《分治》解题报告

重庆市巴蜀中学 徐恺

2024 年 10 月 9 日

题目大意

对一个长度为 n 的非负整数序列 a 进行分治，将每个分治区间的 mex 求和。给定 n 求出所有长度为 n 的序列中和的最大值，对 998244353 取模。

数据范围

对于所有测试数据，$1 \le n < 2^{200000}$，且 n 以二进制形式给出。

测试点编号	$n <$	特殊限制
$1 \sim 2$	8	无
$3 \sim 4$	128	无
$5 \sim 6$	262144	无
$7 \sim 8$	2^{200}	A
$9 \sim 10$		无
$11 \sim 12$	2^{2000}	A
$13 \sim 14$		无
$15 \sim 16$	2^{200000}	A
$17 \sim 20$		无

特殊性质 A: 存在整数 k 使得 $n = 2^k$。

时间限制 3s，空间限制 256MB。

解题过程

特殊性质 A 观察

此时分治过程形如一棵满二叉树。满二叉树每个节点的两个儿子形态相同，所以可以钦定左儿子的 mex $\geq$ 右儿子。

假设当前区间 mex为 a，子区间分别为 b, c 且 $b \geq c$。我们在当前区间中，找到 $[0, a-1]$ 每种数各一个，并认为找到的这些位置进行了一次贡献。因为左儿子有 $[0, b-1]$，可以认为当前区间 $[0, b-1]$ 找到的位置是在左儿子进行过贡献的位置，而因为右儿子的 $c \leq b$，所以右儿子的 $[0, c-1]$ 是不如左儿子优秀的，直接将其舍弃掉。于是我们可以认为，一个位置在当前层做了贡献，如果当前层是父亲层的左儿子，则它能在父亲层继续做贡献，否则之后都不能做贡献了。

我们将数组的下标范围看做在 $[0, n-1]$ 间，那么可以分析出一个位置的贡献上界，就是其二进制表示下最长的连续的 0 的长度。可以达到这一上界，因为这些连续段都是某条重链的前缀（我们将左儿子看作重儿子划分轻重链），所以只需要将长度从大到小排序，依次确定每个位置的数即可。

特殊性质 A 计数

容斥，设 $f_{i,j}$ 表示长度为 i 的二进制段，最长 0 连续段 $\geq j$ 的方案数，则有

$$
\begin{aligned}
f_{i,j} =& \sum_{k=1} \binom{i-(j+1)k+k}{k} 2^{i-(j+1)k} \times (-1)^{k-1} \\
&+ \binom{i-j-(j+1)(k-1)+k-1}{k-1} 2^{i-j-(j+1)(k-1)} \times (-1)^{k-1} \\
=& \sum_{k=1} \binom{i-jk}{k} 2^{i-jk-k} \times (-1)^{k-1} + \binom{i-jk}{k-1} 2^{i-jk-k+1} \times (-1)^{k-1}
\end{aligned}
$$

令 l 为 n 在二进制表示下的数位长度，则满二叉树的情况答案为：$\sum_{j=1}^{n} f_{l-1,j}$。注意到求解 $f_{l-1,j}$ 时 k 的枚举范围是 $O(\frac{l}{j})$ 的，故时间复杂度为 $O(l \log l)$。

拓展到一般情况

现在分治两边大小可能不相等，我们证明最优策略依然是选择更长的儿子作为当前节点的重儿子，在本题中即为左儿子。

如果分治不是对半分，而是可以任意选择位置分成两部分，那么选择更长的儿子这一策略是错误的。本题可以选择更长的儿子的原因是对半分导致左右儿子结构特殊，左儿子可以视作在右儿子的基础上新增了一个节点。

于是我们可以归纳证明出这样的结论，对于一个长度为 n 的子分治过程，始终可以选择左儿子为后继，与祖先的划分方式无关。$n \leq 2$ 时这一结论是显然的，对于 $n > 2$ 的情况，因为左右儿子的划分策略一致，而左儿子比右儿子多一个节点，所以选择左儿子会更优秀。

