后缀数组学习笔记

|  |
| --- |
| 首先定义：  rank[i]为S[i..n]这个后缀是所有后缀里第几大的。  sa[i]为第i大的后缀是哪一个  因为后缀不可能相等，所以rank[i]和sa[i]唯一，且满足sa[rank[i]]=i  我们可以通过类似多关键字排序+构造Sparse Table的方法使构造这两个数组的时间复杂度为O(nlgn)  注意中间过程用基数排序，否则用qsort的话会变成O(n(lgn)^2)  构造完这个，我们还应该加一个height数组，height[i]表示lcp(i-1,i)。  其中i表示排名，即rank[sa[i]]=i  这样就相当于后缀树中两个相邻节点的lca了。  至于如何构造height[i]，这里有我写的code。  其中使用了一个定理：  height[i]>=height[sa[i]-1]-1（看懂什么意思以后，证明显然）  使用这个定理以后构造height的复杂度降到了O(n)  void lcp(){      memset(height,0,sizeof(height));      for (int i=1;i<=n;i++){          if (rank[i]==1) continue;          int st,j,k;          st=max(height[rank[i-1]]-1,0);          j=i+st;          k=sa[rank[i]-1]+st;          while (j<=n && k<=n && s[j]==s[k]){                st++;                j++;                k++;          }          height[rank[i]]=st;      } }  另外再有一个定理lcp(i,j)=min(lcp(k-1,k)) (k∈(i,j])  姑且叫他lcp(i,j)定理  然后我们就可以用它解决很多问题。  总的来说有：  1、多串匹配。  复杂度O( (T+lgN)\*M)   其中T为模式串长度，N为主串长度，M为模式串个数。  主要思想就是二分，然后利用height数组，和lcp(i,j)定理。  2、最长公共前缀。  例题：<http://hi.baidu.com/edwardmj/blog/item/a69c46560990d5143a2935fb.html>  就是max(height[i])  3、最长回文子串。  设给定S'，求其最长回文子串。（设其长度为n）  令S''为S'的倒序串(即S''[n-i+1]=S'[i])  然后S=S'+'#'+S''。（即把两串连接）（长度为2\*n+1）  然后枚举中心i(1<=i<=n)，其镜像中心为2\*n+2-i  然后就是求最多能延伸多长。  即求lcp(i,i')。利用lcp(i,j)定理加上Sparse Table（RMQ）就可以做到在O（1）的复杂度内解决延伸问题。  所以，求最长回文子串的部分，复杂度为O(N)  总复杂度O(nlgn)（加上构造时间）  4、利用分组思想做一些意想不到的操作.  例如：<http://hi.baidu.com/edwardmj/blog/item/e5105b8d97842ef1513d92ed.html> |