模拟赛题解

鸽子

June 28, 2021

题目名称	区间	区间	区间
题目类型	传统型	传统型	传统型
目录	`interval`	`intervbl`	`intervcl`
可执行文件名	`interval`	`intervbl`	`intervcl`
输入文件名	`interval.in`	`intervbl.in`	`intervcl.in`
输出文件名	`interval.out`	`intervbl.out`	`intervcl.out`
每个测试点时限	2 秒	4 秒	1 秒
内存限制	512 MiB	512 MiB	512 MiB
子任务数目	5	7	6
是否捆绑测试	是	是	是

区间（interval）

算法零

我会暴力！

每次询问枚举所有可能的子序列，然后再判断就行。

怎么判断？显然可以 $O(n^2)$ 嘛。

时间复杂度 $O(qm^n n^2)$，期望得分 0 分。

算法一

容易发现上面那个算法其实干了很多没有意义的枚举。

考虑直接枚举最长公共子序列，这样直接在原序列中删去一个就行，只有 $O(n)$ 个。

然后再在某个位置添加一个，只有 $O(nm)$ 种。

把和原序列相同以及重复的去掉，就可以得到答案了。

复杂度 $O(qn^2m)$，期望得分 5 分。

算法二

设 a, b 的最长公共前缀长度为 i，最长公共后缀长度为 j。

不妨假设询问的是整个序列。

若 $i + j = n - 1$，那么显然有 $m - 1$ 种方案。

否则 $i + j \le n - 2$，此时 $a[i + 1]$ 和 $a[n - j]$ 中至少一个出现在最长公共子序列中，分下面三种情况：

- $a[i + 1]$ 出现在某个最长公共子序列中
  - 显然 $a[n - j]$ 不在这个子序列中，于是 $a[i + 1..n - j - 1]$ 与 $b[i + 2..n - j]$ 相等。由于 $b[n - j] \ne a[n - j]$，$a[n - j - 1] \ne a[n - j]$，此时任取 $b[i + 1] \ne a[i + 1]$ 即可，有 $m - 1$ 中方案。
- $a[n - j]$ 出现在某个最长公共子序列中
  - 类似上面可得只要 $a[i + 1] \ne a[i + 2]$ 就有 $m - 1$ 种。
- $a[i + 1]$ 和 $a[n - j]$ 均出现在某个最长公共子序列中
  - 显然 $a[i + 1] \ne a[i + 2]$ 且 $a[n - j] \ne a[n - j - 1]$。
  - 此时 $a[i + 1..n - j - 1] = b[i + 2..n - j], a[i + 2..n - j] = b[i + 1..n - j - 1]$，从而 $a[i + 1..n - j - 2] = a[i + 3..n - j]$。

这样 $O(n^2)$ 枚举 i, j，容易 $O(1)$ 求出对应方案数。

复杂度 $O(qn^2)$，期望得分 $5 \sim 24$ 分。

算法三

实际上，对于刚才的前两种情况，是可以直接计算结果的。

对于第三种情况，我们考虑枚举满足条件的 i，显然 $j \geq n - i - 2 - \text{lcp}(a[i+1..n], a[i+3..n])$。同时，显然此时 j 一定满足 $a[n-j] \neq a[n-j-1]$。

