Divljak 解题报告

福州一中 董克凡

1 试题来源

coci2015 Divljak

BZOJ地址：http://www.lydsy.com/JudgeOnline/problem.php?id=3881

2 试题大意

给出n个模式串s_i，共进行m次操作，每次操作有两种形式：

- 插入一个新的文本串t_i
- 询问当前第i个模式串被多少个文本串包含

$n, m \leq 10^5, s \leq 2 * 10^6$，其中$s$为文本串与模式串各自长度之和。

3 算法介绍

3.1 AC自动机维护串信息

首先注意到模式串输入之后就已经确定，那么考虑从模式串入手。对于模式串建立AC自动机，并且构造出fail树，每个模式串在fail树上就对应着一个节点。对于操作一，可以直接把文本串在AC自动机上跑一下，把经过的节点标记下来。那么对于模式串s_i，如果s_i的对应节点在fail树的子树内一定至少有一个节点被标记过，那么s_i就被当前这个文本串包含。所以每一次操作一，只需要把满足子树内有标记的的所有模式串s_i的答案+1，在操作二的时候直接输出答案即可。

考虑需要进行+1操作的所有模式串，这些节点就一定在所有标记节点到根的链的并上面。故需要用一个数据结构支持对某些点到根的链取并+1。

3.2 数据结构维护链并

3.2.1 算法一

一种经典做法是，用fail树的dfs序来统计答案。首先将所有的标记节点去重，然后按照dfs序进行排序，按照dfs序的顺序处理，每次考虑整个结构（即树链的并）的增量，当在dfs序后新增加一个节点时，增加的路径即当前点到其与前一个标记节点的lca的一条链，那么然只需要把这一条链上的点答案+1。进行书上差分之后可以转化为在当前点到根路径+1，lca处-1。

故可以使用树状数组+dfs序维护。转化为单点修改，询问一个点的值时询问这个点的子树中的权值和。

这样维护时间复杂度为$O(s*log(s))$

3.2.2 算法二

这里我使用了另一种方法来维护树链的并。

考虑link-cut-tree的access操作，这个操作的的作用就是依次把一个点到根路径上的点放到一棵平衡树中。可以将access操作稍加修改用以维护树链的并：把多个access操作放在一起考虑，每次access 做到已经做过的节点就停止，而不是一直做到根，用一个时间戳维护每个节点是否已经做过。那么在时间戳更新之前，每次进行access 操作就能避免更新已经被做过的节点。也就是说，对于一个操作一，在经过的每一个节点调用这个修改后的access操作，就相当于直接把加入了这个节点以后产生的**增量**放在了一颗平衡树中。直接在这棵平衡树中进行+1操作即可。在一整个操作一结束之后再更新时间戳。这样所有的需要更新的节点就只操作了一遍。

使用这种方法维护，不需要对于经过的节点按照dfs序排序。而且对于不会被询问到的节点，修改操作同样是不需要执行的，故可以把这些点缩起来而不影响答案，只保留每个模式串所在的一个节点，就能将lct的规模限制在$O(n)$级别。这样时间复杂度就是$O(s*log(n))$

用这种方式维护一些点到根路径的链并，可以非常自然地避免重复计算的问题，所以可以不需要支持修改操作有逆。而且这样操作可以不需要对要修改的进行离线排序，可以在线进行。对应到原问题就是，可以在线在当前插入的文本串末尾插入一个字符（对应于进行一次access操作），或者宣称这个文本串

已经结束（对应于清空时间戳），以及询问（对应于平衡树上的查询）。

这个方法虽然使用了lct 这个大常数的数据结构，但是其效率比经典算法要快近一倍（在BZOJ 的提交情况上可以看出）。除了复杂度更优，另一个重要原因就是，虽然执行了s 次access 操作，但这些操作并没有把一整条链都访问一遍，事实上每次access 操作的时间复杂度基本都是不满的。所以在$2 * 10^6$的数据范围下仍能快速出解。从时空复杂度的角度来说，算法二完全优于算法一。

4 复杂度分析

时间复杂度：$O(slog_2n)$

空间复杂度：$O(s+n)$