
ENSSTWENTYTWOTSTWOUSFIVEVSEIGHTWSFOURXSFIVEYSANDONLYONEZ

往左边走是 0, 往右边走是 1.



最优编码问题

问题: 给出n个字符的频率c_i, 给每一个字符赋予一个 0, 1 编码串. 要求任意一个编码不是另一个字符编码的前缀, 而且编码后总长度(每个字符的频率与编码长度乘积的总和)尽量小.

对应关系: 满足上述的内容的东西是可以表示为一个二叉树

- 叶子节点是字符
- 经过的路径是字符对应的编码

问题转化为: 如何构造一棵最优二叉编码树?

构造最优编码树

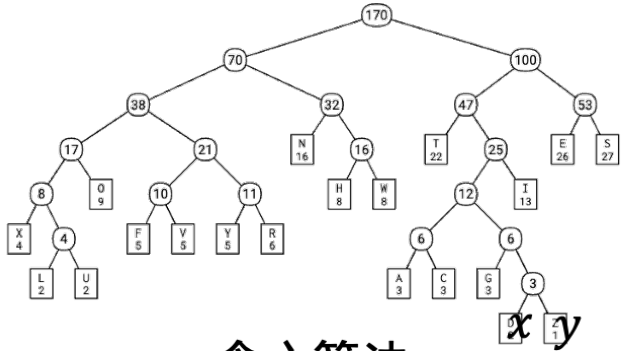
策略:

- 把每个字符看做一个单节点的子树放到树集合中,
  - 每棵子树的权值等于对应字符的频率.
- 每次取当前集合中权值最小的两元素; 构造成一个新树. 新树的权值等于原来的权值之和.

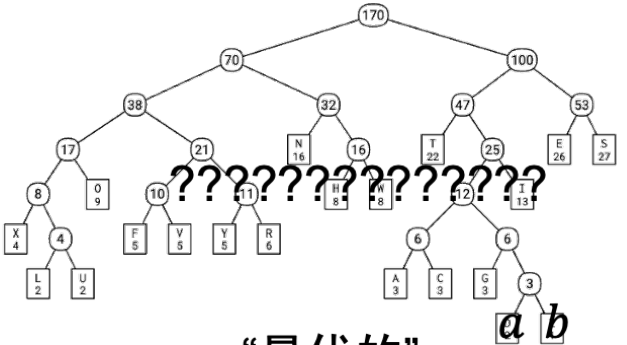
1. (初始的时候对吗?) 设x, y是两个出现频率最小的字符, 存在前缀码使得x, y的长度相等, 只有最后一位不同.

   证明: 假设不是这样的, 可以把出现次数最小的东西换过来, 就会得到更小的值.

构造最优编码树



贪心算法



“最优的” $\neq x, y$

即, 不妨设$f(x) \leq f(y), f(a) \leq f(b)$, 根据假设条件有$f(x) \leq f(a), f(y) \leq f(b)$. 如果$x \neq a$, 就交换$x \leftrightarrow a$, 如果$y \neq b$, 就交换$y \leftrightarrow b$. 这样出现更多的就变浅了, 不会变得更糟糕. ■

2. (原问题最优解包含子问题的最优解) 如果T是字符集C的一个最优编码树, x, y是T的两个互为兄弟的叶子节点, z是x, y的父节点, 如果把z看做频率为$f(z) = f(x) + f(y)$的字符, 那么

构造最优编码树

$T' = T - \{x, y\}$是字符集$C' = C - \{x, y\} \cup \{z\}$的最优编码树.

证明:

- T'的编码长度为L, 字符$\{x, y\}$对应深度为h, 如果把x, y拆成两个的话, 长度变为

$$L - (f(x) + f(y))h + (f(x) + f(y))(h + 1)$$
$$= L + f(x) + f(y)$$

因此只有T'是C'的最优编码树, 才可以整个最优.

