[**后缀数组**](http://kmplayer.javaeye.com/blog/559721)

**文章分类:**[**C++编程**](http://www.javaeye.com/blogs/category/cpp)

好难,慢慢学.   
  
1,定义:   
  
(1)约定一个字符集Σ和一个字符串 S，设 len(S)=n，且 S[n]='$'，也就是说 S 以一个特殊字符'$'结尾，并且'$'小于Σ中的任何一个字符。除了 S[n]之外，S 中的其他字符都属于Σ。对于约定的字符串 S，从位置 i 开头的后缀直接写成 Suffix(i)。   
例如:S=mississippi+'$'   
(2)后缀数组    
后缀数组SA 是一个一维数组，它保存 1..n 的某个排列   
SA[1],SA[2],...SA[n]，并且保证  Suffix(SA[i])<Suffix(SA[i+1]),1≤i<n。也就是将S 的 n 个后缀从小到大进行排序之后把排好序的后缀的开头位置顺次放入SA中。   
例如:   
T1  = mississippi$ = txt   
T2  = ississippi$   
T3  = ssissippi$   
T4  = sissippi$   
T5  = issippi$   
T6  = ssippi$   
T7  = sippi$   
T8  = ippi$   
T9  = ppi$   
T10 = pi$   
T11 = i$   
T12 = $                
排序后结果为:   
T12 = $   
T11 = i   
T8  = ippi   
T5  = issippi   
T2  = ississippi   
T1  = mississippi   
T10 = pi   
T9  = ppi   
T7  = sippi   
T4  = sissippi   
T6  = ssippi   
T3  = ssissippi   
得到:SA=[12 11 8 5 2 1 10 9 7 4 6 3]   
(3)名次数组    
名次数组 Rank=SA-1，也就是说若 SA[i]=j，则 Rank[j]=i，不难看出 Rank[i]保存的是 Suffix(i)在所有后缀中从小到大排列的“名次”.   
得到:Rank=[6 5 12 10 4 11 9 3 8 7 2 1]   
  
2,如何构造后缀数组？   
等价于得到SA和Rank.改进的核心正在于此.   
(1)最直接最简单的方法当然是把 S 的后缀都看作一些普通的字符串，按照一般字符串排序的方法对它们从小到大进行排序。   
分析:这种做法是很笨拙的，因为它没有利用到各个后缀之间的有机联系，所以它的效率不可能很高。即使采用字符串排序中比较高效的 Multi-keyQuick Sort，最坏情况的时间复杂度仍然是O(n2)的.   
(2)倍增算法(Doubling Algorithm)   
充分利用了各个后缀之间的联系，将构造后缀数组的最坏时间复杂度成功降至 O(nlogn).   
  
单独讲解倍增算法   
1,定义u的k-前缀: