互测题解题报告

南京外国语学校 魏佳泽

November 6, 2022

Contents

1 题目大意

给定一棵 n 个节点的树 T 以及树上的 m 条 **不同的** 路径 $I_i = (u_i, v_i)(u_i \neq v_i)$。具体地，$I_i$ 表示树上 u_i 和 v_i 之间的简单路径上所有点形成的点集。

考虑 T 上某条路径 $I = (u, v)$，定义 $f(I) = \sum_{i=1}^{m} \sum_{j=1}^{m} [I_i \cup I = I_j \cup I]$。

对于 T 上所有不同路径 I，求 $f(I)$ 之和，并将答案对 998244353 取模。也就是说，你需要求 $(\sum_{u=1}^{n} \sum_{v=u}^{n} f(u, v)) \bmod 998244353$。

2 输入格式

第一行三个整数 S, n, m，分别表示子任务编号，树的大小与路径条数。

接下来 $n-1$ 行，每行两个整数 x_i, y_i 表示 T 上的一条边。

接下来 m 行，每行两个整数 u_i, v_i 表示第 i 条路径 I_i。

保证给定路径两两不同。

3 输出格式

输出一行一个整数表示答案。

4 数据范围

本题采用捆绑测试，共 25 个子任务。

子任务编号模 5 的余数将子任务按数据大小划分。

- 若子任务编号模 5 余 0，则 $n, m \leq 100$。
- 若测试点编号模 5 余 1，则 $n, m \leq 500$。
- 若测试点编号模 5 余 2，则 $n, m \leq 1557$。
- 若测试点编号模 5 余 3，则 $n, m \leq 85500$。
- 若测试点编号模 5 余 4，则 $n, m \leq 2 \times 10^5$。

子任务编号除以 5 的商将子任务按特殊限制划分。

- 若子任务编号除以 5 商 0，则 T 是一条链。
- 若测试点编号除以 5 商 1，则 T 是一个菊花。
- 若测试点编号除以 5 商 2，则所有路径端点互不相同。
- 若测试点编号除以 5 商 3，则所有路径以同一点为一端。

- 若测试点编号除以 5 商 4，则无特殊限制。

对于所有测试点，$2 \le n \le 2 \times 10^5$，$1 \le m \le \min(\frac{n(n-1)}{2}, 2 \times 10^5)$，$1 \le u_i, v_i, x_i, y_i \le n$，且所有 (x_i, y_i) 形成一棵树，所有 $I_i = (u_i, v_i)$ 互不相同，$u_i \neq v_i$。

模 5 余 $i(0 < i < 5)$ 的子任务依赖于模 5 余 $i-1$ 的子任务。除以 5 商 4 的子任务依赖于除以 5 商 $0, 1, 2$ 或 3 的子任务。

各子任务分值如下表所示。

$S \bmod 5 =$ / $\lfloor \frac{S}{5} \rfloor =$	0	1	2	3	4	总和
0	1	2	3	7	7	20
1	1	2	3	4	4	14
2	1	2	5	7	7	22
3	1	3	5	4	5	18
4	2	3	3	9	9	26
总和	6	12	19	31	32	100

5 解题过程

5.1 一些约定

称 (i, j) 合法当且仅当 $I_i \cup I = I_j \cup I$。

设 $size(u, v)$ 表示以 u 为根时 v 的子树大小，包括 v。

设 $chain(u, v)(u \neq v)$ 表示经过 (u, v) 的路径条数，则 $chain(u, v) = size(u, v)size(v, u)$。

设 $center_u$ 表示经过 u 的路径条数，可预处理得到 $center_u = \binom{n+1}{2} - \sum\limits_{v \in N(u)} \binom{size(u,v)+1}{2}$。

设 $suc(u, v)(u \neq v)$ 表示 u 在 v 方向上的后继。

称一条路径是“竖直”的，当且仅当其两端具有祖先后代关系。

注意，以下所有讨论 **忽略了贡献式中可能出现的常数 2，读者根据具体含义自行推断是否需要将贡献乘以 2**。

以下设 n, m 同阶。

5.2 $n, m \le 100$

改变每个 I 的贡献形式：连通块大小平方和等于求合法有序对 (i, j) 个数。等于求合法无序对 $(i, j)(i \neq j)$ 个数乘以 2 加上 n。

因此，对每个 u, v 预处理对应路径 $I_{(u,v)}$ 覆盖到的点。枚举 $I = (u, v)$，i 和 $j(i \le j)$，若 (i, j) 合法则答案加 1。显然可以用 bitset 检查，时间复杂度 $\mathcal{O}(\frac{n^5}{w})$。可以获得 6 分签到分。

