PTZ winter 2020 Day3 C

考虑我们找到一条尽可能靠左下（即，可以向下就向下，否则向右）的 $(1,1)$ 到 (n,m) 的路径 U 和一条尽可能靠右上的路径 V，那么假如我们只能 ban 一个位置，我们需要的就是把一个同时出现在两条路径上的位置 ban 了，相当于求两条路径的交集大小。

现在我们可以 ban 两个位置，假如我们其中一个 ban 了同时出现在 U,V 的位置，那么另一个可以随便放一个，先单独算这种情况。接下来，我们考虑 ban 掉只在 U 上的位置，这样我们求出新的尽可能靠左下的路径 U'，那么另一个石头的选择数就是 U',V 交集大小。

考虑我们先找出所有可以被 $(1,1)$ 到达的且可以到达 (n,m) 的点，假如我们把 (x,y) ban 了，我们找到 $x'+y'=x+y$ 且 y' 最小，则 U' 一定是经过 (x',y') 的尽可能靠左下的路径。接下来不难 $O((n+m)^2)$ 或 $O(nm)$ 得到答案。

PTZ summer 2021 Day4 B

考虑一个孤立点复制之后会变成一个匹配，可以操作变成两个孤立点。所以我们可以考虑先把能操作的操作了。

考虑最后每个点都是偶度数的，那么每条边都在一个环上。考虑那些拥有至多一个属于超过两个环的点的环，假设它是 $p_0\to p_1\to\cdots\to p_m\to p_0$（当然它是无向图，我们为了方便只是随意给环定了一个向而已），其中 p_0 是那个属于超过两个环的点（假设不存在就随便取一个）。考虑 p_i 的复制点为 p'_i，那么我们依次操作 $p_1,p'_2,p_3,p'_4\ldots$，然后操作 p'_1，当 $2\mid m$，操作 p_m，否则是 p'_m。那么容易发现，原来环上的边和复制的边都被删去，对于 $1\le i\le m$，(p_i,p'_i) 也被删去，而其他边都没有影响。那么我们每次就会把一个环剥离，直到整张图都没有边，显然 0 是最小值。

复杂度 $O(n+m)$。

CF843E

考虑到，在最小割上的边一定是满流的，其他边一定可以通过调整边权来不满流。此时最小割的边数就是答案。

如何求出一组边数最少的最小割呢？考虑最小割就是最大流，我们考虑此时一定不能有 S 到 T 的增广路。

那么对于 $g_i=0$，我们连一条 $(u_i,v_i,\inf)$，表示因为这条边没满流 u_i 一定可以走到 v_i。

对于 $g_i=1$，我们连边 $(u_i,v_i,1),(v_i,u_i,\inf)$，表示因为该边有流量，所以反向边一定有残余容量，同时，假设该边没满流，那么 u_i 可以到 v_i，否则我们可以选择把它加入最小割并认为它满流了。

此时跑一个 S 到 T 的最小割即可。

然后考虑构造方案。跑一个上下界网络流得到一组使所有要有流的边都有流的方案，此时每条边的最终流量就是它此时流量（注意此时下界为 1，不要忘记加上）。然后假如它不在最小割集内，则设容量为 $\inf$，否则设为它此时流量，即为它满流了。

复杂度 $O(n^2m)$。

AGC012E

考虑到整个过程中，V 只有 $O(\log)$ 个不同的取值，对于每个取值，我们可以处理出所有可以互相到达的连续段。

那么我们此时就可以处理出 f_S，表示使用了 S 中那些 V 的取值，可以走到的最长前缀长度。同样的，处理出 g_S 表示最长后缀长度。

设除掉一开始的 V 之外剩下 V 的取值集合为 U，那么我们枚举 $S \subseteq U$，那么假如 $f_S + g_T \geq n$，那么全部绿洲都是合法的。否则，假如 $[f_S + 1, n - g_T]$ 这些点可以用初始的 V 互相到达，那么这些点所属的连续段都是合法的。

复杂度 $O(V \log V + n)$。

PTZ summer 2021 Day5 D

考虑离散化后，数轴被我们划分成若干段不交部分，那么我们对每个部分维护它的贡献，假如 $i_1, i_2, \ldots, i_m$ 这些区间包含这个部分。设 $f_{l,r} = |\bigcup_{i=l}^{r} [L_i, R_i]|$。那么相当于对于每个 j，我们对所有 $l \leq i_j \leq r$，对 $f_{l,r}$ 加上这个区间的长度，然后对于每个 j，我们对所有 $l \leq i_j, i_{j+1} \leq r$，对 $f_{l,r}$ 减去这个区间的长度。

那么每次询问都是矩阵求和，假如我们得到了所有的矩阵加操作，我们就可以扫描线解决该题。

对于第一种操作，和起来就相当于对 $l \leq i \leq r$，对 $f_{l,r}$ 加上 $R_i - L_i$。

对于第二种操作，我们同样可以扫描线，我们会在某个时刻加入一个区间，然后在某个时刻删去一个区间。那么我们加入一个区间时，可能插入在某个 (i_j, i_{j+1}) 中间，那么维护这个对出现的时间，然后就可以维护这个对贡献的操作。删去的时候同理。观察这个过程发现只会有 $O(n)$ 次矩阵加。

复杂度 $O((n + q) \log n)$。

QOJ#7881

当 $n \geq 13$，$f(n), g(n)$ 式子中对 n 取 $\max$ 的部分可以删去。

考虑 $f'(n) = f(n) - f(n-1), g'(n) = g(n) - g(n-1)$，则当 $n \geq 14$：

$$f'(n) = [2|n]g'(n/2) + [3|n]g'(n/3) + [5|n]g'(n/5) + [7|n]g'(n/7)$$
$$g'(n) = [2|n]f'(n/2) + [3|n]f'(n/3) + [4|n]f'(n/4) + [5|n]f'(n/5)$$