■

- 原问题的最优解包含子问题的最优解 $\Rightarrow$ 最优子结构(optimal substructure)

问题的结构

直观感受

从大的集合里面

- 选出一些东西
- 排一个顺序

达成我们的某个目标(最小化某个东西)

问题的另一种感知方式

- 正常思路: 构造一个某算法, 使其产生最优解
- 奇怪的思路: 可以先随便找一个顺序, 然后如果更换顺序的过程中产生了更优解, 就相当于有了个更好的解; 直到再次交换无法使目标变得更小.

例如: 冒泡排序

```
for (int i = 0; i < n; i++)
    for (int j = i + 1; j < n; j++)
```

直观感受

```
        if (a[i] > a[j]) {
            swap(a[i], a[j]); // 铲除逆序对
        }
```

实际上表达的是

```
while (!is_sorted(a)) {
    // 返回任意 i < j, a[i] > a[j]
    auto [i, j] = find_unordered();
    swap(a[i], a[j]);
}
```

- 如果不存在$(i < j) \wedge \left(a_i > a_j\right)$, 一定排好序了
- 不存在$a_i > a_i + 1$, 一定排好序了 $\leftarrow$冒泡排序做的事情

交换论证

回顾刚才: 提出一个策略P, 采用反证法

- 假设还有一个 “更优的解答"...
- 把 “更优的解答” 的某个东西换掉, 按照我这个策略的某个东西填上
- 得到 “更更优” 的解答(矛盾)

或者说贪心法的原理:

- 分成了若干个阶段, 子阶段最优会让全部的最优(最优子结构)
- 我这个策略在任何的情况下并不会让解答变坏(可以贪心选择)

覆盖一类问题的理论

定义 0.1: (拟阵理论) 拟阵是一个满足下列条件的二元组 (S, L)

- S是一个有穷集合. (如刚刚硬币问题里面的$S = \{1, 2, 5, 10\}$)
- L是S的非空子集构成的集合的一个子集. L中的元素被称为独立集 (如$L = \{\{1\}, \{2\}, \{5\}, \{1, 2\}, \{1, 5\}, \{2, 5\}\}$, 有 6 个独立集).
- M满足遗传性: 对于独立集B, 若$B \in L$,则$\forall A \subset B$, 有 $A \in L$. (如$B = \{2, 5\}, A = \{2\}, A \in L$).
- M满足扩展性: $\forall A \in L, B \in L, |A| < |B|, (\exists x \in (B - A), \text{s.t. } A \cup \{x\} \in L)$.
  - 如$A = \{2\}, B = \{2, 5\}, x = 2, A \cup \{2\} = \{2, 5\} \in L$.

有点眼熟

- 遗传性: 刚刚的 “小区间可以被满足, 那么大区间一定可以”
- 扩展性: 可以由一个 “已经怎么决定的区间” 的最优解扩展到了大一点的 “还没决定好” 的区间.

贪心的结果: 为S中的每一个元素设定一个权值$w(i)$, 最优化这个

- 最大独立集: 如果一个独立集合不能再扩充了, 就称为最大独立集.

定理 0.1: 同一拟阵的最大独立集的元素个数相同.

证明: 若不然, 假设$|A|, |B|$是两个大小不同的独立集, 且不妨设$|A| < |B|$. 那么根据交换性,A是可扩充的, 不满足最大独立集的定义. 矛盾! ■

贪心的结果: 为S中的每一个元素设定一个权值$w(i)$, 最优化这个

- 对于S的子集A构成的权值$w(A) = \sum_{x \in A} w(x)$.
- 权值最大独立集一定是极大独立集. 怎么找?

还是贪心!

```
Greedy(拟阵<S, L>, 权值 w)
  A={}
  sort S by w // 把 S 内的元素 x 按照权值 w(x)从大到小降序排序
  for x in S
    if(A ∪ {x} ∈ L)
      A=A ∪ {x}
  return A
```

- 能够构造成拟阵的一定可以使用贪心
- 但是不能的有时候也可以用...

