2025 广西邀请赛部分题目讲评

isaunoya

Hangzhou Dianzi University

目录

E. Elimination

F. Faster Elimination

L. Lost in MST

附录

目录

E. Elimination

F. Faster Elimination

L. Lost in MST

附录

E. Elimination：题意

2025 广西邀请赛 E。给定小写字母串 S，选择非空串 T。每次可以：

1. 删除 T 的末字符；
2. 找到 T 在当前 S 中的一次出现，删除 S 从开头到这次出现的**起点**，再删除 T 的末字符。

最大化第二种操作的次数。多组数据，总长度不超过 6000。

E. Elimination：题意

2025 广西邀请赛 E。给定小写字母串 S，选择非空串 T。每次可以：

1. 删除 T 的末字符；
2. 找到 T 在当前 S 中的一次出现，删除 S 从开头到这次出现的**起点**，再删除 T 的末字符。

最大化第二种操作的次数。多组数据，总长度不超过 6000。

一次匹配后只越过它的起点，**没有删掉整次出现**。后续匹配允许重叠，起点必须严格递增。

E：只保留成功的匹配

若某个方案成功匹配 k 次，第 t 次匹配时，模式长度至少为 $k-t+1$。

将初始模式截成前 k 个字符，删掉所有不产生匹配的操作；在原来的起点依次匹配长度 $k, k-1, \ldots, 1$，每次仍然合法。

因此存在最优方案：初始模式长度等于答案，每次都成功匹配，然后删去末字符。

下面只需考虑这样的方案。数组下标从零开始，子串用半开区间表示。

E：从首次匹配的起点转移

令 f_i 为首次匹配起点是 i 时，最多成功匹配多少次。若下一次从 $j > i$ 开始，剩余模式长度 h 满足

$$h \le f_j, \quad h \le \mathrm{lcp}(i, j).$$

于是

$$f_i = 1 + \max_{j>i} \min\bigl(f_j, \mathrm{lcp}(i, j)\bigr).$$

没有后继时最大值取零。倒序计算 f_i，答案为 $\max_i f_i$。

E：转移为何充分

长度 f_j 的可行方案可以截短为任意 $h \le f_j$ 的方案：保留前 h 次匹配，并将每次模式都缩成相应前缀。

若另外有 $h \le \mathrm{lcp}(i, j)$，先在 i 匹配 $S[i, i + h + 1)$，再从 j 继续这个长度 h 的方案即可。

长度不会越界：归纳有 $f_j \le n - j$；由于 $j > i$，总有 $h + 1 \le n - i$。

反过来，任意非单次方案都有这样的第二个起点 j，故转移也不会漏掉最优方案。

E：滚动维护 LCP

$$\operatorname{lcp}(i,j)=\begin{cases}1+\operatorname{lcp}(i+1,j+1) & S_i=S_j\\ 0 & S_i\neq S_j\end{cases}.$$

计算第 i 行只依赖第 $i+1$ 行，可用一个数组 L。外层 i 倒序，内层 $j=i+1,\ldots,n-1$ 正序，保证 $L[j+1]$ 仍是旧行的值。

更新 $L[j]$ 后，立即用 $1+\min\bigl(f_j,L[j]\bigr)$ 更新 f_i；每行从 $f_i=1$ 开始。$L[n]=0$ 作为边界。

