

「琐记」字符串复习

Jiayi Su (ShuYuMo)

2021-01-19 15:35:06

后缀数组的一次复习和整理，大部分精简自 oi-wiki。

「琐记」字符串复习

SA 后缀数组

整理自 OI-Wiki.

应用

两字符串最长公共前缀长度

$$\min\{\text{height}[\text{rk}[i] + 1, \text{rk}[j]]\}$$

比较字符串大小关系 不妨设： $A = S[a..b], B = S[c..d]$

$$A < B \iff \begin{cases} |A| < |B|, & |\text{LCP}(A, B)| \geq \min(|A|, |B|) \\ \text{rk}[a] < \text{rk}[b], & \text{otherwise} \end{cases}$$

不同子串的数目

$$\frac{n(n+1)}{2} - \sum_{i=2}^n \text{height}[i]$$

出现至少 k 次的子串 最大长度 如果某个子串出现了 k 次，那么一定有连续（后缀排序后） k 个后缀的 LCP 是其超串。

$$\max_i \left\{ \min_{j=i+1}^{i+k} \{\text{height}[j]\} \right\}$$

最长 不重叠出现两个次以上的子串 考虑二分一个值 $|S|$ 表示子串长度。考虑根据 $|S|$ 把 Height 数组断成若干段。每一段内的 height 的值都 $\geq |S|$ ，检查每一段内的所有值的下标，判断是否重复。

「NOI2015」品酒大会 如果从大到小枚举 r ，后缀排序后，合法的串一定挨在一起，并且随着 r 的减小，合法的区间会扩大、合并。

方案和最大值都好维护，同时维护最大值、次大值/最小值、次小值（存在负数），就可以在每次合并的过程中更新答案。

最大、次大、最小和次小不好维护...

- 可能某个值不存在，不能使用 -1 表示不存在，可以考虑特殊化一下，特别定义某个值表示不存在，不是很可能冲突。
- 合并的时候讨论比较麻烦，只需要考虑把这些数字都放入一个 vector，排序去重后一一取出即可，可以避免讨论。

「AHOI2013」差异 给出长度为 n 的字符串 S ， T_i 表示从字符 i 开始的后缀，求：

$$\sum_{1 \leq i \leq j \leq n} |T_i| + |T_j| - 2|\text{LCP}(T_i, T_j)|$$

因为 $|\text{LCP}(T_i, T_j)|$ 直接对应后缀排序里的值，考虑对这个值单调栈求一下取值为其的区间，算贡献即可。

模板

```
const int _ = 1e6 + 100;
int sa[_], rk[_], ht[_];
void Suffix_Array(char *S, int n){
    static int oldrk[_], cnt[_], id[_], px[_];
    int m = max(n, 300), i, p, T, k;
    for(i = 1; i <= n; i++) cnt[rk[i] = S[i]] ++;
    for(i = 1; i <= m; i++) cnt[i] += cnt[i - 1];
    for(i = n; i >= 1; i--) sa[cnt[rk[i]] - 1] = i;
    for(T = 1; T <= n; T <= 1){
        memset(cnt, 0, sizeof(int) * (m + 5));
        memcpy(id, sa, sizeof(int) * (n + 5));
        for(i = 1; i <= n; i++) cnt[px[i] = rk[id[i] + T]] ++;
        for(i = 1; i <= m; i++) cnt[i] += cnt[i - 1];
        for(i = n; i >= 1; i--) sa[cnt[px[i]] - 1] = id[i];

        memset(cnt, 0, sizeof(int) * (m + 5));
        memcpy(id, sa, sizeof(int) * (n + 5));
        for(i = 1; i <= n; i++) cnt[px[i] = rk[id[i]]] ++;
        for(i = 1; i <= m; i++) cnt[i] += cnt[i - 1];
        for(i = n; i >= 1; i--) sa[cnt[px[i]] - 1] = id[i];

        memcpy(oldrk, rk, sizeof(int) * (n + 5));
        for(p = 0, i = 1; i <= n; i++){
            if(oldrk[sa[i]] == oldrk[sa[i - 1]]
                && oldrk[sa[i] + T] == oldrk[sa[i - 1] + T])
                rk[sa[i]] = p;
            else rk[sa[i]] = ++p;
        }
        if(p == n) break;
    }
    for(k = 0, i = 1; i <= n; i++){
        if(k) k--;
        while(S[i + k] == S[sa[rk[i] - 1] + k] && k <= n) k++;
        ht[rk[i]] = k;
    }
}
```