《基础图论练习题》解题报告

戴江齐

2021.9

1 题目大意

有一张 n 个点的有边权无向图，结点按 $\{0,1,\cdots,n-1\}$ 编号。初始时有 b 条边，第 i 条连接 u_i 和 v_i，边权为 w_i。

接下来依次对它进行 a 次操作。第 i 次操作中，每对编号差为 d_i 的结点之间被连了一条权为 x_i 的边。

记最终得到的图为 G，其连通块依次为 $G_0,G_1,\cdots,G_{k-1}$。记 $f(G_i)$ 为 G_i 的最小生成树大小（边权和），求 $\sum_{i=0}^{k-1} f(G_i)$。

答案对 998244353 取模。

2 数据范围

对于所有测试点：$1\le n\le 10^{18}$，$0\le a,b\le 5\times 10^4$，$1\le d_i<n(1\le i\le a)$，$0\le x_i<998244353(1\le i\le a)$，$0\le u_i,v_i<n,u_i\ne v_i(1\le i\le b)$，$0\le w_i<998244353(1\le i\le b)$。

特殊限制 A：所有 x_i 和 w_i 均为 1。

subtask 1(8 pts)：$n\le 2\times 10^5,a\le 10$；

subtask 2(6 pts)：$n\le 2\times 10^5$；

subtask 3(6 pts)：$a=2,b=0$；

subtask 4(16 pts)：$a=2,b\le 5\times 10^4$；

subtask 5(12 pts)：$a\le 1000,b=0$，满足特殊限制 A；

subtask 6(12 pts)：$a\le 1000,b\le 200$;

subtask 7(12 pts)：$b=0$；

subtask 8(10 pts)：满足特殊限制 A；

subtask 9(18 pts)：无特殊限制。

时间限制 3s，空间限制 512MB。

3 解题过程

思路 1

考虑 $n \leq 2 \times 10^5$ 的情况。

由于问题是求最小生成树，很容易想到 Kruskal 算法。

算法 1.1

直接对所有边使用 Kruskal 算法，时间复杂度 $O((na+b)\log(na+b))$，可以通过 subtask 1。

算法 1.2

将边排序后，问题被转化成维护一张图，需要支持连接每对差为 d_i 的点、连接两个点和查询连通块数三种操作。

考虑建立一个有 $\log(n)$ 层、类似 ST 表的结构，其中在第 $i(i \geq 0)$ 层里连接 a 和 b 表示对所有 $j \in [0, 2^i)$ 在原图中连接 $a+j$ 和 $b+j$。插入一组差为 d_i 的边时，只需在第 $\lfloor \log(n-d_i) \rfloor$ 层插入两条边；当第 i 层的两个连通块合并时，需要将第 $i-1$ 层对应的两组点对合并。该算法实现时有一些细节，由于与最终解法基本无关，此处不再赘述。

时间复杂度 $O((n+a+b)\log(n)\alpha(n))$，可以通过 subtask $1, 2$。

思路 2

考虑 $a = 2$ 的情况。

瓶颈仍然是每插入一组边（或一条散边）求当前连通块数。

算法 2.1

当 $b = 0$ 时，只需求出所有边都被插入后的连通块数，容易发现这是 $\max(d_1 + d_2 - n, \gcd(d_1, d_2))$。证明可以使用裴蜀定理或直接套用（字符串中的）periodicity lemma；由于算法 2.2 中的性质可以直接推出该命题的正确性，此处不再赘述。

时间复杂度 $O(\log(n))$，可以通过 subtask 3。

算法 2.2

不失一般性地，考虑整组（差为定值的）边都被插入后图的连通性的变化。建一张 n 个点的图 H 使得 H 中有且仅有差为 d_1 或 d_2 的点对之间连边，则只需对 u_i, v_i 中的每个点求出它在 H 中所在连通块的一个代表元（不妨令其为连通块中最小的元素），再对这些代表元跑朴素 Kruskal 算法即可。

考察 H 的性质。不妨设 $d_1 < d_2 < n$，则 H 中的点 $v = n - d_1, n - d_1 + 1, \cdots, n - 1$ 内部没有边，且 v 只要不是孤立点就一定和 $v - d_1$ 相连；另一方面，对任何两个差为 d_2 的点 $(x - d_2, x)$，$(x - d_2, x - d_1)$ 必然在一个连通块里。因此，记 H' 为一张 $n - d_1$ 个点的图使得其中只有差为 d_1 或 $d_2 - d_1$ 的点对相连，则 H 中 v 的代表元为：

$$R(H, v) = \begin{cases} R(H', v) & v < n - d_1 \\ v & n - d_1 \le v < d_1 \\ R(H', v - d_1) & v \ge \max(d_1, n - d_1) \end{cases} \tag{1}$$

其中 $R(H, v)$ 表示图 H 中 v 所在连通块的代表元。

注意到这是辗转相减的过程。将辗转相减改写为辗转相除，即可在 $O(b \log(n))$ 的时间内求出所有涉及到的连通块的代表元。

时间复杂度 $O(b \log(n))$ [1]，可以通过 subtask 3, 4。

思路 3

考察一般情况。

算法 3.1

考虑 $b = 0$ 且 x_i 全为 1 的情形。仿照算法 2.2，建一张 n 个点的图 H 使得 H 中有且仅有差为 $d_i (1 \le i \le a)$ 的点对之间连边，则此时之需求 H 的连通块数。