练习题

P1181 数列分段

问题：对于给定的一个长度为 N 的正整数数列 A_i，现要将其分成连续的若干段，并且每段和不超过 M (可以等于 M)，问最少能将其分成多少段使得满足要求。

- 空间限制比较死，所以肯定是能加一个进去就加一个进去了.

P1208 混合牛奶

问题：M 乳业从一些奶农手中采购牛奶，并且每一位奶农为乳制品加工企业提供的价格可能相同。此外每位奶农每天能提供的牛奶数量是一定的。每天 M 乳业可以从奶农手中采购到小于或者等于奶农最大产量的整数数量的牛奶。给出 M 乳业每天对牛奶的需求量，还有每位奶农提供的牛奶单价和产量。计算采购足够数量的牛奶所需的最小花费。每天所有奶农的总产量大于 M 乳业的需求量。

策略: 每次选单价最小的，直到完成任务

P4995 跳跳

问题: 有一堆高矮不同的石头, 第i块高度为h_i. 地面的高度为$h_0 = 0$. 从第i块跳到第j块耗费的体力值为$\left(h_i - h_j\right)^2$. 要求跳到每个石头上面各 1 次, 耗费最多的体力值. 问跳跃序列.

策略: 考虑相邻两次差值尽可能大?

- 最左一下, 最右一下, 第二左一下, 第二右边一下...

P1199 三国游戏

问题：《三国》的游戏中，共有$N \geq 4$，N是偶数个武将. 两个武将之间有一个“默契值”，表示若此两位武将作为一对组合作战时，该组合的威力有多大。

游戏开始，小涵和计算机要从自由武将中挑选武将组成自己的军队，规则如下：小涵先从自由武将中选出一个加入自己的军队，然后计算机也从自由武将中选出一个加入计算机方的军队。接下来一直按照“小涵 $\rightarrow$ 计算机 $\rightarrow$ 小涵 $\rightarrow$ ...”的顺序选择武将，直到所有的武将被双方均分完。然后，程序自动从双方军队中各挑出一对默契值最高的武将组合代表自己的军队进行二对二比武，拥有更高默契值的一对武将组合获胜，表示两军交战，拥有获胜武将组合的一方获胜。

P1199 三国游戏

计算机一方选择武将的原则是尽量破坏对手下一步将形成的最强组合，它采取的具体策略如下：任何时刻，轮到计算机挑选时，它会尝试将对手军队中的每个武将与当前每个自由武将进行一一配对，找出所有配对中默契值最高的那对武将组合，并将该组合中的自由武将选入自己的军队。

UVA1615 高速公路

问题:

1. 高速公路是从原点 $(0,0)$ 到 $(L,0)$ 的线段。
2. 村庄在平面上标记为点集 (x_i,y_i)。
3. 每个村庄到最近的天桥的欧几里得距离不超过 D。

目标: 最少需要修建多少个天桥，使得每个村庄到最近天桥的距离不超过 D。

类似与区间覆盖问题

- 区间是什么?

CF1995C 平方

问题:

- 给一个数组a, 每次可以选择一个下标, 把那个数平方.
- 最少多少次, 使得数组单调不降?

范围: $n \leq 2 \times 10^5; a_i \leq 10^6$

- 前面的内容应该尽可能的小, 遇见不是单调的内容元素就去平方它

这是因为 $x, y > 0 \Rightarrow x^2 > y^2$

- 反过来, $x^2 > y^2 \Rightarrow |x| > |y|$. 根据$x, y > 0$, 意味着$x > y$

问题:

- $a_i \leq 10^6$, 这平方两下不就爆炸了?

补充知识: 对数的发明

计算很大的数: 16A.D.的欧洲天文学

- 天文学蓬勃发展, 计算成为了主要的障碍. (数据非常不友好)
- Scotland 的 Napir 说乘法有简单的方法.

“数据那么大, 干脆用基于某个数的指数就好了嘛!”