考虑 $n = xk$，其中 $\gcd(2 \times 3 \times 5 \times 7, x) = 1$，可以归纳证明，假如 $x \geq 14$，则 $f'(n) = g'(n) = 0$。

所以对于 f, g 的差分，只有 $O(\log^4)$ 个位置有值，分别求出，每次询问时二分查询前缀和即可，复杂度 $O(\log^4 V + T \log(\log^4 V))$，其中 $V = 10^{15}$。

PTZ winter 2020 Day3 G

考虑我们随意选择一个生成树进行点分治。此时注意到由于包含分治中心的环至多 k 个，所以跨越不同子树的非树边也至多 k 条，对于每个这样的边，我们都分别选出一个端点，我们称这些点以及分治中心是**有用**的。假如我们把路径跨越**有用**点的情况处理了，那么删掉**有用**点后仍两个不同子树之间没有连边，可以分治下去。

对于询问和修改，我们把它们下放到点分治过程中，每个操作只会被考虑 $O(\log n)$ 次。在分治中，因为我们只需考虑跨越**有用**点的路径，所以我们可以预先从每个**有用**点开始 bfs 求出到每个点的距离，然后维护距离每个**有用**点最近的标记点即可。

复杂度 $O((n + q)k \log n + m\alpha(n))$。

QOJ#7788

假如存在一行没有车，那么必然每列都有车，我们对于每列都分别二分，用没有车那一行作为指示位即可 $\lceil \log_2 n \rceil$ 次获得 n 个车的位置。存在一列没车同理。

那么我们先询问 $\{(i,1),(i,i)\mid 2\le i\le n\}$，此时若 $c_{1,1}=0$，则第 1 行无车，用上述方法解决。若 $c_{i,i}=1$ 则 $(i,1),(i,i)$ 中必有一车，否则 $(i,1),(i,i)$ 都没车。

若不存在 i 使得 $c_{i,i}=0$，则询问 $\{(i,i)\mid 1\le i\le n\}$，则此时对于 $i\ge 2$，每行都找到了一个车，同时我们直到第 1 行有车，用 $\lceil\log_2 n\rceil$ 次二分即可。

否则，我们询问 $\{(i,i),(1,i)\mid c_{i,i}=0\}\cup\{(i,1)\mid c_{i,i}=1\}$，则若新的 $c'_{i,1}=1\wedge c_{i,i}=0$，则第 i 行无车，用上述方法解决。若 $c'_{i,i}=1\wedge c_{i,i}=0\wedge i\ge 2$，则 $(1,i)$ 必有车，若这样的 i 不存在，且 $c'_{1,i}=1\wedge c_{i,i}=0\wedge i\ge 2$ 存在，则 $(1,1)$ 必有车，否则 $(1,1)$ 必没车。

此时，对于每一行，要么已知车存在于 $(i,1),(i,i)$ 之一，要么已知该行有车且 $(i,1)$ 定无车。

对于后者，我们可以使用类似一行没有车的做法。同时，在二分的过程中，我们可以把满足前者的那些行在主对角线的元素插入在二分时的询问中，实现仔细的话，可以在 $\lceil\log_2 n\rceil$ 次询问中把每行的车的位置全都确定。

总询问次数 $2+\lceil\log_2 n\rceil$。

CF1540E

首先注意到，假如 i 可以复制 j，并且 j 在第 d_j 天后 $a_j>0$，那么在 d_j+1 天 $a_i>0$。所以我们可以先在 $O(n^2)$ 内算出来 d_i。

发现 d_i 只会更改 $O(n)$ 次，我们每次暴力算即可。

考虑每个点对询问的贡献，设 $A_{i,j}=j[i\ can\ copy\ from\ j\vee i=j]$，$e_i$ 为一个长度为 n 的列向量，且 $e_{i,j}=[j=i]$。

那么答案即为：$(\sum\limits_{i=l}^{r}e_i^T)(\sum\limits_{d_i\le k}A^{k-d_i}e_ia_i)+\sum\limits_{d_i>k}a_i$

发现 A 是一个上三角矩阵，有 n 个特征根分别为 $1,\dots,n$。我们考虑求特征向量，设 i 对应的特征向量为 v_i。

那么 $A\times[v_1,\dots,v_n]=[v_1,2v_2,\dots,nv_n]$。设 $v_{i,i}=1$，那么 $v_{i,j}$ 可以大力解方程得出。

发现 v 形如一个下三角矩阵，所以 v_i 之间线性无关，所以我们可以把 e_i 分解成 $\sum c_{i,j}v_j$ 的形式。

其中 $vc^T=I$，暴力求逆即可获得 c。

那么原式变成了 $(\sum\limits_{i=l}^{r}e_i^T)(\sum\limits_{d_i\le k}A^{k-d_i}\sum c_{i,j}v_ja_i)$。考虑 $Av_i=iv_i$，所以原式变成

$\sum a_i\sum j^{k-d_i}c_{i,j}\sum\limits_{p=l}^{r}v_{j,p}$。

对于每个 k 维护 $d_i\le k$ 位置上的 $S_j=\sum a_ic_{i,j}j^{-d_i}$，剩下的都可以预处理。这一步可以使用树状数组。

重构复杂度为 $O(n^2)$，修改和询问都是 $O(n\log n)$，所以总复杂度为 $O(n^3+qn\log n)$。