字符串学习笔记(1) 基础概念与kmp前置

严格鸽
柚子厨/萝莉控/ACM银牌

已关注

定义字符串 S，下标从 1 开始，其长度为 $|S|$， $S[i]$ 表示字符串 S 的第 i 个字符。

$S[L,R]$ 表示 $S[L],S[L+1],S[L+2]....S[R]$ 的构成的字符串，也就是子串。

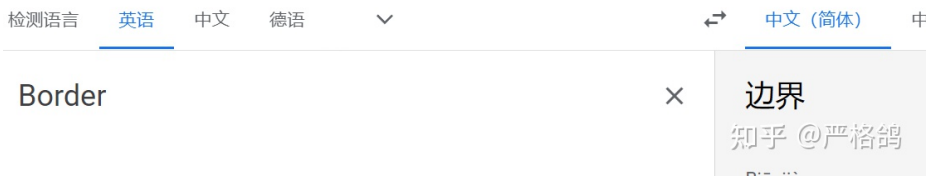
注：子串要求连续，子序列不需要。

pre_S 表示 S 的前缀， $pre_S[i]=S[1,i]$

suf_S 表示 S 的后缀， $suf_S[i]=S[n-i+1,n]$ 无特殊说明 $n=|S|$

Border

对于某个字符串，如果如果前缀和后缀完全相同（ $pre_S[i]=suf_S[i]$ ），则称前缀和后缀字符串(或者长度)为这个 S 的一个Border。



一个字符串可以有多个Border

比如 $bbabbab$,其Border有
b
$bbab$
$bbabbab$(本身)

Border不具有二分性。

什么是二分性，比如回文子串，一定存在一个 x ，使得任意长度小于等于 x 的回文都会存在，而一定不存在长度大于 x 的回文。也就是如果我们二分长度并且check的话，回文是这样的 $[1,1,1,1,1,0,0,0,0,0]$