[1] 从此处起，无特殊说明时，复杂度分析中统一认为 $O(\log(a+b)) < O(\log(n)) < O(a+b) < O(n)$

不妨设 d_i 单调增。记 H' 为一张 $n-d_1$ 个点的图使得其中只有差为 d_1 或 $d_i-d_1(2\le i\le a)$ 的点对相连，则

$$R(H,v)=\begin{cases}R(H',v) & v<n-d_1\\ v & n-d_1\le v<d_1\\ R(H',v-d_1) & v\ge\max(d_1,n-d_1)\end{cases}\tag{2}$$

仍然成立。

记一轮操作为不断进行上述操作直至 $n<d_1$ 或 d_1 不再是 d_i 中的最小值（类似辗转相除中的取模），则每过两轮操作 d_i 的最小值至少折半（这是因为 $d_1 \bmod (d_i \bmod d_1)\le\frac{d_1}{2}$），也就是说总共只需 $O(\log(n))$ 轮操作。过程中可能出现值为 0 的 d_i，需删去。

时间复杂度 $O(a\log(n))$，可以通过子任务 5。

算法 3.2

在算法 3.1 的基础上，维护所有 u_i,v_i 所在连通块的代表元，再跑 Kruskal 算法即可。

事实上，不需要在每插入一组边时都显式地跑一遍 Kruskal；事实上，令 G' 为 G 中只保留整组边（即删去 b 条散边后）得到的图，则只需求出所有 u_i,v_i 在 G' 的 Kruskal 重构树上的虚树（相当于这些点在对 G' 进行 Kruskal 过程时两两被合并的时间），将虚树的权值从答案里减去，再将虚树里的边和 b 条散边一起跑 $Kruskal$ 算法。

时间复杂度约为 $(a([x_i\text{ 中不同的值个数}]+b)\log(n))$（取决于具体实现），可能通过子任务 $6,8$。

算法 3.3

考虑 $b=0$ 的情形，此时只需要每插入一组边求出当前的连通块数。

注意到对固定的序列 d_i，算法 3,1 中对 H 进行的操作只有 $O(\log(n))$ 轮，也就是说只有 $O(\log(n))$ 个 d_i 真正对图的连通性起了作用。进一步地，只需在每插入一组边后都只保留这 $O(\log(n))$ 个有效的 d_i，即可在 $O(a\log^2(n))$ 的时

间内解决问题。

用堆优化取最小 d_i 的过程还可以达到 $O(a\log(n)\log(\log(n)))$ 的时间复杂度，可通过子任务 7。

算法 3.4

当 $b \neq 0$ 时，还需对求所有关键点（即所有 u_i, v_i）在 G' 的 Kruskal 重构树上的虚树（即 Kruskal 过程中被不断两两合并形成的树形结构）这一部分进行优化。

使用整体二分，记过程 $solve(L, R, S)$ 表示：已知插入第 L 组边前 S 中元素任意两个均未被合并，插入完第 R 组边后 S 中元素被合并到同一个连通块里，求 S 中所有元素在这段时间的具体合并方式。具体地，令 $M = \lfloor L + R2 \rfloor$。用 $O(|S|\log(n))$ 的时间求出插入第 M 组边后 S 中元素形成的连通块 $S_1, S_2, \cdots, S_k$，并向左递归 $solve(L, M, S_i)(1 \leq i \leq k)$，向右递归 $solve(M+1, R, \{v_1, v_2, \cdots, v_k\})$（$v_i$ 为 S_i 中任一元素）。注意到分治树中每一层 $|S| - 1$ 的总和不变，故该部分复杂度为 $O(b\log(n)\log(a))$。

总时间复杂度 $O(a\log(n)\log(\log(n)) + b\log(n)\log(a))$，可以通过全部测试数据。

4 命题思路

本题的灵感来源于 Grand Prix of Moscow K: Bipartite Graph [2] 和 TopCoder SRM 600.5 Division One Level Two: Huge Graph，分别对应子任务 4 和 5。事实上，本题是这两题的加强和推广。

本题的子任务层层递进，有助于产生较好的区分度，在思维上也起到一定提示作用。前两个子任务不需要观察性质，分别使用朴素和加优化的 Kruskal 算法即可通过；子任务 3 和 4 则需要借助辗转相除（减）的过程分析 $a = 2$ 时图的性质；子任务 $5, 6, 8$ 需要将 $a = 2$ 时的性质推广到一般情况；子任务 $7, 9$ 则需要在发现性质的基础上对算法进行优化。

综上，本题思维难度较高，代码难度适中，涉及了对最小生成树、辗转相除

[2]需要 yandex 账号才能查看

和整体二分等经典算法的深入挖掘，全面、有深度地考查了选手分析性质和实现算法的能力，是一道不可多得的好题。

5 备注

值得一提的是，本题中图的结构与字符串 border 理论中 border group 的结构极为相似，有兴趣的读者可以自行了解。事实上，笔者在构造极限数据时充分利用了这一相似性。

另外，最终做法的效率瓶颈在于堆优化和整体二分，与图本身的结构基本无关，笔者认为这或许是因为图的特殊性质仍未被充分利用。因此，若有读者发现了复杂度更优或复杂度相同但省去了堆或整体二分的做法，欢迎与笔者交流探讨。

6 参考资料

[1] XVI Open Cup, Grand Prix of Moscow K: Bipartite Graph, https://official.contest.yandex.ru/opencupXVI/contest/2366/problems/K

[2] TopCoder SRM 600.5 Division One Level Two: HugeGraph, https://community.topcoder.com/stat?c=problem_statement&pm=12832

[3] 王鉴浩，浅谈字符串匹配的几种方法，国家集训队 2015 论文集