最大异或和 解题报告

杭州学军中学 金策

1 试题来源

2015年集训队互测

2 试题大意

维护一个长度为n的数列$a_1, a_2, \cdots, a_n$，其中每个a_i都是m位二进制数。操作数量为q，类型包括以下三种：

- 1 x y w 对于所有$x \le i \le y$，将a_i修改为$a_i \operatorname{xor} w$；
- 2 x y w 对于所有$x \le i \le y$，将a_i修改为w；
- 3 询问$a_1, a_2, \cdots, a_n$的最大子集xor和。

数据规模：$1 \le n, m, q \le 2000$。

3 算法介绍

3.1 算法一

操作1和操作2可以$O(nm)$暴力实现。

对于每个询问，枚举所有2^n个子集并计算异或和，求出最大的即可。

每次询问的复杂度是$O(2^n nm)$或$O(2^n m)$，可以通过10%的数据。

3.2 算法二

将所维护的数列视为$n \times m$的01矩阵。

每次询问时对矩阵做一遍高斯消元，得到一个行阶梯形矩阵。复杂度是$O(n^2m)$。

然后按照从高位到低位的顺序枚举主元，并贪心地选取行即可。这一步复杂度是$O(nm)$。

于是询问一次的复杂度是$O(n^2m)$，可以通过30%的数据。

3.3 算法三

在算法二的基础上，用bitset存储矩阵的每一行。在进行消元等操作时，可以使用bitset的^运算进行加速。

这样，每次修改是$O(nm)$，每次询问仍然是$O(n^2m)$，但常数都变为了原来的$1/32$，可以通过40%的数据。

3.4 算法四

现在整个算法的瓶颈在于询问时第一步的消元过程。为了优化这个过程，我们需要在每次修改操作后快速更新消元得到的阶梯形矩阵。

设原矩阵为A，其中A的第i行为a_i的二进制表示。设经过消元得到的行阶梯矩阵为B。同时，我们把消元的过程记录下来，用可逆矩阵P表示，即$PA = B$。

先来考虑修改单个a_i的情况：

假设我们将a_i改为$a_i \operatorname{xor} w$，则A被修改为$A' = A + e_i w$。于是B变为$B' = PA' = B + Pe_i w$。即对于所有$P_{ji} = 1$的j，B的第j行都被异或了w。令其中最大的$j = j_{\max}$，即第$j_{\max}$行是所有被异或的行中最靠下方的一个，我们用这一行去消剩下所有被异或的行，设其为j行，那么对应的B_j会变成$(B_j \operatorname{xor} w) \operatorname{xor} (B_{j_{\max}} \operatorname{xor} w) = B_j \operatorname{xor} B_{j_{\max}}$，从而$w$对第$j$行的影响就被抵消了。又因为在行梯形矩阵中，$j_{\max}$在$j$的下方，所以第$j$行的主元不会被影响。

这样一轮操作过后，第$j_{\max}$行的主元可能保持不变，也可能会变成w的最高位。如果是后者，则还需检查原本这一列上有没有主元，若有的话还需要对其对应的那一行进行一次消元。

运用上述方法，可以在每次修改单个a_i之后，用$O(nm)$时间更新阶梯形矩阵B（以及对应的矩阵P)。可以通过50%的数据。

3.5 算法五

算法四中，我们实现了将单个a_i异或上w。这一方法可以容易地推广到多个a_i的情况：

设被异或的行的编号组成的集合是S，令列向量$s = \sum_{i \in S} e_i$。

于是，A被修改为$A' = A + sw$时，B变成了$B' = PA' = B + Psw$。只要将列向量Ps计算出来（这可以用bitset的&运算和count()函数），就可以同算法四一样处理了。可以通过60%的数据。

3.6 算法六

现在只需要解决操作2。

首先由异或运算的性质可知，对于若干个相同的a_i，我们只需要保留其中一个，而不会影响答案。

由此，我们把序列$\{a_i\}$分成若干个连续段，每个连续段中的a_i相同，只在矩阵中存一份即可。

每次操作2时，我们把涉及到的连续段一个个从矩阵中删除，然后合并这些段，并在矩阵中插入连续一段w。由于每次区间操作后，连续段的总数至多增加2，容易证明这样的方法每次只要做均摊$O(1)$次矩阵操作，复杂度为均摊$O(nm)$。

因此，本题时间复杂度为$O\left((n+q)nm\right)$，空间复杂度$O(nm)$，用bitset优化后常数均为原来的1/32，可以通过100%的数据。

4 总结

本题是一道NOI中等偏下难度的题目，主要考察对高斯消元的基本理解和利用位运算优化程序的技巧。预计集训队互测中绝大多数选手能拿到40分及以上，约有5名选手可以拿到满分或接近满分。