这样容易做到 $O(n)$。

复杂度 $O(qn)$，期望得分 $24 \sim 46$ 分。

算法四

实际上，对于前两种情况，是可以合并的，换句话说只要 $a[i+1] \neq a[i+2]$ 那么就有 n 种方案。

这样就只需要考虑第三种情况了，此时相当于对 $\text{lcp}(a[i+1..n], a[i+3..n])$ 求和。

显然对于前一部分 i，答案是不变的。而对于后面的 i，lcp 会同时减去一个值。

二分一下就好了。

复杂度 $O(q \log n)$，期望得分 $80 \sim 100$ 分。

算法五

显然这只和右端点有关，而且关于右端点是单调的，双指针扫扫就好了。

复杂度 $O(n+q)$，期望得分 100 分。

区间（intervbl）

算法一

我好像知道一个东西叫做 sort！
期望得分 15 分，实际得分 15 分。

算法二

对于子任务二，所有操作都是对整个序列的。
那不是几个变量就解决了。
结合算法一可获得 25 分。

算法三

对于子任务三，只有一次排序。
实际上这相当于不进行排序。
用 Splay 维护之前的三种操作就好了。
结合算法一、二可获得 37 分。

算法四

对于子任务四，所有数的权值都很小。
用 Splay 维护区间 $0, \pm 1$ 的个数即可。
结合算法一、二、三可获得 50 分。

算法五

对于后三个子任务，用线段树维护排序过的区间，Splay 维护这些区间即可。注意空间常数。
期望得分 $64 \sim 100$ 分。

区间（intervcl）

记 C 为满足条件的数的个数。

先来考虑如何求出 C，可以发现询问的答案和原序列的顺序无关，因此可以对原序列排序后二分求出区间 $[l', r']$，这部分的复杂度是 $O(q \log n)$。

当 $C < k$ 时，直接依题输出 -1。否则，由于抽取 k 个数是等概率随机的，因此期望就是所有方案的平均值，故我们只需求出所有方案的样本 (最大值) 总和与方案的个数。

算法一

容易发现第一个子任务满足 $k = 1$，也就是**只抽取一个数**。

那么方案数就是 C，最大值总和就是区间所有数的和。

由于我们对原序列进行了排序，因此所询问的数一定是一段连续的区间。

故原题就变成了求**静态区间求和**的问题，可以使用一个叫做前缀和~~的数据结构~~来维护。

别忘了最后还要求 C 的逆元，你用哪种方法求逆元都可以过 ~~(不要告诉我你一种都不会)~~，期望得分 8 分。

算法二

先来看一下方案的个数。

方案的个数很简单，由组合数 (二项式系数) 的定义，可得 C 个数选 k 个数的总方案数等于 $\binom{C}{k}$，可以使用阶乘逆元~~或杨辉三角~~预处理。

对于所有方案的最大值总和，可以考虑暴力模拟，即使用 dfs 搜索枚举的 k 个数，然后将最大值加起来即可。

时间复杂度 $O\left(q \cdot \binom{n}{k}\right)$，期望得分 $0 \sim 11$ 分。

算法三

我们对区间中的每个数进行分析。

考虑一个排序后的 (大小为 C 的) 区间 $b_1 \geq b_2 \geq b_3 \geq \cdots \geq b_C$，在其中抽取 k $(1 \leq k \leq C)$ 个数。

那么有多少种情况满足 b_1 为最大值呢？那么 b_1 必须要选上，且其余 $C-1$ 个中还要选 $k-1$ 个，即 $\binom{C-1}{k-1}$ 种方案，故 b_1 的贡献为 $\binom{C-1}{k-1} \cdot b_1$。

类似地，又有多少种情况满足 b_i 为最大值呢？因此前面 $i-1$ 个数不能选，且 b_i 必须选，后面的 $C-i$ 个中选 $k-1$，因此方案数为 $\binom{C-i}{k-1}$，故贡献为 $\binom{C-i}{k-1} \cdot b_i$。

因此最大值总和 (以下简称**答案**) 就由下式给出：

$$\sum_{i=1}^{C}\binom{C-i}{k-1}\cdot b_i$$

单次询问 $O(n)$，总复杂度 $O(qn)$。可以通过第二个子任务，期望得分 11 分，结合算法一可得 19 分。

算法四

之后不妨假设原序列是**降序** (从大到小排序) 的。

考虑第三个子任务，恒有 $l=0$，因此询问的**右端点** r' 一定为 n。

那么设左端点为 u，则答案就是

$$\sum_{i=u}^{n}\binom{n-i}{k-1}\cdot b_i$$

可以发现，被求和项与左端点 u **无关**，且 k 为定值，因此可以对 $\binom{n-i}{k-1}\cdot b_i$ 作后缀和，那么第 i 项 s_i 就是 $u=i$ 时的答案。每次 $O(1)$ 得到答案。