两层循环各处理一对起点，时间 $O(n^2)$、空间 $O(n)$。状态 f_j 不被覆盖，$n=1$ 时内层为空，答案自然为一。

目录

E. Elimination

F. Faster Elimination

L. Lost in MST

附录

F. Faster Elimination：题意

2025 广西邀请赛 F。给定小写字母串 S，选择模式串 T；每次可以单独删去它的末字符，或匹配一次 T，越过这次出现的起点，再删去末字符。

最大化成功匹配次数。多组数据，总长度不超过 5×10^5；匹配可以重叠。

F. Faster Elimination：题意

2025 广西邀请赛 F。给定小写字母串 S，选择模式串 T；每次可以单独删去它的末字符，或匹配一次 T，越过这次出现的起点，再删去末字符。

最大化成功匹配次数。多组数据，总长度不超过 5×10^5；匹配可以重叠。

仍令 f_i 为首次匹配从 i 开始时的答案。二次转移已经成立；现在要快速求出所有已处理起点能提供的最长前缀。

F：把转移变成字符串集合查询

处理 i 前，对每个 $j > i$，将 $S\left[j, j + f_j\right)$ 及其全部前缀加入集合 $\mathcal{P}$。

令 L 为 $S[i, n)$ 属于 $\mathcal{P}$ 的最长前缀长度，则

$$L = \max_{j>i} \min\left(f_j, \mathrm{lcp}(i, j)\right), \quad f_i = L + 1.$$

计算完 f_i 后，将它对应的串及全部前缀插入。空串始终存在，因此第一次查询也有定义。

集合对前缀封闭；需要维护的不是任意一组字符串长度。

F：不必显式插入全部前缀

令集合 $\mathcal{P}$ 初始只含空串。每次查询最长已存在前缀 L，然后只插入长度 $L+1$ 的串。

若插入前集合对前缀封闭，则这个新串的长度 L 前缀已经在集合中，它的更短前缀也已经在。因此插入后集合仍然对前缀封闭。

由归纳，每个位置只新增一个串，就等价于把它及全部前缀加入集合。整个过程中至多保存 n 个非空串。

这一点使得普通哈希表就足够，不需要维护一棵字符串树。

F：二分最长已存在前缀

对当前起点 i，判断 $S[i, i+k)$ 是否在集合中。由前缀封闭性，判断结果随 k 增大必然先真后假。

长度零始终为真；长度 $n-i$ 为假，因为所有已处理起点 $j > i$ 插入的长度至多 $n-j < n-i$。

在这两个边界之间二分，得到最后一个真值 L。令 $f_i = L+1$，把 $S[i, i+f_i)$ 插入集合，再处理 $i-1$。

必须先查询再插入，否则当前串会被当成已经存在的后继方案。

F：子串指纹与集合键

将 a 至 z 编码为 $1, \ldots, 26$。取素数 p 和随机底数 b，预处理幂与前缀哈希，以下运算均对 p 取模：

$$H_{k+1} = H_k b + \mathrm{code}(S_k), \quad H_0 = 0.$$

于是

$$\mathrm{hash}(l, r) = H_r - H_l b^{r-l}.$$

使用两份独立底数得到两个指纹。哈希表的键为 $(r - l, h_1, h_2)$，长度也参与判等；查询时无需创建或复制子串。

模减法规范化到 $[0, p)$。哈希表自身的桶碰撞由键相等比较处理，与字符串指纹碰撞是不同的问题。

F：从右向左维护前缀集合

先预处理两套前缀指纹。从右到左处理每个起点：在当前集合中二分最长可行前缀 L，令 $f_i = L + 1$，再插入长度 f_i 的一个串。答案为 $\max_i f_i$。

每个位置新增一个键、进行 $O(\log n)$ 次查询。哈希表操作均摊期望为常数时，总时间 $O(n \log n)$、空间 $O(n)$。

两份底数独立均匀取自 $\{2, \ldots, p-2\}$，可取 $p = 2^{61} - 1$。对固定输入，首次错误之前沿精确执行路径作并集界；若查询数为 Q，全程误判至多