- 2 的几次方

1	2	3	4	5	6	7
2	4	8	16	32	64	128

- 这样便可以把乘法换成加法:
  - $8 \times 16 = 2^3 \times 2^4 = 2^7 = 128$
- 问题: 3 怎么用 “2 的几次方” 表示?

补充知识: 对数的发明

- 大概写成 $1.584926\ldots$ 是个无理数
- 2 的 1.584926 次方大约等于 3

当时: 花费了 20 年写了一本书

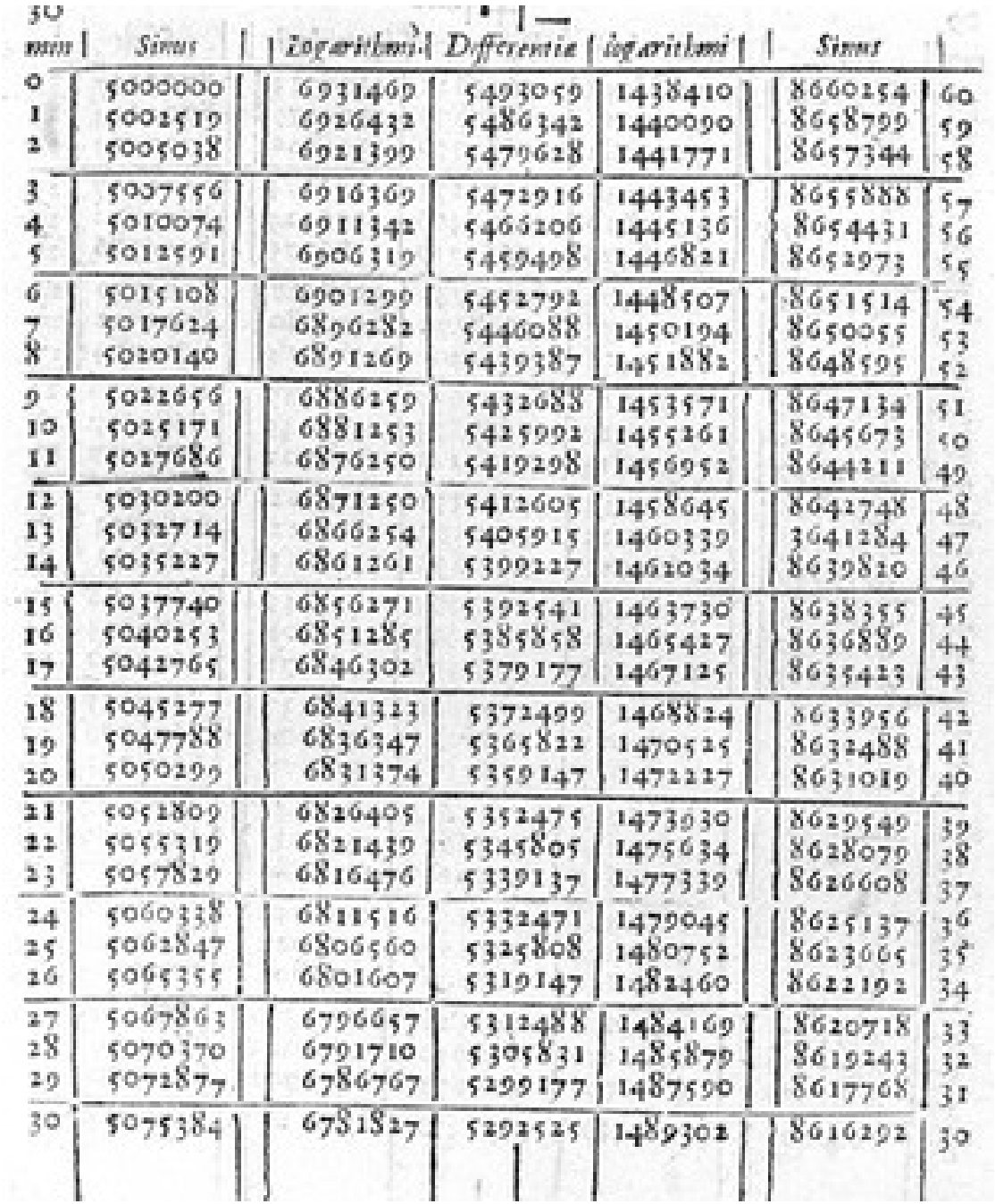
这就是为什么我们要关注$2^{?} = 3$.