5.3 $n, m \le 500$

直接枚举 I 再枚举 i, j 没有前途。考虑直接枚举 i, j 并算出合法的 I 的数量。分两种情况讨论：

- 当 I_i 和 I_j 没有公共端点时，为使得 (i,j) 合法，I 必须经过所有的 u_i, v_i, u_j, v_j 四个端点。这一点比较显然，不作证明。因此，I 必须包含 I_i 和 I_j 的并。若恰好可以用 u_i, v_i, u_j, v_j 中的两个端点 u, v 同时覆盖 I_i 和 I_j，那么合法 I 的个数为 $chain(u,v)$。否则无合法 I。称为 **情况一**。
- 当 I_i 和 I_j 有公共端点时，因 I_i 互不相同故恰有一个公共端点，设其为 u。设另两个端点分别为 v_i, v_j，则 $v_i \neq v_j$，称为 **情况二**。此时我们需要继续分类讨论：
  - 当 $v_i \notin I_j$ 且 $v_j \notin I_i$ 时，I 必须同时经过 v_i 和 v_j。类似第一种情况解决，贡献为 $chain(u_i, v_i)$。
  - 当 $v_i \in I_j$ 或 $v_j \in I_i$ 时，因两者不可能同时发生，设发生前者。考虑 $x = suc(v_i, v_j)$，则 I 必须同时经过 x 和 v_j。当 $x \neq v_j$ 时，类似解决。当 $x = v_j$ 时，贡献为 $center_x$。

以上两种情况均可通过预处理 $I_{(u,v)}$ 并模拟做到 $\mathcal{O}(n^3)$。可以获得 18 分辛苦分。

5.4 $n, m \leq 1557$

预处理 $size(u,v)$，快速判断一个点是否在路径上，快速查询 $suc(u,v)$。

$\mathcal{O}(n^2 \log n)$ 或 $\mathcal{O}(n^2)$ 均可通过。可以获得 37 分辛苦分。

5.5 T 是一条链

这是一开始的题目.jpg。发现当一个区间的右端点和另一个区间的左端点相同时，可直接当成情况一。情况一在链上是二维数点问题。情况二只需分左端点相同和右端点相同两种情况分别计算。计算方式容易推导。

时间复杂度 $\mathcal{O}(n \log n)$。可以获得 20 分二维数点辛苦分。

5.6 T 是菊花

良心送出 14 分.jpg。

5.7 所有路径端点互不相同

从这里开始比较复杂了。

树上路径问题考虑点分治。设当前分治重心为 r，分治子树为 S。设跨过 r 的路径集合为 C，没有跨过 r 的路径集合为 K，其中被 r 的儿子 u 完全包含的路径集合为 K_u。我们需要考虑这些贡献：

- C 和 C 之间的贡献。贡献有两种形式：
  - u_i 为 u_j 的祖先且 v_i 为 v_j 的祖先。此时贡献为 $size(r,u_j)size(r,v_j)$（注意不是分治子树上的 $size$，而是原树 $size$）。树上离线二维偏序，考虑在 u_i 处统计贡献，并令 v_i 为 v_j 的祖先的限制由时间戳满足。也就是说，对 S 进行 dfs 后得到每个点的分治子树子树大小 sz_i 和时间戳 dfn_i，则在 u_j 处往线段树 T_{u_j} 加入下标 $dfn(v_j)$，权值 $size(r,u_j)size(r,v_j)$。线段树合并至 u_i 时查询 T_{u_i} 下标 $[dfn(v_i)+1, dfn(v_i)+sz(v_i)-1]$ 的区间和。复杂度关于 $|C|$ 线性对数。
  - u_i 为 u_j 的祖先且 v_j 为 v_i 的祖先。如果类似上一部分用线段树合并，则需要树剖支持链查询，且查询区间共有 $\log n$ 个。考虑倒过来通过可持久化线段树对 T_i 做从根到叶子的前缀和。在 u_i 处加入下标 $dfn(v_i)$，权值 $size(r,v_i)$。在 u_j 处统计答案，此时只需要查询一次区间和，将其乘以 $size(r,u_j)$ 得到真正贡献。复杂度关于 $|C|$ 线性对数。

