链滴

KMP 其实也没那么难

作者：zouchanglin

原文链接：https://ld246.com/article/1564282798642

来源网站：链滴

Knuth-**M**orris-**P**ratt 字符串查找算法，简称为 **KMP算法**，KMP是我们经常听到的一种字符串匹配法。KMP算法听起来很难，但是如果真正明白它的匹配过程其实不难，接下来看看KMP究竟是如何配字符串的?

假设现在有如图所示两个字符串， 图表所列的是匹配串的所有子串，这个不难理解

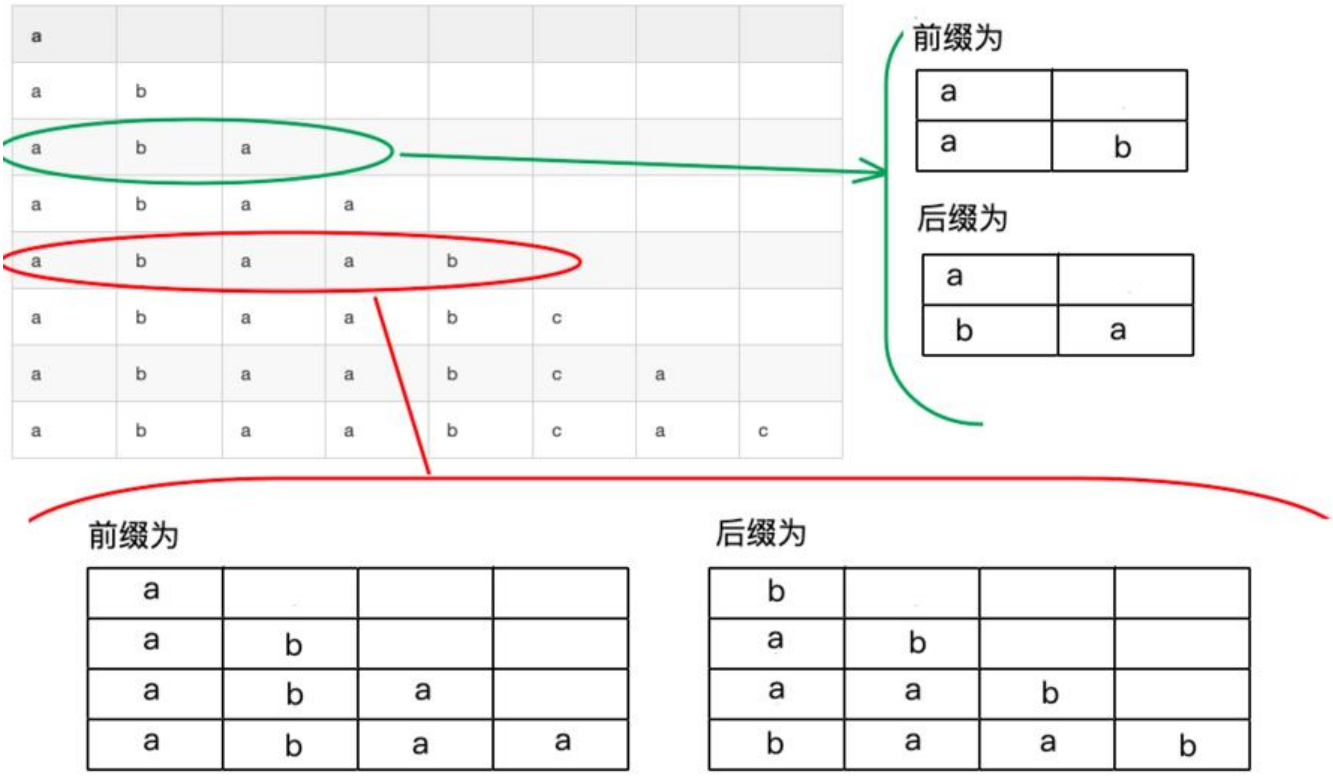
a							
a	b						
a	b	a					
a	b	a	a				
a	b	a	a	b			
a	b	a	a	b	c		
a	b	a	a	b	c	a	
a	b	a	a	b	c	a	c