单次询问 $O(1)$，总复杂度 $O(q\log n)$。可以通过第三个子任务，结合前述算法，期望得分 32 分。

算法五

考虑第四个子任务，恒有 $r=10^8\ge a_i$，故询问的**左端点** l' 一定为 1。

那么设右端点为 v，则答案就是

$$\sum_{i=1}^{v}\binom{v-i}{k-1}\cdot b_i$$

由于 k 为定值，记 $c_j=\binom{j}{k-1}$，则答案就变成 $\sum_{i=1}^{v}b_ic_{v-i}$。

可以看出，这是一个标准的**卷积**形式，利用 ~~FFT~~/FNTT 等多项式乘法算法可以在 $O(n\log n)$ 时间内得到 (对所有 i) 所有 $v=i$ 时的答案 t_i。

总时间复杂度 $O\left((n+q)\log n\right)$，可以通过第四个子任务，期望得分 47 分。

算法六

我们发现，算法五求得的数列有比较好的性质。我们记 $f_k[j]=\sum_{i=1}^{j}\binom{j-i}{k-1}\cdot b_i$。

设询问的左右端点分别为 l,r。由原先的式子可以得到，答案应该为 $\sum_{i=l}^{r}\binom{r-i}{k-1}\cdot b_i$，可以看出，它等于 $f_k[r]-\sum_{i=1}^{l-1}\binom{r-i}{k-1}\cdot b_i$。

因此我们只需求出 $\sum_{i=1}^{l-1}\binom{r-i}{k-1}\cdot b_i$ 的值即可。我们把它尝试表示成 $f_k[l-1]$ 的线性组合，即有

$$\sum_{i=1}^{l-1}\binom{r-i}{k-1}\cdot b_i=\sum_{j=1}^{k}\binom{r-l+1}{k-j}\cdot f_j[l-1]$$

接下来我们来证明这个表达式，先将右边展开，取 b_i 项的系数：

$$\sum_{j=1}^{k}\binom{r-l+1}{k-j}\cdot f_j[l-1]=\sum_{j=1}^{k}\binom{r-l+1}{k-j}\sum_{i=1}^{l-1}\binom{(l-1)-i}{j-1}\cdot b_i$$

因此 b_i 项的系数就为：

$$\sum_{j=1}^{k}\binom{r-l+1}{k-j}\binom{l-i-1}{j-1}=\sum_{j=0}^{k-1}\binom{r-l+1}{(k-1)-j}\binom{l-i-1}{j}=\binom{r-i}{k-1}$$

其中最后一步用了 Vandermonde 卷积公式，~~有直观的组合意义~~。这就证明了这个表达式是正确的。

于是单次询问复杂度从 $O(n)$ 变成了 $O(k)$，这样总的复杂度就是 $O\left(\sum k\right)$。

但是这样预处理的时空复杂度是 $O(n^2)$ 的，不可接受，于是我们希望 $k_{\max}$ 尽可能得小。

注意到题目中有这样一个条件，即 $\sum k\le 10^5$ (记 $P=\sum k$)，根据套路，我们可以**设定一个阈值** th，使得当 $k<th$ 时使用前述算法，$k>th$ 时直接使用暴力 (算法三)。

分析一下时间复杂度，首先排序和二分是必要的，这里就有 $O\left((n+q)\log n\right)$，然后使用 $k<th$ 时的算法需要执行预处理，预处理的时空复杂度均为 $O(th\cdot n)$，单次询问的复杂度为 $O(k)=O(th)$，故这部分的总复杂度为 $O\left((n+q)\cdot th\right)$。

当 $k>th$ 时，由于这样的询问不超过 $O\left(\frac{P}{th}\right)$ 个，每次暴力 $O(n)$，故复杂度为 $O\left(\frac{n\cdot P}{th}\right)$。

因此可以取 $th=O\left(\sqrt{n}\right)$，总复杂度 $O\left(n\sqrt{n}\right)$。分一下类，就可以过掉第五个子任务了，期望得分 63 分。

算法七

当 k 非定值时的算法其实是和算法六一样的，只是不是一开始就分类，是在询问时根据询问的 k 分类。还是可以按照上述分析复杂度，为 $O\left(n\sqrt{n}\right)$，期望得分 100 分。