字符串游戏

【问题描述】

给定N个仅有a~z组成的字符串a_i，每个字符串都有一个权值v_i，有M次操作，操作分三种：

$Cv\ x\ v'$：把第x个字符串的权值修改为v'

$Cs\ x\ a'$：把第x个字符串修改成a'

Q：求出当前的最大权字符串集合，使得这个集合中的字符串经过重新排列后满足除最后一个字符串外，前一个字符串是后一个的前缀（两个字符串相同也是前缀关系，也可以一个字符串都不选）

前50%的数据可以接受离线算法，后50%的数据要求在线算法。

【数据规模】

数据分两种，A类数据可以用**离线**的方法来完成，B类数据必须使用**在线**算法。

令len =输入数据中所有出现的字符串总长度

编号	数据类别	数据范围
1	A	$N, M \leq 30, len \leq 500$
2		$N, M \leq 1000, len \leq 10000$
3		
4		
5		$N \leq 50000, M \leq 10^5, len \leq 10^6$
6		
7		
8		
9		
10		
11	B	$N \leq 50000, M \leq 10^5, len \leq 10^6$
12		
13		
14		
15		
16		
17		
18		
19		
20		

by Eternal_Answer

【试题考察点】

维护树形态的数据结构题

【解题报告】

算法 1：

由于前一个串要是后一个串的前缀，那么可以先对串按长度进行排序，用f_i表示前i个串做完，且选了第i个串的时候的最大权值和。转移如下：

$$f_i = \max(f_j + v_i) \quad j < i \text{ 且 } a_j \text{是} a_i \text{的前缀}$$

暴力判断前缀的复杂度是$O(len)$
所以总复杂度是$O(MN^2len)$

算法类型：在线
空间复杂度：$O(len + N)$
时间复杂度：$O(M * N^2 * len)$
期望得分：5~20分

算法 2：

对于算法 1 中判断前缀是用了暴力的方法，其实，可以把串按长度排序后从短到长插入字母树中，那么对于i，合法的转移的字符串j的末端点是在字符串i在字母树插入的路径上的。

那么对于每个字母树的节点上维护以这个点结尾的字符串j的Max_{f_j}，然后插入i的时候只要把路径上的所有Max值取最大值进行转移即可。

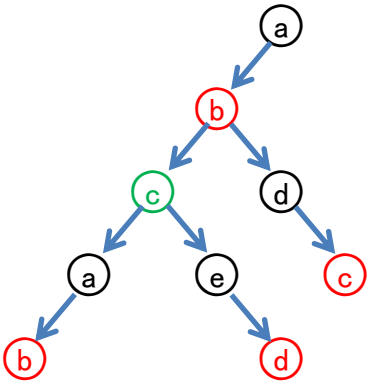
算法类型：在线
空间复杂度：$O(len * 26)$
时间复杂度：$O(M * len)$
期望得分：20分

算法 3：

前两种算法都是每次都重新计算每个串的f值，但每次修改只对一个串修改，所以很多串的f值是不变的，而且f变化的串之间的变化也是有联系的。那么需要我们对题目有更深的理解。

观察算法 2 中的字母树：

假设现在有串：$abcab, abced, abc, abdc, ab$，权值分别为$5,6,-3,7,2$：



图中黑色的节点为普通节点，红色节点为某些字符串的末尾节点并且字符串的权值为非负的，绿色节点为某些字符串的末尾节点并且字符串的权值为负的。在图中，如果对第i个串用end_i即它在字母树中的结束位置来表示这个串，那么串i是串j的前缀当且仅当在字母树中$end_i = end_j$或end_i是end_j的祖先。那么询问的答案相当于求一条从根向下的路径，选择其中的某些节点使得权值和最大，那么我们肯定是把一条路径走到底，去这条路径上所有的非负节点(即图中的所有红色节点)作为这条路径的答案。因此，我们可以对字母树上的每个节点维护它到根的路径上所有非负权值和，然后把所有值中最大的作为答案。