- K_u 和 $K_v(u \neq v)$ 之间的贡献。显然，$I_i \in K_u$ 和 $I_j \in K_v$ 存在合法 I 当且仅当 I_i 和 I_j 在 S 上均竖直。这部分比较好做。设 in_u 表示 K_u 内所有竖直链较深端点对应 $size$ 之和，则贡献为 $\sum\limits_{u \neq v} in_u in_v$，写成 in_u 之和的平方减去 in_u 的平方和可做到关于 $|K|$ 线性。

- C 和 K 之间的贡献。设 f_u 表示是否存在以 u 为较深端点的不跨过 r 的竖直路径。枚举 C 的每条路径 I_i。同样的，$I_j \in K$ 与 I_i 存在合法 I 的必要条件为 I_j 竖直。考虑一个端点 u_i 的贡献，v_i 同理。
  - 当端点 $u_i \neq r$ 时，只有和 u_i 在 r 的同一子树的 I_j 才会产生贡献。因此，枚举 u_i 在 S 上的后代 v，若 f_v 则答案加上 $size(r,v)coef(v_i)$，其中当 $v_i \neq r$ 时 $coef(v_i) = size(r, v_i)$，否则为 n 减去 $size(r, bel(u_i))$，其中 $bel(u_i)$ 即 r 包含 u_i 的儿子。通过对 $f \times size$ 做子树和做到关于 $|C|$ 线性。
  - 当端点 $u_i = r$ 时，另一个端点 $v_i \neq r$。此时 $K_{bel(v_i)}$ 的贡献已经算过了，所以贡献形如 $\sum\limits_{v \notin subtree(bel(u_i))} size(r, u_i) size(r, v) f_v$。同样的，对 $f \times size$ 做子树和做到关于 $|C|$ 线性。

一个细节：如何实时维护 $size(R, u)$。类似 "[ZJOI2015] 幻想乡战略游戏" 一题在点分树上移动时在线维护每个点的子树大小的方法，设 $delta_i$ 表示 i 子树大小的增量，当分治重心从 r 变成 r' 时，$x = suc(r, r')$ 的子树大小会增加 n 减去 $size(r, x)$。则 $size(r', x)$ 等于分治子树 S' 子树大小 sz_x 加上子树内所有点的 $delta$ 之和。

由于 $\sum |C| = \mathcal{O}(n)$，$\sum |K| = \mathcal{O}(n \log n)$，而上述做法关于 $|C|$ 线性对数，关于 $|K|$ 线性，所以总时间复杂度 $\mathcal{O}(n \log n)$。常数很大。可以获得 22 分辛苦分。

5.8 所有路径以同一点为一端

以相同端点为根 R，类似 T 是一条链解决：若两条路径 I_i 和 I_j 其中 $u_i = u_j = R$ 仅在 R 处相交，则合法 I 的数量为 $size(R, v_i) size(R, v_j)$，即 $sz_{v_i} sz_{v_j}$。否则类似 $n, m \leq 500$ 时的情况二：

- 若 $v_i \notin I_j$ 且 $v_j \notin I_i$，则合法 I 的数量为 $chain(v_i, v_j)$。
- 否则不妨设 $v_i \in I_j$，设 $x = suc(v_i, v_j)$，若 $x = v_j$ 则贡献为 $center_x$，否则贡献为 $chain(x, v_j)$。

据此，考虑先将 I_i 与 I_j 之间的贡献视为 $chain(v_i, v_j)$ 即 $sz_{v_i} sz_{v_j}$，再修正错误的贡献。容易发现当 v_i 是 v_j 祖先时贡献错误。考虑 $x = suc(v_i, v_j)$，若 $x = v_j$，则贡献 $center_x$，否则贡献 $chain(x, v_j)$。