“这个工具的诞生使得天文学家的寿命加倍.”

我们更关注它的"性质"而不是"结果", 因此我们可以用一个符号来代表它.

“我们给他起一个$? = \log_2 3$”. 可以不用写 $\log_2^3$.

补充知识: 对数的发明



min	Sinus	Logaritmi	Differentiæ	logaritmi	Sinus	
0	5000000	6931469	5493059	1438410	8660254	60
1	5002519	6926432	5486342	1440090	8658799	59
2	5005038	6921399	5479628	1441771	8657344	58
3	5007556	6916369	5472916	1443453	8655888	57
4	5010074	6911342	5466206	1445136	8654431	56
5	5012591	6906319	5459498	1446821	8652973	55
6	5015108	6901299	5452792	1448507	8651514	54
7	5017624	6896282	5446088	1450194	8650055	53
8	5020140	6891269	5439387	1451882	8648595	52
9	5022656	6886259	5432688	1453571	8647134	51
10	5025171	6881253	5425992	1455261	8645673	50
11	5027686	6876250	5419298	1456952	8644211	49
12	5030200	6871250	5412605	1458645	8642748	48
13	5032714	6866254	5405915	1460339	3641284	47
14	5035227	6861261	5399227	1462034	8639820	46
15	5037740	6856271	5392541	1463730	8638355	45
16	5040253	6851285	5385858	1465427	8636889	44
17	5042765	6846302	5379177	1467125	8635423	43
18	5045277	6841323	5372499	1468824	8633956	42
19	5047788	6836347	5365822	1470525	8632488	41
20	5050299	6831374	5359147	1472227	8631019	40
21	5052809	6826405	5352475	1473930	8629549	39
22	5055319	6821439	5345805	1475634	8628079	38
23	5057829	6816476	5339137	1477339	8626608	37
24	5060338	6811516	5332471	1479045	8625137	36
25	5062847	6806560	5325808	1480752	8623665	35
26	5065355	6801607	5319147	1482460	8622192	34
27	5067863	6796657	5312488	1484169	8620718	33
28	5070370	6791710	5305831	1485879	8619243	32
29	5072877	6786767	5299177	1487590	8617768	31
30	5075384	6781827	5292525	1489302	8616292	30

补充知识: 对数的发明

为什么叫对数?

- 表"对着"的那个东西才能算.
- $m =: \log_a b$ 意味着 $a^m = b$

意思是: a的多少次方等于b? 那个答案就是m.

- a在底下, 右边a也在底下. $a > 0, a \neq 1$.
- b是真数, 就是那个结果, 真数大于 0.
- m叫做对数
- 它们可以互换, 我们可以在不清楚的时候想想.

快速回答: $\log_2 8, \log_{27} 9, \log_{16} 8$;

计算机科学中经常使用以 2 为底的对数.

补充知识: 对数的性质

1. 废话恒等式: $a^{\log_a b} = b$.
   - 比如$8 = 2^3 = 3^{\log_2 8}$
2. 关于 1: $\log_a 1 = 0; \log_a a = 1 (a > 0, a \neq 1)$
3. 运算(优先级高于乘法)

$$\log_a M + \log_a N = \log_a MN$$

$$\log_a M - \log_a N = \log_a \left(\frac{M}{N}\right)$$

$$\log_a M^b = b \log_a M$$

证明: 方法就是化成熟悉的熟悉的指数形式; 参见高中课本. ■

为什么复杂度我们会说$\log n$级别的而不是$\log_2 n$级别的?

补充知识: 对数的性质

- 对数之间 “增长得差不多快”
- 实际上不同底数的对数增长只是差一个常数!