假设没有修改字符串的操作，即只有修改权值操作，那么如果一个点的权值被改成了一个负数，我们可以把这个节点的权值改成 0，然后计算出$delt = v' - v$为权值的变化量，考虑这个节点权值修改的时候影响的点，显然以end_i为根的子树中的所有节点的路径权值和都会加上$delt$。然后自然会想到在树上，一棵子树在dfs序中是连续的一段，那么可以对字母树求出dfs序，然后用线段树进行维护就行了。

现在，有修改字符串的操作，那么我们可以看作是删掉了一个字符串，又加入了一个权值和删去的字符串相同的新字符串，所以我们可以先离线把所有字符串都建好，求出dfs序，然后再重新进行一系列操作，如果碰到把一个字符串删掉，相当于把end_i的权值变成 0。

算法类型： 离线
空间复杂度： $O(len * 26)$
时间复杂度： $O(M * \log len)$
期望得分： 35~50分

算法 4：

在算法 3 中，我们把字母树上的每一个节点都放入了dfs序中，然而其实很多节点都是没有用的，起到作用的只有那些作为字符串末尾的节点，即只有$O(N+M)$个，那么我们可以建一棵只包含有用节点的新的树，每个有用的节点把它在字母树中的祖先中离它最近的有用节点当作新树中的父亲节

点，具体的含义也就是对每一个字符串作为新树的节点，然后一个字符串它的父亲节点是所有其他字符串中它的最长前缀，然后再向算法 3 那样操作。

算法类型： 离线
空间复杂度： $O(len*26)$
时间复杂度： $O(M*\log(N+M))$
期望得分： 50分

算法 5：

对于算法 3 以及算法 4，都是在离线建出树的基础上的算法。

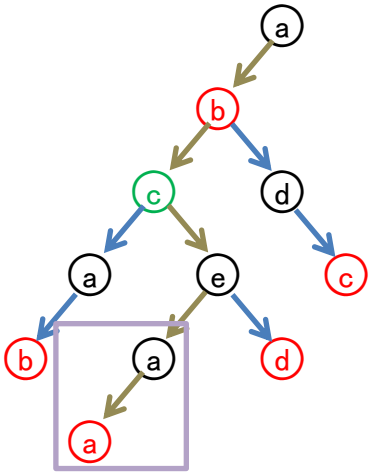
而对于后40%的数据，是要求用在线算法完成的，所以不能用以上算法，而需要我们动态的维护那棵树。

其实，说到底，就是要动态的维护各个点的dfs序以及以它为根的树在dfs序中的范围。

首先，如果删除一个字符串，我们是可以像上面那样保留原节点只是把权值改为 0 即可，那么对树的形态没有产生影响，于是就是插入字符串的问题。对于算法 4，考虑插入一个字符串，那么新插入的字符串的父亲是它在字母树中的最近有用祖先，然而，它的儿子节点会增加很多，那么要维护这棵树就比较困难。因此，只能对算法 3 进行思考。

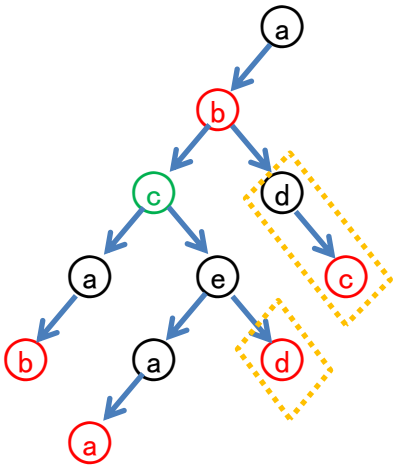
在算法 3 中，我们在字母树中插入一个新的字符串，要不是没有产生任何新的节点，如果产生了，也只会成为一串新的叶子节点，如下图：