为了快速计算贡献，鉴于 x 为 v_j 祖先且 $x \neq v_j$，所以 $size(v_j, x) = n - sz_{suc(x, v_j)}$。因此，设 sum_i 表示 i 子树内所有作为端点的节点的 sz 之和，$sucsum_i$ 表示 i 的所有儿子 x 的 $sum_x(n - sz_x)$。对于树上每个点 i 及其儿子 x，产生贡献为 $sucsum_x$，且若 x 为一端，则再加上 $center_x$ 的贡献。

时间复杂度 $\mathcal{O}(n)$。可以获得 18 分辛苦分。

5.9 无特殊限制

将所有路径端点互不相同和所有路径以同一点为一端的做法有机结合即可。做法非常复杂，但只要细心分类就很有条理。

- 计算 C 和 C 之间的端点不重合的贡献。注意端点互不相同的做法会统计到形如 (u_i, r) 和 (r, v_j) 的 (i, j) 对贡献，需要减去。注意合并和更新顺序，不要算到端点重合的贡献了。

- 计算 C 和 C 之间端点重合的贡献，即枚举 S 内每个点 u，对以 u 为一端的所有另一端 v_i 建出虚树，在虚树上跑所有路径以同一点为一端的 DP。注意这样是正确的因为路径跨过 r。所以以 r 为根和以 u 为根效果相同。这也是为什么对于不跨过 r_i 的以 u 为一端的路径，我们不能在此时考虑它。
- 计算 K_u 和 $K_v(u \neq v)$ 之间的贡献，这是容易的。
- 计算 C 和 K 之间的贡献。端点不重合已经讨论过。对于端点重合，需要对每个点 u 记录 h_u 表示是否存在以 u 为一端的 $v = suc(r, u)$ 的路径，再记录 g_u 表示所有以 u 为一端的 K 内的路径的贡献系数。若 v_i 为 u 的祖先，则当 $v_i \neq suc(r, u)$ 时产生贡献系数 $size(r, suc(v_i, r))$，其中 $suc(v_i, r)$ 也就是 v_i 在 S 上的父亲，当 $v_i = suc(r, u)$ 时在 h 中计算。否则 v_i 不为 u 的祖先，根据之前的讨论，贡献系数为 $size(r, v_i)$。枚举 C 的路径 (u_i, v_i) 并计算与 u_i 重合或 v_i 重合的 K 的路径的贡献。一个重要注意点是 u_i 或 v_i 为分治重心 r 时，无论是重合端点的系数，还是不重合端点的系数都要特殊考虑，也正是为此我们才需要记录 h：例如，当 u_i 重合，$v_j = suc(r, u_i)$ 且 $v_i = r$ 时，不重合部分退化为单点，也就是要用 $center_r$ 贡献的情况。

对 S 树剖，由于虚树大小之和为 $\mathcal{O}(|C|)$，所以我们只需要求 $\mathcal{O}(|C|)$ 次 LCA 和后继 suc。因此使用 $\mathcal{O}(n)$ 预处理 $\mathcal{O}(\log n)$ 查询的树剖，防止预处理使复杂度退化。总时间复杂度 $\mathcal{O}(n \log n)$，达到了复杂度下界。可以获得 100 分满分。代码详见：`https://www.luogu.com.cn/paste/jr3svh48`。

6 Alternative Solution

分端点相同和不同讨论。

- 若端点不同，则 I_i 和 I_j 必须形成一条路径 (u, v)。枚举每条路径 I_i，枚举交叉或包含。对于包含，I_j 的贡献形如二维数点。对于交叉，I_j 一个端点的限制是一条路径，可以树上可持久化线段树，也可以通过一些技巧做到空间线性，但不必要。
- 若端点相同，类似“所有路径以同一点为一端”子任务的做法，以该点为根建虚树，然后树形 DP 即可。

如何以某个点为根建虚树呢？看似我们需要比较两点在以某个点为根时的时间戳，但这个问题是很搞笑的，因为广义上的虚树是没有根的，我们只需要按以 1 为根建出虚树，然后以对应点为根深搜即可。时间复杂度 $\mathcal{O}(n \log n)$，空间复杂度 $\mathcal{O}(n)$ 或 $\mathcal{O}(n \log n)$，代码比点分治好写很多，但依然有大量细节和分类讨论。

7 参考资料

1. ZJOI2015 幻想乡战略游戏：`https://www.luogu.com.cn/problem/P3345`。