YDC的字符串 解题报告

雅礼中学 刘研绎

1 试题大意

给出n个字符串，要求进行q次操作，每次操作形如一下四种：

1. 在第x个字符串后面加上一个字符y。
2. 询问在第k个操作过后的第x个字符串在当前的第y个字符串中的出现次数。
3. 将第x个字符串改成第y个字符串。
4. 读入一个字符串，询问它在n个串中每一个串中的出现次数。

这里的出现次数是特殊定义的，比如说询问串s_1中s_2的出现次数，那么询问时会给出参数l, r，需要回答s_1有多少子串形如$a + s_2$（'+' 号代表字符串连接），其中a是一个字符集中在第$[l, r]$中的字符。不同的询问所给出的l, r可能不同。

要求强制在线。字符集大小为m。

$2 \le n \le 5, 1 \le m \le 10^5, 0 \le q \le 2 \cdot 10^5$

初始n个串的总长度不超过$2 \cdot 10^5 + 20$

保证任意时刻出现的n个串均是一个长度不超过$4 * 10^5$的字符串的子串。

操作4中读入的串总长度不超过10^6

2 试题分析

本题限制条件多，并且要求强制在线，不妨考虑使用后缀自动机(SAM)来处理字符串问题。

后缀自动机即一个可以识别一个字符串s的后缀的有限状态自动机。我们称一个字符串s对应一个SAM的状态x，当且仅当SAM读入s后，会达到状态x。

令$x \to len$为所有能够对应到状态x的串的最长公共后缀长度，令x的$Right$集合包含所有能够对应到x的前缀的结束位置（如果是一棵树，那么前缀为根到当前节点形成的字符串）。

2.1 为Trie建立SAM

题目的修改操作只会在某一个串的末尾加入一个字符，如果我们类似常规字符串题简单的将n个串拼接起来建立SAM就会遇到在串中间插入一个字符的问题，并不是很好处理。如果我们将n个串建立成一颗Trie树，那么每次的加入操作相当于在Trie上新增一个叶节点，这个操作将会具有更多的特殊性。

考虑SAM的建立过程其实是一个不断在原有状态上拓展的过程，即假设一个字符串s已经被建立出SAM，那么我们可以在当前SAM的基础上建立出$s + ch$的SAM，这个过程是可以放在Trie树上的。对应到我们的操作，每次新增一个Trie的叶节点，其实是符合SAM的建立过程，因此我们很轻松地就能建立出一颗Trie树的SAM。

在具体实现上和给一个串建立SAM略微有些不同，我们需要注意一些可能已经存在的转移。例如我们需要给某个s的前缀所对应的状态x新增一个ch的转移，就需要考虑x是否已经存在转移ch，而在之前的算法（给一个串建立SAM）中状态x是不可能存在转移ch 的。

在这一步我们与给一个串建立SAM的过程相类比，成功的给一颗Trie建立了SAM。

2.2 动态维护SAM

我们需要动态地询问串a在串b中的出现次数，那么在SAM中，我们只需要维护在串a所对应的状态x在Parent tree的子树中b的所有前缀所对应的状态的出现次数。在整理SAM的建立操作后不难发现，问题可以转化成：

给出一个有根树，我们需要支持以下几个操作：

- 加入一个叶节点。
- 删除一条边，再加入一条边。
- 询问某个节点的子树信息。

我们可以用动态树来解决这个问题，仔细观察后发现，删边加边实际上只是修改一个节点的父亲，而询问只需要询问子树信息，我们可以用平衡树维护DFS序来解决，代码复杂度会比动态树低。只需要在建立后缀自动机的过程中涉及parent tree修改的操作的同时修改其DFS序即可。

在这一步我们成功的动态维护了SAM。

2.3 平衡树的标记

考虑操作三，将一个串改成另一个串，我们不可能每次暴力遍历所有节点进行修改，我们需要给平衡树打上标记。而考虑$n \le 5$，因为存在标记的节点的信息在标记清空之前都不会改变，即不妨令f_i表示当前第i个串在若干赋值操作后赋值成了原来的第f_i个串。很显然f只是一个$\{1,\ldots,n\} \to \{1,\ldots,n\}$的映射，而映射的复合是满足结合律的。而打标记这个操作一定满足各个标记的先后顺序不会改变，再加上映射本身具有结合律，因此上述f可以被用作平衡树的标记。

2.4 特殊的出现次数

考虑普通的出现次数，我们直接对Parent tree的整棵子树进行询问。而加上前一个字母必须是字符集的一个连续区间的这个限制之后，询问扩展到了一棵或者多棵子树。

由于对固定字符有询问限制，因此我们需要对某一个SAM中的状态x询问出能对应到状态x的串s从后往前第$i(i \le x \to len)$个字符具体是什么，不妨令这个字符为$x \to ch(i)$。为了维护这个询问，SAM的每一个状态都至少要维护一个$Right$集合中的元素。并且要将n个串建立成一棵Trie树，并在Trie树倍增进行查询。

如果仅有一棵子树，那么我们可以直接进行特判。此时一定有$x \to len > |a|$，我们只需要判断$x \to ch(|a|)$是否在给定的$[l,r]$中即可。

如果是多棵子树，那么他们有一个共同的特点：所有子树的父亲是同一个点。考虑在这些子树共同的父亲处处理整个问题，不妨令这些节点的父亲为x，那么肯定有$x \to len = |a| - 1$。而Parent tree中并没有对一个节点的子节点之间的顺序有要求，那么我们可以给定这个顺序，不妨对于每个节点x，让其子节点son_x按照$son_x \to ch(x \to len + 1)$来排序。这样我们需要询问的子树会在DFS序中连续的一段区间，仍然可以利用平衡树的区间查询来进行询问。

而整棵树是动态的，因此维护子节点的顺序也需要动态维护，这里我们仍然需要使用到平衡树（c++选手可以使用stl中的set）。

这样我们就解决了特殊的出现次数的询问。

2.5 总结

我们利用给Trie建立的SAM并动态维护来解决这个问题。

对于操作一，类比给串建立SAM，同时动态维护Parent tree的DFS序。

对于操作二，考虑某个串的历史版本对应的状态x不会消失，我们记录下这个状态x，在Parent tree中x的子树进行询问即可。

对于操作三，对整棵平衡树打上一个标记。

对于操作四，将询问串s读入SAM，按照询问特殊的出现次数的方法询问即可。

本题的字符集较大，运用SAM的时候可能需要借助hash表来完成。

时间复杂度为$O(L \log L)$，L即所有插入（包括初始的串）的总长度。预计有1至3位集训队选手能拿到满分，3至6位能够拿到60至80分，所有人都能拿到至少30分。