但是Border就没有这个性质，它比如上边的字符串 $bbabbab$ 就是 $[1,0,0,1,0,0,1]$

（如果没有特殊说明，Border是不考虑为本身的情况的。

显然呢，我们可以枚举前缀和后缀然后字符串哈希 $O(n)$ 求出来（

但是哈希就没意思了，所以我们还是正儿八经的求吧qwq

Border有一个性质，传递性。即

> S 的 Border 的 Border 也是 S 的 Border

例如下图是一个字符串，有长度为6的Border。

那么对于长度为6的前缀来说，如果它还有个长度为2的Border

同理后缀也会有相同的Border

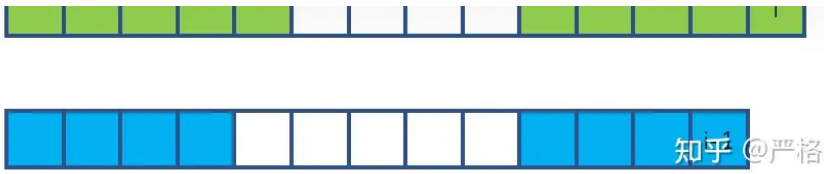
所以这个2也是整个 S 的一个Border。

我们定义 $next[i]$ 为前缀 $pre_S[i]$ 的最大Border的长度。

显然有 $next[1]=0$

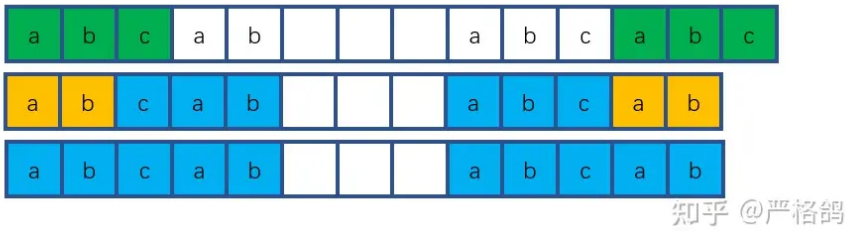
有一个性质

> 考虑 pre[i] 的所有（长度大于 1 的）**Border**，去掉最后一个字母，就会变成 pre[i − 1] 的 Border。



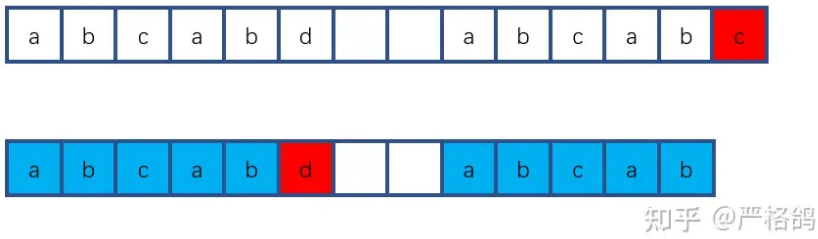
(注意不是变成 $pre[i-1]$ 的最大Border

比如下图 $pre[i]$ 的最大Border为 abc，去掉 c 后变成 ab，是 $pre[i-1]$ 的一个Border但是不是最大的。



所以我们如何求出 $next[i]$ 呢？我们可以遍历 $pre[i-1]$ 的所有的Border，然后看看其后面的一个字符是否等于 $s[i]$。

比如我们可以先比较 $pre[i-1]$ 的最大的Border。

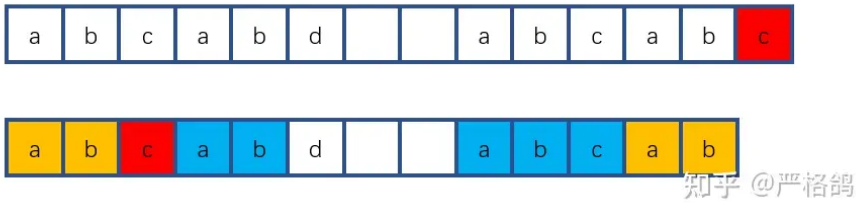


不相等，所以我们要换一个Border了。

$pre[i]$ 的下一个Border可以通过 $next[next[i-1]]$ 获取，因为

> S 的 Border 的 Border 也是 S 的 Border

也就是 $pre[i-1]$ 的最大Border是 $pre[next[i-1]]$，其最大Border $next[next[i-1]]$ 也是 $pre[i-1]$ 的一个Border。



这个时候，我们发现匹配上了，所以有 $next[i]=2+1=3$

这样做法可以类比我们在分解质因数中，先处理出 $minp[i]$，i 的最小的质因子，然后一步步的除以到1。

```
        int p = minp[n];
        int cnt = 0;
        while (n%p == 0) {
            n /= p;
            cnt++;
        }
    }
}
```

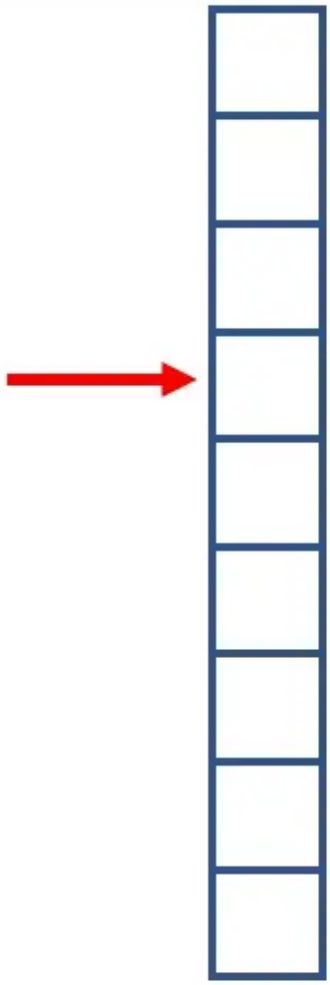
这样我们也可以写出我们的代码，先让 $next[i] = next[i-1]$，看看 $s[next[i]+1] = s[i]$ 吗，如果不等于，就执行 $next[i] = next[next[i]]$ （往会跳，类似跳链表）

一直执行到 $next[i] = 0$ 或者 $s[next[i]+1] = s[i]$ 为止，然后再检查一下是否相等（除去 $next[i] = 0$ 的情况

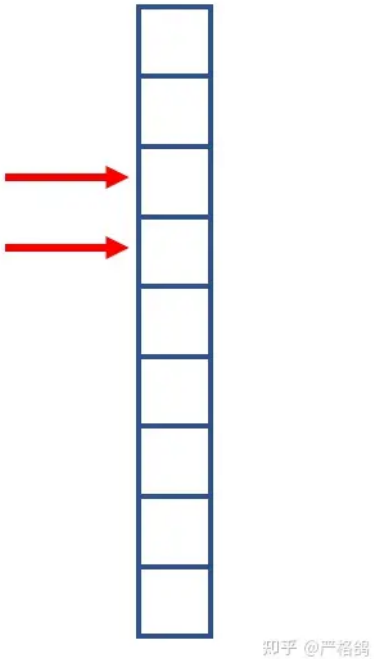
```
void get_next(string &s) {
    nxt[1] = 0;
    for (int i = 2; i < s.length(); i++) {
        nxt[i] = nxt[i - 1];
        while (nxt[i] && s[i] != s[nxt[i] + 1])
            nxt[i] = nxt[nxt[i]];
        if (s[nxt[i] + 1] == s[i])nxt[i]++;
    }
}
```

可能看上去，复杂度为 $O(n^2)$,但是我们可以这样考虑。

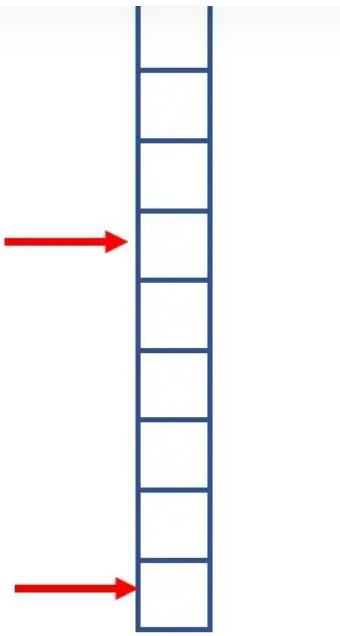
$next[i]$ 的值继承 $next[i-1]$，最多为 $next[i-1]+1$