假设现在有串：$abcab, abced, abc, abdc, ab$，权值分别为$5,6,-3,7,2$，现在要插入串：$abceaa$，权值为$9$



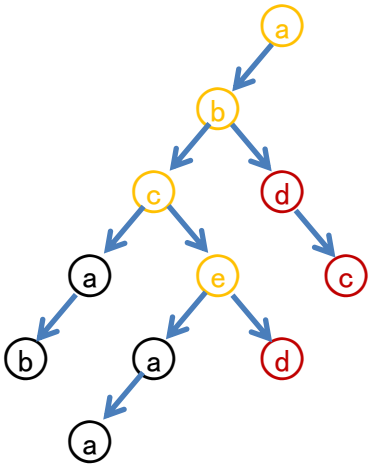
棕色的路径就是插入字符串的路径，矩形框框起来的就是新加入的节点，显然，新加入的只会会成为叶子节点的单独的一支，而不会有其他的点连向他，那么考虑加入这些节点对其他点的dfs序的影响。

首先，在树中加入点是不会影响其他点在dfs序中的相对位置的!!! 其次，观察其他点dfs序的变化情况，还是上面的例子，如下图：



橙色虚线框框起来的点的dfs序都会加 2，其他的不变，而橙色框起来的部分就是dfs序在插入节点后的所有点，也就是说是对一段后缀加新节点数。

然后，再来观察每个点为根的子树的dfs序范围的变化情况，还是这个例子，如下图：



图中，以黄色的点为根的子树在dfs序中的区间右端点都加了 2，而以棕色的点为根的子树在dfs序中的区间则是整体向后移了两位。对于黄色的点，都是插入路径上的点，那么可以暴力修改，而对于棕色的点，都是dfs序相对位置在新插入的点的所有点，所以对一段后缀传一个标记即可。

又由于要随时的插入点，并且点与点之间的相对位置始终不变，所以可以用一棵平衡树来维护字母树的dfs序，每次操作都是在平衡树中插入新的节点后对一段后缀的值进行操作。

算法类型： 在线
空间复杂度： $O(len * 26)$
时间复杂度： $O(len * \log len)$
期望得分： 75~100分

by Eternal_Answer

算法 6：

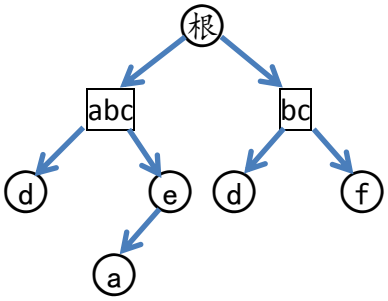
在算法 5 中，对于每个字母都要建一个节点，而正如算法 4 所说，并不是每个节点都是有用的，只有那些作为末尾的节点是有用的，然而在算法 5 中，已经对算法 4 的做法进行了否定，原因是会改变很多树边。

对100%的数据，$len = 10^6$, 算法 5 的复杂度是$O(len * \log len)$, 会超时。

因此，只能对算法 4 进行改进，适当增加一些节点，使得加入新串的时候只会向算法 5 那样的对一个后缀进行操作。

那么，这里要使用到一种特殊的字母树：压缩型字母树。

顾名思义，就是把连续一段字母缩成一个小节点，一段字母分裂的前提条件是有一段字母成为了末尾节点串，或者是这段字母在字母树中出现了分支，举个例子，有字符串：$abcd, abce, abcea, bcd, bcf$：



用框框起来的节点是由于出现出现分支而分裂的，而圆形的点是因为是字符串末尾点而成为一个节点的。

当一个字符串插入的时候，最多只会分裂一个节点以及新增一个节点，所以总结点数是$2 * (N + M)$的。

由于每次只新增一个节点和分裂一个节点，所以每次平衡树里最多只需插入 2 个节点，所以，复杂度得到了保证。

算法类型： 在线

空间复杂度： $O((N + M) * 26 + len)$

时间复杂度： $O\big((N + M) * \log(N + M)\big)$

期望得分： 100分

【小结】

这道题考察了选手对题目的转化能力，并通过观察发现，从而找到正确的算法，是一道比较好的数据结构题，对选手的代码能力也有所要求。在这套互测题中属于中等偏难题。

by Eternal_Answer