$$nQ\left(\frac{n-1}{p-3}\right)^2, \quad Q \le n\lceil \log_2(n+1) \rceil.$$

自适应查询的错误率证明与参数约定

目录

E. Elimination

F. Faster Elimination

L. Lost in MST

附录

L. Lost in MST：题意

2025 广西邀请赛 L。完全图有 n 个点，第 i 个点有正整数点权 c_i，边权为

$$w(x,y)=\gcd(x,y)+\operatorname{lcm}(x,y).$$

$n\le 10^5$，$1\le c_i\le 10^6$，求最小生成树权值。

L. Lost in MST：题意

2025 广西邀请赛 L。完全图有 n 个点，第 i 个点有正整数点权 c_i，边权为

$$w(x,y)=\gcd(x,y)+\operatorname{lcm}(x,y).$$

$n\le 10^5$，$1\le c_i\le 10^6$，求最小生成树权值。

完全图无法显式建出。先合并相同点权，再为每个因子选择一个最小倍数作为代表；最后处理被合并的副本。

L：因子代表与候选边

暂时假设所有点权互不相同。对每个因子 d，令 z_d 为现有点权中最小的 d 的倍数。

对每个满足 $d \mid x$ 且 $x \neq z_d$ 的点加入候选边 (x, z_d)，暂定边权为

$$W_d(x, z_d) = d + \frac{x z_d}{d}.$$

同一对端点可以产生多条候选边，直接保留即可。候选权值不一定等于真实 gcd/lcm 边权，正确性需要单独证明。

L：候选权值不会低估

取任意候选边 (x, y)，真实 gcd 为 g。因为 $d \mid g$，有

$$0 < d \le g \le \sqrt{xy}.$$

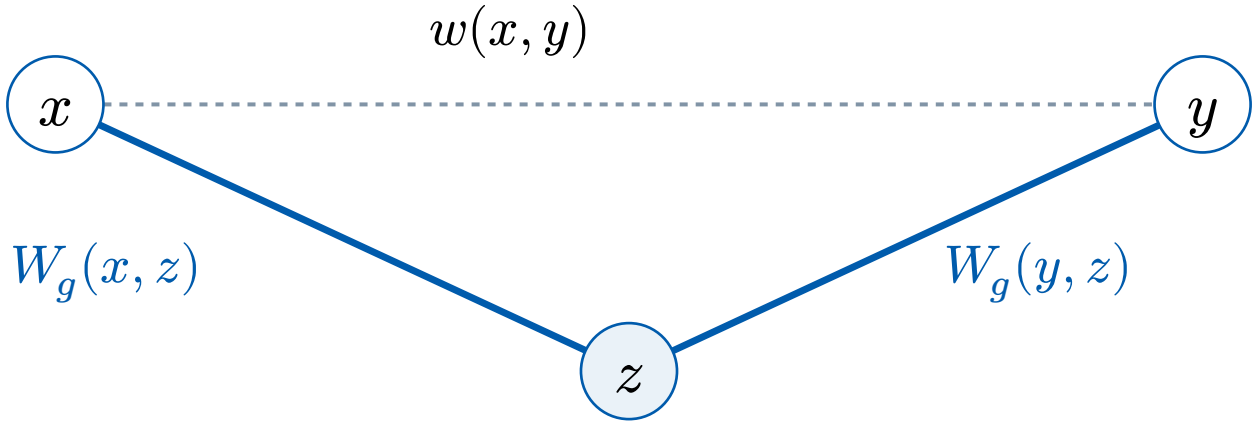
函数 $t + x\frac{y}{t}$ 在此区间单调不增，因此

$$w(x, y) = g + \frac{xy}{g} \le d + \frac{xy}{d} = W_d(x, y).$$

候选图中阈值不超过 t 的边，在原图中也不会超过 t。接下来证明，候选图也没有漏掉必须的连通关系。

L：原图的一条边可由两条候选边替代

取原图边 (x, y)，令 $g = \gcd(x, y)$，$z = z_g \le \min(x, y)$。



$$\max\big(W_g(x, z), W_g(y, z)\big) \le g + \frac{xy}{g} = w(x, y).$$

两条候选边都不比原边重，因此原图阈值 t 下的一条边，可以换成候选图同一阈值下的路径。若 z 恰好是端点，路径退化为一条边。

L：阈值连通性决定 MST 权值

综合两个方向，两图在每个权值阈值下的连通块划分完全相同。

Kruskal 按边权递增合并。若权值 t 处连通块数量减少 k，任何最小生成树都恰需在这一层付出 kt 的代价。

所以两图最小生成树的总权值一致，可以直接用 W_d 跑 Kruskal，无需为每条候选边再计算 gcd。

因子 1 已把所有不同点权连接到最小值，候选图一定连通。

L：候选边生成与复杂度

将不同点权从小到大扫描，枚举 x 的所有约数 d。维护 $\mathrm{first}[d]$：若已存在，就连边；否则记录 x。