一般情况计数

整个分治过程可以这样描述：当前区间长度为 x，若 $x = 1$ 则停止；否则令 $x = \lfloor \frac{x}{2} \rfloor$ 或 $\lceil \frac{x}{2} \rceil$。每次当 $x = 1$ 时，

过程中最长连续上取整次数 $+1$ 就是该位置的贡献。在区间长度的二进制表示下观察这个过程，就是每次取出最低位删除，如果取出的是 1 则需要进一位。

容易发现，分治次数只可能为 $l-1$ 或 l 次，其中为 l 次是因为前 $l-1$ 次分治过程中向第 l 位进了位，与原本的 1 相加之后再进了一位，于是就需要再进行一次分治。

我们可以这样计算答案：先算长度为 l 的满二叉树的答案，再加上进行 l 次分治且最后一次分治选择上取整的方案数。其原因是因为满二叉树答案将需要进行 l 次分治的方案忽略了最后一次分治，可以视作只计算了第 l 次分治下取整的方案，所以需要再补充第 l 次上取整的方案数。

考虑如何计算进行 l 次分治且最后一次分治选择上取整的方案数。

对于 1 到 $l-1$ 的每一位，我们都确定其是上取整还是下取整，并将连续的上取整或下取整称为一个连续段。

我们在二进制表示下从 $l-1$ 方向向 1 方向看，扫描每个连续段。若当前是上取整段，如果这一段有 1 则断定合法，否则继续扫描；若当前是下取整段，如果这一段有 0 则断定非法，否则继续扫描；若扫完了，则非法。

枚举 i，表示最远到 i 都判断不出合法性，但是之后一段 $i-1$ 到 j 就判定为了合法。容易发现 $l-1$ 到 i 的划分方案唯一。因为 $i-1$ 到 j 这一段需要含有 1，所以这一段有一个长度下限，并且结合 $l-1$ 到 i 的划分方案可以得到这些方案中最长上取整段的下限，于是我们之后只需要统计更长的答案。

如何去计算 $i-1$ 到 1 的答案，我们考虑去改造之前的容斥。这里其实只多了一个限制：第一段是长度不低于一

个下限的上取整段。于是我们在容斥的过程中将第一段特殊处理即可，同时还要分两种情况：第一段作为被钦定段，第一段不作为被钦定段。

具体推导是大同小异的，就不展开叙述，我们假定它形如：

$$\sum_{j=1}\sum_{k=1}^{i-jk\geq 0}\binom{i-jk}{k}2^{i-jk-k}\times(-1)^k$$

实际得到的式子与上述有差别，不过经过一些微小的处理可以认为是类似的形式。

对于单个 i 复杂度依然是 $O(i\log i)$，总复杂度为 $O(l^2\log l)$。

一般情况计数优化

注意到有个关键条件是 $i-jk\geq 0$，于是我们可以考虑根号分治，分为 $j\leq\sqrt{l}$ 和 $k\leq\sqrt{l}$ 且 $j>\sqrt{l}$。

先看 $k\leq\sqrt{l}$ 的情况。我们对于每个 k 都维护一个 $O(l)$ 的数组 $f_i=\sum_{j=1}\binom{i-jk}{k}2^{i-jk-k}\times(-1)^k$，有递推式 $f_i=f_{i-k}+\binom{i-k}{k}2^{i-k-k}\times(-1)^k$。

再看 $j\leq\sqrt{l}$ 的情况。维护 $g_i=\sum_{k=1}\binom{i-jk}{k}2^{i-jk-k}\times(-1)^k$，有递推式 $g_i=2g_{i-1}-g_{i-1-j}+\binom{i-1-j}{0}2^{i-j-1}\times(-1)^1$，只需要用 $\binom{n}{m}=\binom{n-1}{m-1}+\binom{n-1}{m}$ 展开即可推出。

自此，可以在 $O(l\sqrt{l})$ 的复杂度下解决这一问题。

题外话

本题题面中问题的复杂度，也即本题的答案，根据 oeis.org[1] 中给出的数据估算大致为 $O(n\log\log n)$ 量级。

参考资料

[1] OEIS A119706, https://oeis.org/A119706