考虑$\log_a M \sim \log_b M$, 改成指数形式:

$$a^x = M, b^y = M, a^x = M = b^y$$

使用废话恒等式:

$$b^{\log_b a^x} = b^{x \log_b a} = b^x \Longrightarrow y = x \log_b a$$

用对数的记号表示: $\log_b M = \log_a M \log_b M$, 也就是

$$\log_a M = \frac{\log_b M}{\log_b a}$$

数学素养(Mathematical Maturity)

The Five-Strand Model of Mathematical Proficiency

Once we have become more familiar and fluent with using language that distinguishes between the intellectual, behavioral, and emotional domains, it is useful to further specify proficiency within those domains. One means of achieving this can be found in the 2001 National Research Council report *Adding It Up: Helping Children Learn Mathematics*, where a five-strand model of mathematical proficiency was introduced. While this model was motivated by research on student learning at the K-8 level, in my opinion it is an excellent model through at least the first two years of college, if not beyond. In this model, mathematical proficiency is defined through the following five attributes (see Chapter 4 of *Adding It Up* for details).

1. *conceptual understanding* — comprehension of mathematical concepts, operations, and relations
2. *procedural fluency* — skill in carrying out procedures flexibly, accurately, efficiently, and appropriately
3. *strategic competence* — ability to formulate, represent, and solve mathematical problems
4. *adaptive reasoning* — capacity for logical thought, reflection, explanation, and justification
5. *productive disposition* — habitual inclination to see mathematics as sensible, useful, and worthwhile, coupled with a belief in diligence and one's own efficacy.

The five-strand model and the three psychological domains weave together well. In particular, one can view the first two strands as refinements of the intellectual domain, the third and fourth strands as refinements of the behavioral domain, and the fifth in alignment with the emotional domain.

回到刚刚的练习题

感觉“满足某种情况”, 但是难找规律?

数学符号来这里理理头绪!

- 假设对a_{i-1}操作k_{i-1}次, a_i操作k_i次
- 满足条件: $a_{i-1}^{2^{k_i-1}} \leq a_i^{2^{k_i}}$
- 幂次难以处理, 两边取对数把东西丢到前面来: $\log_2 a_{i-1}^{2^{k_i-1}} \leq \log_2 a_i^{2^{k_i}}$
- ...
- $k_{i-1} + \log_2 \log_2 a_{i-1} - \log_2 \log_2 k_i \leq k_i$
- $k_{i_{\min}} = \lceil k_{i-1} + \log_2 \log_2 a_{i-1} - \log_2 \log_2 k_i \leq k_i \rceil$

于是可以由k_{i-1}推出k_i. !

- 注意浮点数的误差

CF1992D Test for love

问题：一个人在水里游泳(W), 总共不能游超过k个格子. 水中有鳄鱼(C)和木头(W).

- 从左边(0)移动到右边($L+1$). 有n个字符表示第i米的河水什么情况

规则:

- 在物品的表面上的时候(陆地/木头): 可以往前跳不超过$m(1 \le m \le 10)$米;
- 如果在水里的时候, 每次只能往前进 1 米
- 无论如何不能接触含有鳄鱼的那格子.

问: 这人是否可以到达对岸?

- 策略:

CF1992D Test for love

- 尽可能跳到当前尽可能远的可以着陆的地方
- 否则, 实在不行只能游泳(不能超过m格子)
- 中间有鳄鱼就寄了

为什么可行? 有没有可能跳了更近的反而更容易到终点?

- 假设跳到了一个更近的原木而不是最远的, 还可能会损失下一跳原本能跳到的更远的地方.
- 中间的内容要么要游泳白白消耗体力

CF 1990C Mad MAD Sum

问题：定义一个序列的 MAD 为这个序列中最大的出现最多次数的数字. 如果没有数字出现了 2 次, 值为 0.