两个概念：前缀和后缀

前缀 指除了最后一个字符以外，一个字符串的全部头部组合；

后缀 指除了第一个字符以外，一个字符串的全部尾部组合。

比如对于第三行和第五行



a			
a	b		
a	b	a	
a	b	a	a

b			
a	b		
a	a	b	
b	a	a	b

原文链接：KMP 其实也没那么难

求出每一个子串中前缀和后缀相等的部分的最大长度，比如第5行求出来最大长度为2

a	b	a	a	b			
a	b	a	a	b	c		
a	b	a	a	b	c	a	
a	b	a	a	b	c	a	c

求前缀后缀最大公共元素长度

求得原匹配串的所有子串对应的各个前缀后缀的公共元素的最大长度表：

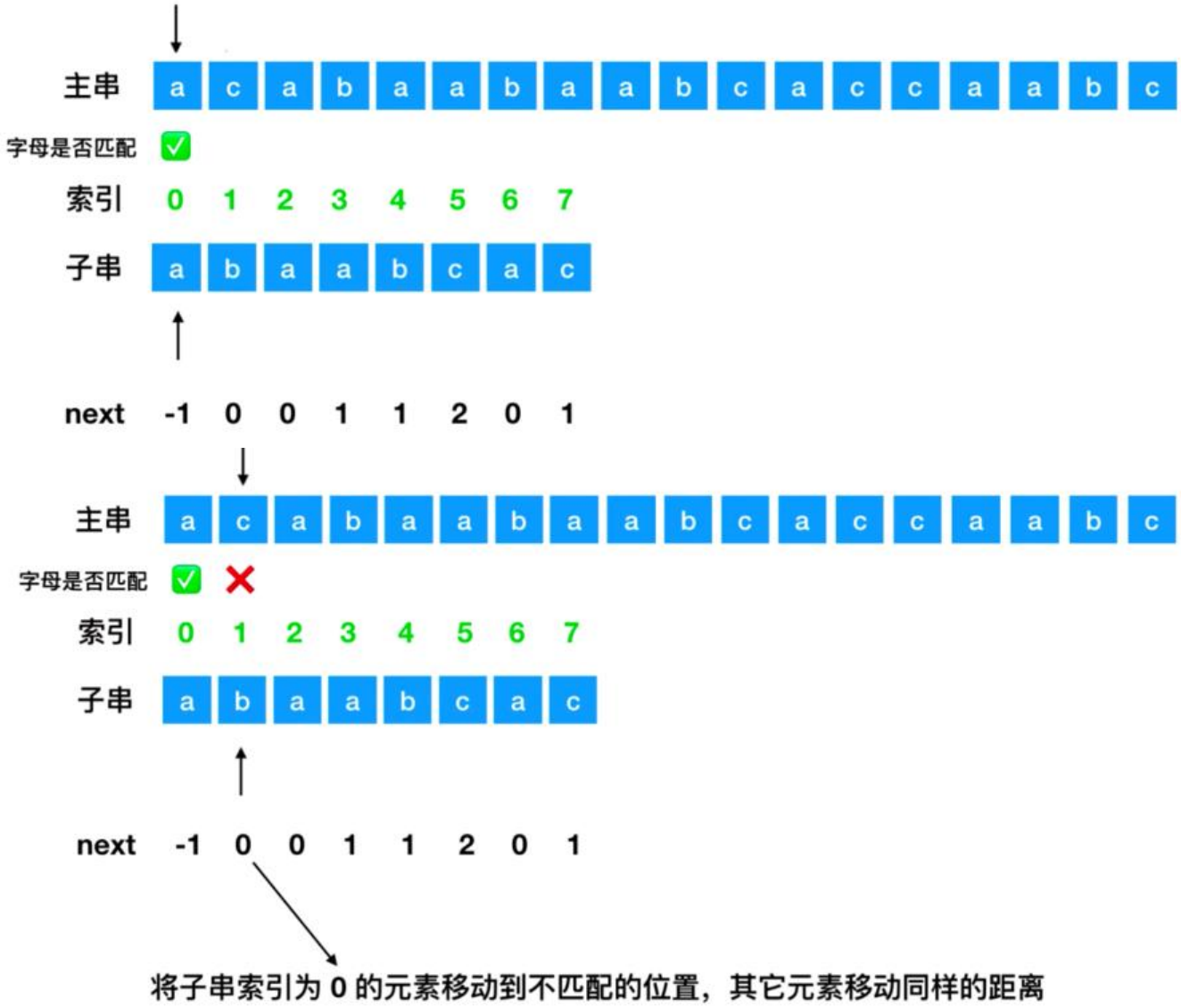
序号	1	2	3	4	5	6	7	8
匹配串	a	b	a	a	b	c	a	c

0	a							
0	a	b						
1	a	b	a					
1	a	b	a	a				
2	a	b	a	a	b			
0	a	b	a	a	b	c		
1	a	b	a	a	b	c	a	
0	a	b	a	a	b	c	a	c

