湖南省队集训 Day5

题解

下落的数字（fall）

$n, m \leq 5000$

直接模拟即可。

无修改

注意到到达任意一个点的数字都是一段区间，把每个点对应的区间简单维护出来即可。

无特殊限制

暴力模拟复杂度太高，我们考虑跳重链来优化复杂度。

每个点走到它的重儿子的条件都是一段区间，一个数字能沿一段重链一直走下去的条件是这个数字在这条重链上的所有区间的交中。

对每一条重链建线段树维护区间交，然后线段树上二分即可找到这个数字走到重链上的那个点，然后再跳轻儿子即可。

修改也很容易实现。复杂度 $O(n \log^2 n)$。

也可以用 LCT 实现 $O(n \log n)$ 的复杂度，但具体实现比较麻烦，这里不细讲。

序排速快（tros）

$R \le 10^7$

考虑算每个位置对答案的贡献，其实就是它递归下去的次数。

我们重新定义一下分割点的概念, 第 i 个元素和第 $i+1$ 个元素之间的那条线叫做分割点 i。先给出结论, 一个位置停止递归当且仅当他前面与后面的分割点都已经出现, 即对于位置 i , 分割点 i 和分割点 $i-1$ 均已出现, 不妨设第 i 个分割点出现的时间为 t_i 那么每一个位置 i 对于答案的贡献次数就是 $\max\{t_{i-1}, t_i\}$。

那么现在我们就要考虑这个 t_i 要如何计算, 对于一个分割点 i, 它出现的时间就是所有 $1 \sim i$ 的数都在 $[1, i]$ 之间的第一个时刻。观察每次冒泡的过程，设所有小于等于 i 的数的最大的位置为 p_i，那么在分割点 i 出现之前每次冒泡后 p_i 必然会减少 1，所以 $t_i = p_i - i$。

每个位置加起来，对于一个确定的排列的答案就是（p 的意义和上面一样）:

$$\sum_{i=1}^{n} \max\{p_{i-1} - (i-1), p_i - i\}$$

然后分别考虑 $t_{i-1} \le t_i, t_{i-1} > t_i$，两种情况的式子分别为:

$$\sum_{i=1}^{n}\sum_{j=i}^{n}\binom{n-i}{n-j}(j-1)!(n-j)!(j-i)!$$

$$\sum_{i=1}^{n}\sum_{j=i}^{n}(i-1)\binom{n-i}{n-j}(j-1)!(n-j)!(j-i+1)!$$

（推的方法就是对于 i 考虑数 i 放在哪个位置，然后 j 是枚举的 p_i/p_{i-1}）

两个式子加起来就是

$$\sum_{i=1}^{n}\sum_{j=i}^{n}(j-1)!(n-j)!\binom{n-i}{n-j}(ij - i^2 + i - 1)$$

以有没有 j 分类分别推出

$$\sum_{i=1}^{n}\sum_{j=i}^{n}(j-1)!(n-j)!\binom{n-i}{n-j}(-i^2 + i - 1) = \sum_{i=1}^{n}(-i^2 + i - 1)(n-i)!(i-1)!\binom{n}{i}$$

$$\sum_{i=1}^{n}\sum_{j=i}^{n}(j-1)!(n-j)!\binom{n-i}{n-j}ij = \sum_{i=1}^{n}i(i!)(n-i)!\binom{n+1}{i+1}$$

最后把组合数用阶乘表示，发现预处理的东西只和 i 有关，直接预处理前缀和即可，时间复杂度 $\mathcal{O}(R)$。

树（tree）

$n \le 7$

直接把所有树全部存下来，记忆化搜索即可。

时间复杂度 $\mathcal{O}(n^{n-2}\text{poly}(n))$。

$n \le 15$

答案就是 `Yes`，证明可以考虑归纳。

假设当前 $T_1 \cap T_2 = T_3$，取 T_3 中最大的一个连通块为 T。

当 $|T| = n$ 时，不需要操作。

当 $|T| = n-1$ 时，直接旋转不在 T 中的那个点就可以了，显然满足条件。

否则考虑找到两条边 $e_1 = (x_1, y_1), e_2 = (x_2, y_2)$ 满足 $y_1, y_2 \in T, x_1, x_2 \notin T, T \cup \{e_1\} \subseteq T_1, T \cup \{e_2\} \subseteq T_2$ 且 $e_1 \ne e_2$（肯定是可以找到的，不然 T_3 中最大的那棵树就不是 T 了）

如果 $x_1 \ne x_2$，那么令 T' 为包含 $T \cup \{e_1\} \cup \{e_2\}$ 的一棵原图的生成树，由归纳假设，T_1 可以变成 T'，T' 可以变成 T_2。

否则，因为原图是点双连通图，所以一定可以再找到一条边 $e_3 = (x_3, y_3)$ 满足 $x_3 \ne x_1, x_3 \notin T, y_3 \in T$，令 T' 为包含 $T \cup \{e_1\} \cup \{e_3\}$ 的生成树，T'' 为包含 $T \cup \{e_2\} \cup \{e_3\}$ 的生成树，同理，T_1 可以变成 T'，T' 可以变成 T''，T'' 可以变成 T_2，证明完毕。

构造的话直接按照证明模拟就可以了。

$n \le 100$

随便选择一个点作为根，考虑按照 T_1 的 dfn 从大到小依次考虑，找到第一个不满足其在 T_1 上的父亲和在 T_2 上的父亲相等的点 x，可以发现其 T_2 上的父亲不在 T_1 中这个点的子树内。

考虑定义过程 $\text{solve}(x)$ 表示将 x 的所有子节点全部移出 x 的子树，那么只需要调用 $\text{solve}(x)$ 即可令 x 变为叶子节点，然后再做一次操作就可以令 x 的父亲是它在 T_2 上的父亲，然后倒序将之前做的操作撤回即可变回原样。

考虑如何构造 $\text{solve}(x)$。首先找到 x 在 T_1 子树中的一个点 y 满足其存在一条边使得 y 和子树外某个点连通（由于原图是点双所以一定存在这样一个点），递归调用 $\text{solve}(y)$ 后用一次操作即可将 y 移出 x 的子树，不断迭代这个过程直到 x 成为叶子节点为止。

可以毛估估一下发现复杂度上界是 $\mathcal{O}(2^n)$，但是出题人并不能造出使该算法操作次数超过 $\mathcal{O}(n^2)$ 的数据，如果影响了大家的思考的话出题人感到非常的抱歉。如果有同学能卡掉该算法（或者证明复杂度正确）的话欢迎联系出题人。