树上问题

花花

IIIS@THU

December 17, 2024

碎碎念

第一，树结构在 OI 中经常出现，而且可以和许多东西结合，基本上没有什么特定的思路。所以基本上是杂题选讲……

碎碎念

第一，树结构在 OI 中经常出现，而且可以和许多东西结合，基本上没有什么特定的思路。所以基本上是杂题选讲……

第二，即便如此，我还是把题目分到了几个类别中，方便大家体会一类题目的那股劲！

碎碎念

第一，树结构在 OI 中经常出现，而且可以和许多东西结合，基本上没有什么特定的思路。所以基本上是杂题选讲……

第二，即便如此，我还是把题目分到了几个类别中，方便大家体会一类题目的那股劲！

第三，由于按照类别来讲，因此题目不按照难度排序，所以如果你掉线，下个题的时候记得重连一下。

碎碎念

第一，树结构在 OI 中经常出现，而且可以和许多东西结合，基本上没有什么特定的思路。所以基本上是杂题选讲……

第二，即便如此，我还是把题目分到了几个类别中，方便大家体会一类题目的那股劲！

第三，由于按照类别来讲，因此题目不按照难度排序，所以如果你掉线，下个题的时候记得重连一下。

第四，如果会了一个题或者有任何思路，都可以举手上来讲题。

碎碎念

第一，树结构在 OI 中经常出现，而且可以和许多东西结合，基本上没有什么特定的思路。所以基本上是杂题选讲……

第二，即便如此，我还是把题目分到了几个类别中，方便大家体会一类题目的那股劲！

第三，由于按照类别来讲，因此题目不按照难度排序，所以如果你掉线，下个题的时候记得重连一下。

第四，如果会了一个题或者有任何思路，都可以举手上来讲题。

第五人格启动！

P2515

luogu.com.cn/problem/P2515

P2515

令 D_i 为 i 的父亲，我们得到了一个树和基环树。对于基环树的环，环上的物品要么全选要么全不选，因此可以看成一个大物品，其权值为权值和，重量为重量和。

令 $dp_{u,i}$ 表示 u 子树内选了一个包含根的联通块，使得选的重量和不超过 i。

初始化为 $dp_{u,i} = V_u, (i \geq W_u)$，转移为 $dp_{u,i+j} \leftarrow dp_{u,i} + dp_{v,j}$。

时间复杂度为 $O(nm^2)$。

CF494D

luogu.com.cn/problem/CF494D

CF494D

$f(u,v)$ 可以写成 $2\sum_{x\in S(v)} d(u,x)^2 - \sum_{x\in S(1)} d(u,x)^2$。因此，我们只需要求 $\sum_{x\in S(v)} d(u,x)^2$ 即可。

考虑 $d(u,x)^2 = (d(1,u) + d(1,x) - 2d(lca(u,x))^2$，把括号拆开逐项计算。

若 $u \notin S(v)$，这种情况比较简单，因此 $lca(u,x) = lca(u,v)$，因此 $d(1,u)$ 和 $d(lca(u,x))$ 是常数。只要维护 $\sum_{x\in S(v)} d(1,x), \sum_{x\in S(v)} d(1,x)^2$

若 $u \in S(v)$，则 u 到 v 路径上的每个点都可能成为 lca，对于路径上的每条边 (w, fa_w) 都预处理出 $\sum_{x\in S(fa_w), s\notin S(w)} d(fa_w, x)$ 和 $\sum_{x\in S(fa_w), s\notin S(w)} d(fa_w, x)^2$。查询的时候相当于查询链和，因此维护一下前缀和即可。

时间复杂度 $O(n + q\log n)$，带 $\log$ 是因为要求 lca。

P4757

luogu.com.cn/problem/P4757

P4757

令 dp_u 表示只考虑 u 子树内的链，答案最大是多少。首先我们知道 $dp_u \geq \sum_{v \in son(u)} dp_v$。除此之外，我们还需要多考虑两类链：

- 形如 (u, x) 的链，这类链只会影响一个子树。
- 形如 (x, y) 的链，(x, y) 经过 u，这类链会影响两个子树。

注意到，就算一个子树被影响导致子树内答案减少了至少 1，那么也最多给答案增加 1，这肯定不优。因子要优先保证子树 v 内的方案达到 dp_v。

P4757

于是我们令 $f_{u,x}$ 表示 u 子树内选 dp_u 条链，能否做到 (u,x) 不选。这样第一类链 (u,x) 能用当且仅当 $f_{v,x}=1$。第二类链 (x,y) 能用当且仅当 $f_{v,x}=f_{w,y}=1$。为了选最多的链，第一类链肯定能选就选，第二类链相当于做了一个一般图最大匹配，由于 $\deg\le 10$，这里可以 $O(2^{\deg})$ 做一般图最大匹配。

考虑如何计算 $f_{u,x}$，若 x 所在的子树可以放第一类链肯定亏了，此时 $f_{u,x}=0$。否则，如果 x 所在的子树可以不在最大匹配里，且 $f_{v,x}=1$，则 $f_{u,x=1}$。

时间复杂度 $O(n^2+n2^{\deg})$。

ARC101E

luogu.com.cn/problem/AT_arc101_c

ARC101E

考虑容斥，我们钦定可以边的集合 S 没有被覆盖，树删掉这些边之后被分成了若干个联通块。一个大小为 $m(m \bmod 2 = 0)$ 的连通块方案数为 $m!!$。考虑用 dp 计算这个容斥式子，$dp_{u,i}$ 表示 u 子树内，从 u 所在的连通块的大小是多少（如果为 0 认为 u 和 fa_u 的边被删了）。转移是 $dp_{u,i+j} \leftarrow \sum dp_{u,i} dp_{v,j}$，$dp_{u,0} = -\sum dp_{u,2i}(2i)!!$。

