Day1 讲题

讲题人：xjt

2024 年 1 月 6 日

吐槽环节

简短题意

给定一个具有 $2n$ 点的二分图，边权为 01，试构造一个边权异或和为 0 的完美匹配。

$n \leq 500$

$n \leq 8$

读懂题意即可。

直接对每个左边点枚举右边匹配的点是哪个，暴力检验所有匹配。

复杂度 $O(n^n)$。

$n \leq 18$

点的匹配关系其实不重要，只需要关心剩下能匹配的点集以及当前边权异或和即可。

可记 $dp_{i,S,0/1}$ 表示考虑了前 i 个左边点，匹配了右边点集为 S，且当前边权异或和的值为 $0/1$ 是否可能。

转移枚举 i 的匹配边即可。

复杂度 $O(n^2 2^n)$，期望得分 $40pts$。

性质分析

先求任一个完美匹配 X。

如果当前匹配边权异或和已经为 0，则直接输出即可。

否则考虑最终答案 Y，若 X 和 Y 取对称差，则会得到若干个偶环，并且其中一个偶环边权异或和恰好为 1。

不妨假设 X 就是 i 匹配 $i+n$ 。

那么对于一个长度为 $2k$ 的环 $x_1, x_1+n, x_2, x_2+n, \cdots, x_k, x_k+n$，可以把它缩成一个长度为 k 的环 $x_1, x_2, \cdots, x_k$ ，每条边的边权是原来两条边的异或和。

即对于原图的 $(y, x+n)$ ，在新图里连边 $x \to y$ ，边权为 $(x, x+n), (x+n, y)$ 的异或和。

则我们的目标就是在新图里找一个边权异或和为 1 的环。

在边权随机的情况下，可以随机 dfs 并只考虑返祖边导出的环（实际上非常难卡也许能得到 100 pts）

正确做法可以通过拆点求 $(x, 0)$ 到 $(x, 1)$ 的路径得到。

复杂度瓶颈为求完美匹配，即 $O(n^3)$。

简短题意

在数组 $a[1...n]$ 上定义一个操作序列 $op[1...m]$，每个操作形如：

- C：区间 $a[l...r]$ 赋值为 v
- Q：区间 $a[l...r]$ 求和

支持多组询问，每组询问：

- 依次 $op[L...R]$ 的所有操作，求所有求和操作的结果总和

$n \leq 5 \times 10^5$

$n, m, q \leq 100/500$

读懂题意即可。

对每个询问按顺序执行操作，每次操作可暴力 $O(n)$ 或使用数据结构 $O(\log n)$ 加速。

总复杂度 $O(nmq)/O(mq \log n)$。

每次操作都是求和操作

因为 a 初始化后没有修改恒为 0，所以求和一定为 0。

$L = 1$

相当于对一个前缀求答案。

从左往右执行操作即可，需要使用数据结构 $O(\log n)$ 加速。

总复杂度 $O(q \log n)$。

满分做法：题意简化

不妨考虑一个简单的版本：把区间赋值 C 改为区间加法 A。

考虑对操作序列按 $B = \sqrt{n}$ 分块。

假设我们能够对任意 i, R 预处理出 $op[iB, R]$ 的答案。

则询问 $op[L, R]$ 的时候，我们可以将贡献分成三部分：

- $op[L, iB)$ 的 A 对 $op[L, iB)$ 的 Q 的贡献；
- $op[L, iB)$ 的 A 对 $op[iB, R]$ 的 Q 的贡献；
- $op[iB, R]$ 的 A 对 $op[iB, R]$ 的 Q 的贡献；

显然第三部分已经预处理，考虑前面两部分。

题意简化解法

第一部分：$op[L, iB)$ 的 A 对 $op[L, iB)$ 的 Q 的贡献

该部分可以看成是一个后缀 $op[*, iB)$ 的贡献，也可以预处理。

在后缀移动的时候直接对每个操作 A 暴力检验对其后面 Q 的贡献即可。

该部分总复杂度 $O(n\sqrt{n})$。

题意简化解法

第二部分：$op[L, iB)$ 的 A 对 $op[iB, R]$ 的 Q 的贡献

注意到

- $op[L, iB)$ 的 A 会将 $a[1...n]$ 划分成不超过 $O(B)$ 个连续段

在从小到大枚举 R 时可以通过差分标记求得 $op[iB, R]$ 对 $O(B)$ 的每个连续段共求和过多少次。

询问的时候可暴力 $O(B)$ 还原差分标记，并 $O(B)$ 枚举 $op[L, iB)$ 的 A 求得贡献。

题意简化解法

预处理第三部分：$op[iB, R]$ 的 A 对 $op[iB, R]$ 的 Q 的贡献

实际上该部分可以拆成：

- $op[iB, (i+1)B)$ 的 A 对 $op[iB, (i+1)B)$ 的 Q 的贡献；
- $op[(i+1)B, R]$ 的 A 对 $op[(i+1)B, R]$ 的 Q 的贡献；
- $op[iB, (i+1)B)$ 的 A 对 $op[(i+1)B, R]$ 的 Q 的贡献；