接下来需要一个next数组相当于最大长度值数组整体向右移动一位，然后把0号位置赋为-1，多出来一位已经用不着了，直接划掉。

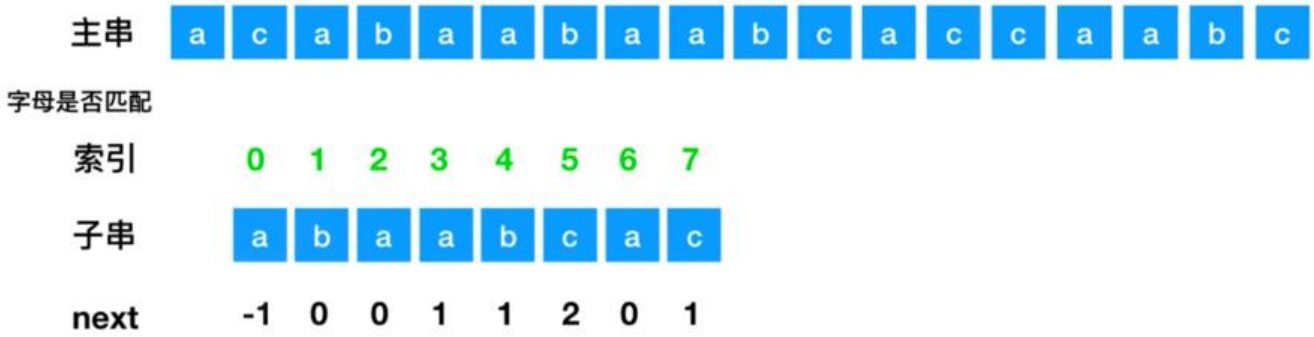
原文链接：KMP 其实也没那么难

接下来开始匹配



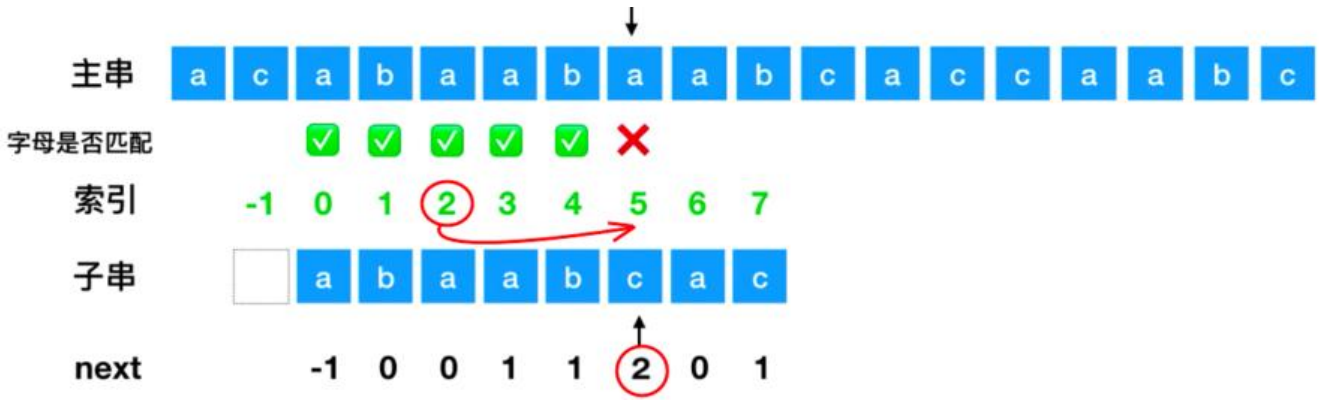
原文链接：KMP 其实也没那么难

如果字母匹配直接都朝着右边走，如果不匹配就需要把子串索引为0的元素移动到不匹配的位置，其的元素移动同样的距离，如下图所示：

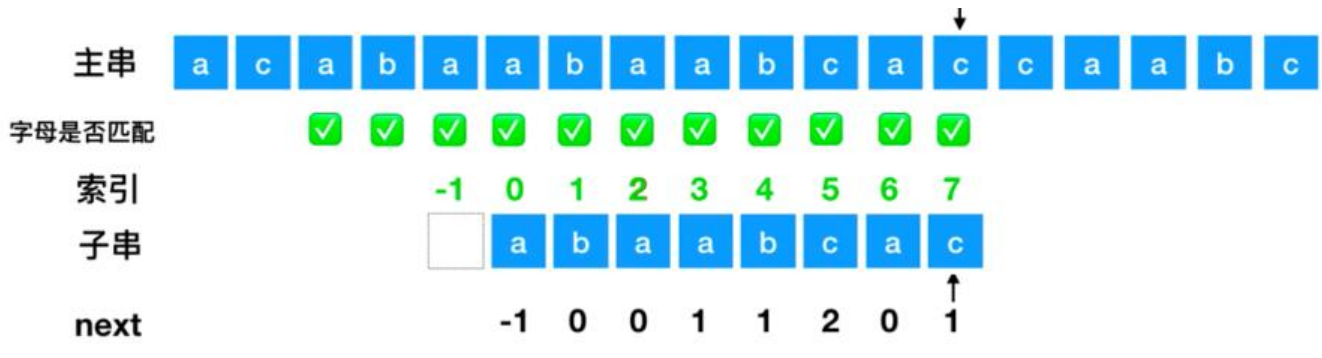


现在不匹配的位置是next=-1，所以需要把匹配串中索引为-1的元素移动至不匹配的位置，其他元素动同样的长度

现在匹配到5号位置如图所示发生了对应字母不一致，此时next数组对应值为2，所以匹配串中的索引2的应该移动至发生对比不一致的地方：



这样便可以一直对比下去，由此可见KMP的高效匹配原来是这样完成的！



原文链接：KMP 其实也没那么难

所以对KMP算法总结如下几点：

1、先求出匹配串所有子串

2、求每一个子串中前缀和后缀相等的部分的最大长度

3、这样便求出了next数组，以-1开头的next数组是更容易理解的方式

4、有了next数组剩下的就是匹配了，匹配时注意遇到不一样的字母时先看next数组对应值，此值就匹配串索引处应该向不匹配位置挪动的字母索引，简单说就是：匹配串中的那个字母应该挪动到不匹处由next数组决定