根据经典的时间复杂度分析技巧可以得到，时间复杂度为 $O(n^2)$。

loj2144

loj.ac/p/2144

loj2249

loj.ac/p/2249

loj4176

loj.ac/p/4176

loj4176

loj233

loj.ac/p/2339

loj3661

loj.ac/p/3661

loj6733

loj.ac/p/6733

loj3365

loj.ac/p/3365

loj6669

loj.ac/p/6669

loj3031

loj.ac/p/2398

loj2398

loj.ac/p/2398

CF1919G

luogu.com.cn/problem/CF1919G

CF1919G

首先考虑计算，$f(rt,u)=\mathrm{OR}!f(rt,v)$。

那么如果有 $f(i,x)=0,f(j,x)=1$，则说明从 i 的视角看 x 的儿子全是必胜，从 j 的视角看至少有一个必败。

因此可以得到只有 x 到 i 放向的那个子树是必败的，其余均是必胜。

假设 x 到 i 方向的点是 y。把点集合 S 划分成 x 一侧和 y 一侧两部分 S_1,S_2。

判断划分到哪里也很简单，只用看 $f(i,x)$ 即可，0 就丢到 S_2，1 就丢到 S_1。

CF1919G

划分之后要判定是否合法也很简单，

1. S_2, S_2 内部要能递归构造出来。
2. $\forall y \in S_1, z \in S_2, f(z, y) = f(x, y)$。
3. $\forall y \in S_1, z \in S_2 \neq x, f(x, z) = f(y, z)$。

然后我们需要判断哪个点 $y \in S_2$ 与 x 有边。
这个点 y 一定满足性质：

1. $f(y, y) = 1$。
2. $\forall z \in S_2, z \neq y, f(y, z) = f(x, z)$。
3. $\forall z \in S_1, f(y, z) = f(x, z)$。

第三条前文已经满足了，并且由这三条可以唯一确定一种可能的点。如果有多个 y，这些 y 显然是等级的，任选一个即可。

CF1919G

现在要处理的问题稍微有点变化。

对于一个点集合 S 中存在一些点 x 连了一个必败的子树，且这个子树在 S 外面，但前面的分析仍然全部适用。

CF1919G

接下来要处理的问题就是 $\forall i, j, x, f(i, x) = f(j, x)$ 的树了。
将点分成三类。

1. V_1: $f(x, x) = 0$ 这类点只能连向 V_2。
2. V_2: $f(x, x) = 1$ 且 x 没有连出去必败子树，这类点至少两条边连向 V_1。
3. V_3: $f(x, x) = 1$, x 连出去必败子树，这类点不可以连向 v_1。

CF1919G

考虑以下情况。

1. $|V_1| = 0, |v_2| = 0$，$|v_3|$ 连一起就行。
2. $|V_2| = 0, |V_3| = 0$，$|V_1| = 1$ 才可以，否则不连通。
3. $|V_2| = 0, |V_1| \neq 0, |V_3| \neq 0$，不合法。
4. $|V_2| > 0$，若 $|V_1| \leq |V_2|$ 则不合法。因为边数会大于等于点数。否则以一个 V_1 为根，儿子连所有 V_2，每个 V_2 节点下面挂至少一个 V_1 即可。V_3 可以直接挂在一个 V_2 后面。

CF1919H

luogu.com.cn/problem/CF1919H

CF1919H

首先不妨假设操作 1 的边权可以为 0。

假设我知道某个边的其中一个端点也是叶子。

以这个叶子为根考虑有根树，每条边代表了下侧的点，那么可以求出每个点的深度（用操作 2）。

然后按照深度的顺序确定每个边的父亲。假设正在处理深度为 i 的边，把深度为 $i-1$ 的边设计为 id，把根设为 inf。对于每条深度为 i 的边，把这条边设为 inf 做一次查询，即可得到父亲的权值。

CF1919H

若作为根的边两端都不是叶子，思路是类似的。

考虑维护深度为 $i-1$ 的所有边，仍然领其权值为 id。

对于深度为 i 的边 a,b，把两者的权值设计为 inf，进行一次询问。如果两者的父亲相同会得到 $2inf$，否则会得到 $2inf+id(fa_a)+id(fa_b)$。

对于深度为 i 的边 a,b,c，如果 $fa_a\neq fa_b\neq fa_c\neq fa_a$，则两两进行一次询问可以得到 $id(fa_a),id(fa_b),id(fa_c)$。

CF1919H

设深度为 i 的边为 $e_1, e_2, \cdots e_k$。对于 $i > 1$ 问一遍 e_1, e_i。

把得到的值相同的 e_i 缩在一起。

若此时仍有大于等于 3 个本质不同的元素。将 e_1, e_a, e_b 三个本质不同的元素取出，问一下 e_a, e_b，即可得到所有人的父亲。

若目前只有 2 个本质不同元素，若上一层的边有大于 2 个，则说明我们找到了一个叶子，用开头的那个方法即可。同样的，只要当前层本质不同元素个数小于上一层边数，就能找到叶子。有了叶子就可以用开头的做法。因此每层边数是递增的。

反之，上面每层一定一直有恰好两个边，也就是说是一个链，由于对称性可以随便连。

谢谢大家