设不同值集合为 A，$V = \max c_i$，实际候选边数为 E，则

$$E \le \sum_{x \in A} \mathrm{tau}(x) \le \sum_{x=1}^{V} \mathrm{tau}(x) = O(V \log V).$$

可用倍数筛预处理约数，或用最小质因子分解后生成约数。Kruskal 排序时间 $O(E \log E)$，空间取决于实际保存的约数和边数。

不能把单个数的约数个数统一写成 $O(\log V)$。

L：重复点权的接入代价

值 x 出现 b_x 次。副本之间边权均为 $2x$，它们到任意外部顶点的边权也完全相同。

令 m_x 为 x 到其他不同值的最小边权；只有一种值时令 $m_x = \infty$。额外副本每个增加的最优代价为

$$c_x = \min(2x, m_x).$$

阈值小于 c_x 时，每个副本都孤立；达到 c_x 后，它们或者内部连通，或者通过共同外部邻点连通。

因此额外的 $b_x - 1$ 个连通块恰好在阈值 c_x 消失。

L：从已求出的树取得最小邻边

任意一棵 MST 中，顶点 x 的最小树边权等于图中它的最小邻边权。

否则加入图中更轻的邻边，形成一个环；删去环上另一条与 x 相接的树边，会严格减小总权值，矛盾。

候选图和原图具有相同的阈值连通性，顶点第一次不再孤立的阈值相同。

因此扫描 Kruskal **选中的树边**，即可取得所有 m_x。

不需要为重复值另做一次完全图最短邻边搜索。

L：合并答案与边界

$$\text{answer} = \text{MST}(A) + \sum_{x \in A} (b_x - 1) \min(2x, m_x).$$

先按值压缩并记录出现次数；生成候选边；运行 Kruskal；统计每个代表的最小树边；最后加回副本代价。

只有一种值时答案为 $(n-1) \times 2x$；$n = 1$ 时答案为零。计算 xy、候选权值和总答案都使用 64 位整数。

若复用预处理处理多组数据，只清空本次访问过的因子代表；不要让上一组的 first 留在下一组。

目录

E. Elimination

F. Faster Elimination

L. Lost in MST

附录

F：指纹碰撞如何影响答案

在所有查询都正确的条件下，集合与二分严格实现前面的 DP。不同串指纹相同可能使集合产生假阳性，继而影响后续插入。

不能把这些受此前结果影响的查询直接当成独立试验。固定输入后，考虑**完全使用字符串判等的理想执行**；第一次发生错误时，程序此前仍与理想执行相同。

因此只需对理想执行中的查询与当时已存字符串对做并集界。若查询总数为 Q，需考虑的比较对至多 nQ。

F：选取参数与实现边界

同长度不同串的差多项式次数小于 n。两份底数独立均匀取自 $\{2,\dots,p-2\}$，则全程误判概率至多

$$nQ\left(\frac{n-1}{p-3}\right)^2,\quad Q\le n\lceil\log_2(n+1)\rceil.$$

可取 $p=2^{61}-1$，用两份独立随机底数；乘法用无符号 128 位中间值。两个固定底数没有这条随机试验保证。

哈希表预留容量、控制负载，并在测试间清空；整体正确性与哈希表的期望耗时分别分析。答案仍按原题输出整数。

返回 F 的算法与复杂度