第一个与之前讨论的询问没有本质区别；

第二个可直接继承后一块的答案；

对于第三个，可以使用 $O(\sqrt{n})$ 修改，$O(1)$ 查询的值域分块，总复杂度 $O(n\sqrt{n})$。

对于零散的部分，也即询问的 $[L,R)\in[iB,(i+1)B)$ 的情况，可以视为

- $op[L,(i+1)B)$ 的 A 对 $op[L,(i+1)B)$ 的 Q 的贡献；
- 减去 $op[R,(i+1)B)$ 的 A 对 $op(R,(i+1)B)$ 的 Q 的贡献；
- 减去 $op[L,R)$ 的 A 对 $op[R,(i+1)B)$ 的 Q 的贡献；

前两部分已经通过后缀贡献预处理得到，对于第三部分：

- 对每个 $op[R,(i+1)B)$ 打差分标记
- 对 $op[L,R)$ 的 A 暴力枚举贡献即可

原题意转化

回到原题意，考虑操作都是同一个区间的情况，设在时间 t 赋值为 v：

- 若该操作为第一次赋值，则视为在时间 t 将值加等于 v；
- 若其上一次赋值为 (t', v')，则视为在时间 $[t', t]$ 将值减等于 v'。

上述操作均可抽象成操作序列轴上的 (tl, tr, v)，当且仅当询问 $[L, R]$ 包含 $[tl, tr]$ 的时候，该加法操作才生效。

- 不是同一个区间的情况，可以通过颜色数均摊的方法转化成 $O(m)$ 个等价的相同区间。

满分做法

转化后与简化版本唯一的不同是区间加法操作 A 是有生效范围限制的，即

- 令 $A = (tl, tr, l, r, v)$，当且仅当询问 $[L, R]$ 包含 $[tl, tr]$ 的时候对 $a[l, r] += v$

考虑每个部分需要怎么适配。

预处理第三部分：$op[iB, R]$ 的 A 对 $op[iB, R]$ 的 Q 的贡献

该部分没有变化，注意 $op[iB, R]$ 的 A 指 $[tl, tr] \in [iB, R]$ 即可。

第二部分：$op[L, iB)$ 的 A 对 $op[iB, R]$ 的 Q 的贡献

- $op[iB, R]$ 的差分标记这部分不变。
- 在 $O(B)$ 暴力枚举 A 的时候注意只需要枚举满足 $[tl, tr] \in [L, R]$ 的 A。

第一部分：$op[L, iB)$ 的 A 对 $op[L, iB)$ 的 Q 的贡献

该部分由于 R 未知，所以每个 A 是否生效未知，不能直接简单预处理每个后缀的答案。

但由于 $R \geq iB$，所有 Q 都是生效的，所以对每个 $A = op[j]$ 求得其对后所有 $Q \in op(j, iB)$ 的贡献。

然后再 $O(B)$ 暴力枚举 A 检验是否生效即可。

零散部分：$op[L,R) \in op[iB,(i+1)B)$

由于预处理后缀的方法已经不可行了，故考虑另一个方法：

- 容：先假设所有 $A \in op[L,R)$ 都在 $Q \in op[iB,R)$ 前面
- 斥：然后扣掉额外多统计的 $Q \in op[iB,j)$ 对 $A = op[j]$ 的贡献

通过容斥之后就转化成了前缀 Q 和前缀 $Q+A$ 的贡献，通过差分标记预处理，每个询问可以预处理 $O(\sqrt{n})$ 求得。

综上所述，总复杂度 $O(n\sqrt{n})$。

如某个部分没有平衡好，$O(n\sqrt{n}\log n)$ 能得到 $60pts - 80pts$。

简短题意

定义数组 $a[1...n]$ 的 $f(a) = b[1...n]$ 为 b_i 表示把 a_i 改为 0 后 $a[1...n]$ 的最大独立集和。

求 $a_i \in [0, m]$ 能得到多少种本质不同的 $f(a)$。

$n \leq 3000$

$n, m \leq 5$

读懂题意即可。

暴力枚举所有可能的 a 求出 $f(a)$ 后去重即可。

总复杂度 $O(m^n)$。

推性质

设 $i \le 0$ 或 $i > n$ 时令 b_i 为全局最大独立集。显然 $n \ge 2$ 时 $b[1 \dots n]$ 中的最大值即为全局最大独立集。

- 令 f_i, g_i 分别表示 $a[1, i], a[i, n]$ 的最大独立集。则 $b_i = f_{i-1} + g_{i+1}$。
- 令 $p_i = f_i - f_{i-1}, q_i = g_{i+1} - g_{i+2}$。则 $b_{i+1} - b_i = p_i - q_i$。

根据 f_i, g_i 的递推式可以得到 $p_i = \max(a_i - p_{i-1}, 0), q_{i-1} = \max(a_i - q_i, 0)$。