- 给一个大小为n的数组a, 初始的时候变量 sum=0

只要a中存在不为 0 的数, 就重复下列循环:

- 令$\text{sum} := \text{sum} + \sum_{i=1}^{n} a_i$
- 给定b是一个大小为n的数组. $b_i := \text{MAD}([a_1, a_2, ..., a_i]), \forall i = 1..n$, 然后令$a_i := b_i, \forall i = 1..n$

观察:

- 在一次操作之后, 序列变得单调不减
- 在所有数都相等的时候, 似乎所有的数往右移动了
- 而且由于我们要最大的值, 所以最大值是从左往右继承的

观察 2: 只出现 1 次的数, 操作 2 次就会消失

- 操作 2 次后, 其他的数都大于 0.
- 继续右移, 第i个数的贡献是$a_i \times (n - i + 1)$.

P1080 国王游戏

问题：国王邀请n个大臣玩一个有奖游戏

- 每个人 (国王和大臣) 在左、右手上各写下一个整数 (给出这些数字)
- 所有人排成一队，国王站在队伍的最前面
- 所有的大臣都会获得国王奖赏的若干金币，每个大臣获得排在该大臣前面的所有人 (包括国王) 的左手上的数的乘积除以他自己右手上的数，然后向下取整得到的结果

国王不希望某个大臣获得特别多的奖赏，所以他想请你帮他安排一下队伍的顺序，使得获得奖赏最多的大臣所获奖赏尽可能的少

P1080 国王游戏

- 希望右手越小越好; 往后面排; 但是排在后面可能前面的乘积越大...
- 直觉不够用了

怎么办?

- 不要猜，试图去理解问题
  - 套过所有 “模板”，找不到就放弃?
- 数学是帮我们简化问题的手段
  - 符号弥补了人类短时记忆有限的缺点

简化的情况

- 国王的左手, 右手是l_0, r_0; 第一个人的是l_1, r_1, 第二个人的是l_2, r_2.
- 按照$1, 2$的顺序:
  - 第一个人: $\frac{l_0}{r_1}$, 第二个人$\frac{l_0 l_1}{r_2}$

P1080 国王游戏

- 按照2, 1的顺序
  - 第一个人: $\frac{l_0}{r_2}$, 第二个人$\frac{l_0 l_2}{r_1}$

即: 何时$\max\left(\frac{l_0}{r_1}, \frac{l_0 l_1}{r_2}\right) < \max\left(\frac{l_0}{r_2}, \frac{l_0 l_2}{r_1}\right)$?

- 化简, 把l_0都约去, 有$\max\left(\frac{1}{r_1}, \frac{l_1}{r_2}\right) < \max\left(\frac{1}{r_2}, \frac{l_2}{r_1}\right)$?

问: 什么时候交换是有好处的 $\rightarrow$ 解这样不等式的值

处理的方法: 分类讨论

1. (1)(3)前提是$\frac{1}{r_1} \geq \frac{l_1}{r_2}, \frac{1}{r_2} \geq \frac{l_2}{r_1}$, 满足$\frac{1}{r_1} < \frac{1}{r_2}$
   - $(r_2 \geq l_1 r_1) \wedge (r_1 \geq l_2 r_2) \wedge (r_1 > r_2)$.
   - 不可能的情况!
2. (2)(4) $\frac{l_1}{r_2} > \frac{1}{r_1}, \frac{l_2}{r_1} > \frac{1}{r_2}, \frac{l_1}{r_2} < \frac{l_2}{r_1}$.
   - $(r_2 < l_1 r_1) \wedge (r_2 l_2 > r_1) \wedge (l_1 r_1 < l_2 r_2) \leftarrow$ 排序的依据
3. 4. 请自行推导.

P1080 国王游戏

意味着$l_1r_1 < l_2r_2$的要小的会小.

思考题: 在推导的时候, 如果把整数变为$\mathbb{R}$, 就不能贪心了

总结

Takeaway messages

贪心的两种表达方式:

- 排好一个 “不后悔” 的顺序
- 交换交换, 使得结果变得更优

用好草稿纸