也就是对于这个箭头，每次最多向上移动一格。

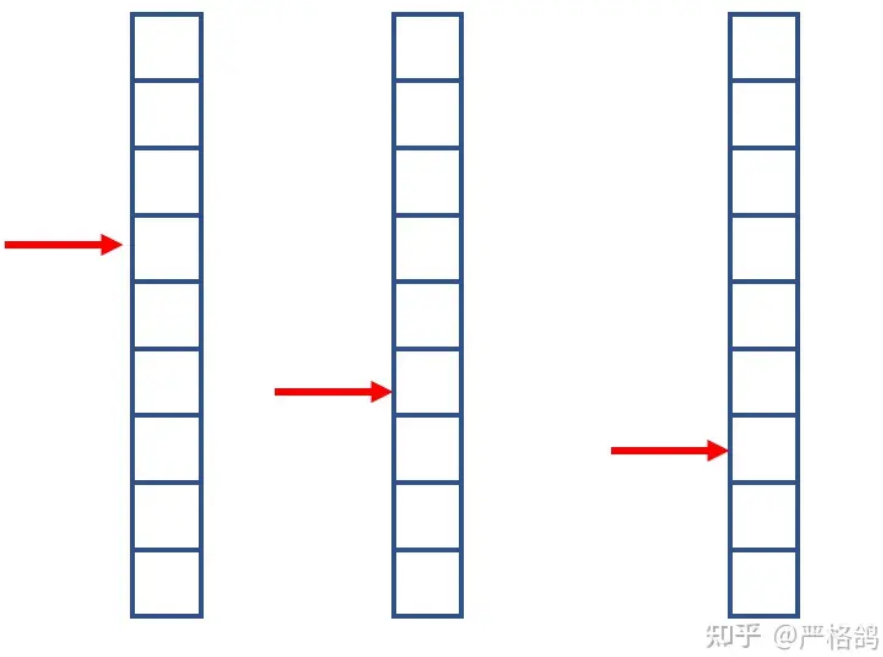


或者向下移动，可以移动到底部。



时间复杂度和箭头移动的距离相同。

箭头可以向下移动任意距离，但是最多向上移动一格。



所以，总的移动距离为 $O(n)$ 时间复杂度也是 $O(n)$ 的。

同样的分析方式：严格鸽：Codeforces Round #254 (Div. 1) C(线段树)

那么kmp就是利用最大的Border来优化匹配的，不过这个就是后面的内容了。

发布于 2022-07-23 09:37

字符串　ACM 竞赛　OI（信息学奥林匹克）

我怎么这么装呢
回文子串，一定存在一个x，使得任意长度小于等于x的回文都会存在，而一定不存在长度大于x的回文，字符串ABAC存在长度3，1的回文子串，不存在长度2的回文子串，这个二分性有点问题吧🤔🤔🤔，ygg应该指的是回文串的半径有二分性吧🥶🥶🥶
2022-11-02 · IP 属地山东 回复 喜欢

严格鸽 作者
是的
2022-11-02 · IP 属地四川 回复 1

我怎么这么装呢 ‣ **严格鸽**
🤐
2022-11-02 · IP 属地山东 回复 喜欢

少说多看
蹲一个BM，BM的造表预处理太难了😭
2022-07-23 · IP 属地江苏 回复 喜欢

严格鸽 作者
等我学到那里qwq
2022-07-23 · IP 属地四川 回复 喜欢

TAdmin
感谢😔
2022-07-23 · IP 属地湖南 回复 喜欢

张大老实
😭这图也太用心了
2022-07-23 · IP 属地北京 回复 喜欢

严格鸽 作者
qwq
2022-07-23 · IP 属地四川 回复 喜欢

我借此火
格鸽会发暑假牛客和杭电多校的题解嘛
2022-07-23 · IP 属地浙江 回复 喜欢

严格鸽 作者
会的，打过的就发qwq
2022-07-23 · IP 属地四川 回复 喜欢

发布一条带图评论吧

文章被以下专栏收录

字符串学习笔记
为了不在卡字符串的题目，勇敢勇敢我的朋友

推荐阅读

[C#.NET 拾遗补漏]01：字符串操作
精致码农 发表于C#.NE...

JAVA精讲（七）字符串处理
Shersty

盘点世界上现存的数字符号系统
从小时候起，老师都一直说，0、1、2、3、4、5、6、7、8、9这些阿拉伯数字是世界通用……无可否认这构成了我很长一段时间对数字符号系统的根本认识。 直到后来由于一些原因经常要接触一些外...
北极星飞过什切青

左右用R在 符串合并!
杜雨