若我们固定了所有的 $c_i = b_{i+1} - b_i$，则合法的 b 应当满足其全局最大值在一段区间中，也即 f_n 在一段区间中，因此我们只需要考虑 $f_n = \sum p_i$ 的上下界。

推性质（续）

先考虑对于一组确定的 $c_{1\sim n}$ 如何确定 f_n 的上下界。

容易发现 b_i 中任意相邻两个位置中至少有一个最大值，因此若 $c_{i-1} < 0$ 则 $c_i = -c_{i-1}$。

假设我们已经确定了 $a_{0\ldots i-1}, p_{0\ldots i-1}, q_{0\ldots i-1}$，再固定 a_i，则

- p_i 可以唯一确定。
- 若 $q_{i-1} > 0$ 则 q_i 也可以唯一确定。
- 否则 $q_{i-1} = 0$，此时 q_i 可以是任意一个 $\geq a_i$ 的数。

推性质（续）

具体来说，有如下若干种情况：

- 若 $c_{i-1} < 0$，则此时一定有 $c_i = -c_{i-1}$，可以任选 $a_i \in [q_{i-1}, m]$，有 $p_i = a_i - p_{i-1}, q_i = a_i - q_{i-1}$。
- 若 $c_{i-1} \geq 0$，则此时一定有 $c_i \leq 0$：
- 若 $|c_i| < |c_{i-1}|$，则有 $a_i = q_{i-1} + |c_{i-1}|, p_i = 0, q_i = |c_i|$。
- 若 $|c_i| > |c_{i-1}|$，则要求 $q_{i-1} = 0$，可以任选 $q_i \in [c_i, m]$，有 $a_i = q_i - |c_i| + |c_{i-1}|, p_i = q_i - |c_i|$。
- 若 $|c_i| = |c_{i-1}|$，则可以任意归入上述两种情况之一。

根据上述过程可以发现 f_n 可以取到的最小值为 $\frac{1}{2}\sum |c_i|$，这是因为只需要始终保持 $\min(p_i, q_i) = 0$ 一定合法且最优。所以我们只需求出 f_n 可以取到的最大值就可以了。

部分分

因为 $f_n = \sum_{i=0}^{n-1} p_i$，并且只有相邻的 a, p, q 的取值有影响；

考虑 $dp_{i,x,y}$ 表示 $p_i = x$，$q_i = y$ 的时候，$\sum p$ 最大可以是多少。

然后可以将对着 dp 写出 dp 套 dp 就可以解决原问题。

复杂度大约是 $O(nm^{m^2})$，实现难度不低于 NOI2022 移除石子。

满分做法

可以发现，只需要在不违反限制的情况下贪心地使得当前的 p_i 最大即可使得 f_n 最大。

（限制只会在 $c_{i-1} \geq 0, |c_i| > |c_{i-1}|$ 时产生）。

证明可以考虑调整法，即考虑当前 p_i 未取到上限的情况，将它与 p_i 取到上限的情况比对，后者一定不劣。

因此我们只需要知道 $p_{i-1}, q_{i-1}, c_{i-1\ldots i+1}$ 即可唯一确定 a_i, p_i, q_i 在最优贪心过程中的取值。

$dp_{i,x,y,c1,c2,\sum p}$ 表示考虑到当前想确定 p_i 和 q_i 的值，p_{i-1} 和 q_{i-1} 的值分别是 x, y，c_{i-1} 和 c_i 分别是 $c1, c2$，前面 p 的总和是 $\sum p$。

再枚举 c_{i+1} 是多少那么根据上述结论可以推出 p_i 与 q_i 的取值。

不加以优化的话时间复杂度是 $O(n^2m^6)$，期望得分 $50pts$。

实际上 $\sum p$ 不需要记在状态上，令

- $f_{i,x,y,c1,c2}$ 当前状态为 $i,x,y,c1,c2$ 的方案数。
- $g_{i,x,y,c1,c2}$ 当前状态为 $i,x,y,c1,c2$ 的 $\sum p - \frac{1}{2}\sum |c_i|$ 之和。

复杂度可优化至 $O(nm^5)$，期望得分 $50 \sim 85pts$。

而我们基本只关心 $c_{i-1\ldots i+1}$ 之间的大小关系。

$c_{i-1\ldots i+1}$ 的具体数值只在 a, p, q 的相互转移中用到。

考虑只记录 c_{i-1}, c_i 的大小关系并使用前缀和优化 dp 即可做到 $O(nm^2)$。

实际上，分析贪心过程的性质可得一定有 $\min(p_i, q_i) = 0$ 或 $\max(p_i, q_i) = m$。

也即总状态数只有 $O(nm)$，总复杂度 $O(nm)$。

没有前缀和加速或者没有意识到 p, q 之间的关联，复杂度为 $O(nm^2)$，可以获得 $70 \sim 85$ 分。

提问 + 吐槽环节 $\times 2$