Longest Substring
&
Memorization Game

数据结构教学团队 丁铭

Longest Substring 题目描述

- T组任务
- 给定字符串s和最少重复次数m，求出s中出现次数大于等于m且长度最长的子串
- 输出长度和最右子串的位置

$$m \leq Len(s) \leq 40000, s[i] \in \{a, b, ..., z\}$$

蛮力算法

- 首先考虑到长度不固定，如果枚举起始点、长度就会超时
- 但是只要知道最长的出现m次子串

- 如果存在长度为L的出现m次子串，那么L-1一定存在！
  - 二分答案
  - 只考虑长度为mid的子串，看它们中是否有超过m次的

- 问题转化为有n=Len(s)个字符串，是否存在超过m个相同？

蛮力算法

- 蛮力枚举，两两匹配
  - $O(n^2 mid)$，加上二分总复杂度$O(n^3 \log n)$
- 计算hash值，假设通过hash表可以O(1)判断该项出现次数
  - 计算hash值O(n*mid)，扫描判断出现次数O(n)
- 考虑到n个字符串之前有关系的，后一个仅仅是前一个“掐头续尾”
  - 可以设计一个O(1)就可以有前一个串的hash计算的hash函数
  - 该问题O(n)解决，总时间O(nlog n)

被忽略的细节

- 但是，为什么只有85分啊！

- 我们之前认为hash可以O(1)实现判断出现次数，但是朴素的hash实现中，如果hash值相同需要比较原串
  - 如果每次都有比较的话时间复杂度$O(n^2 \log n)$

- 那么干脆直接认为hash值相同，两个串就相同好了……
  - 可能出错（生日悖论）

错误率与时间消耗的折中

- 容易想到如果我们设计若干个hash函数，他们都相同认为这两个串相同
  - 错误率被控制
  - 时间仅增加常数倍

- 助教并没有办法构造出让你出错的数据
  - 因为hash是你设计的，而数据是面向所有人的

其他算法

- 那么有没有错误率严格为0的做法呢?
- 提供一种供有兴趣的同学了解:
  - 后缀suffix[i]是指原字符串中i-n位置的子串
  - 后缀数组sa[i]指字符串比较中，排名为i的后缀的起始位置
  - 排名数组rank[i]指suffix[i]的排名
  - 以上两者可以通过倍增法O(nlog n)或DC3算法O(n)求出
  - Height[i]代表排名为i的后缀suffix[sa[i]]与排名为i-1的后缀suffix[sa[i-1]]之前的最长前缀长度，H[i]表示第i位置的后缀与排名为rank[i]-1的后缀的最长前缀，根据后者性质H[i]≥H[i-1]-1可以将两者O(n)求出
  - 根据Height[i]可以通过扫描统计长度为m窗口的最小值求出结果，同时注意记录最右点位置。总时间复杂度O(nlog n)或O(n)

Memorization Game

- 对序列进行各种操作，是一道传统的数据结构题

  - A x y D //在{Ax...Ay}上加上D。例如，在{1,2,3,4,5}上执行"A 2 4 1"，则结果为{1,3,4,5,5}。
  - R x y //将{Ax...Ay}进行逆转。例如，在{1,2,3,4,5}上执行"R 2 4"，则结果为{1,4,3,2,5}。
  - T x y K //将{Ax...Ay}循环右移K位。例如，在{1,2,3,4,5}上执行"T 2 4 2"，则结果为{1,3,4,2,5}。
  - I x P //在Ax后插入P。例如，在{1,2,3,4,5}上执行"I 2 4"，则结果为{1,2,4,3,4,5}。
  - D x //删除Ax。例如，在{1,2,3,4,5}上执行"D 2"，则结果为{1,3,4,5}。
  - M x y //在{Ax...Ay}中查询最小值。例如，在{1,2,3,4,5}上执行"M 2 4"，则结果为2。

$$n, m \leq 10^5$$

选取数据结构

- 虽然提示上写了“线段树”，但是其实线段树并不合适……（抱歉）
- Splay更合适一些
  - 伸展树如何维护区间操作?
  - 维护的序是左右顺序，每个节点维护该子树对应区间的信息
  - 在本题中需要维护子树大小、最小值、翻转标记、区间增加标记
  - 访问节点时将标记下压到子节点（同时考虑交换左右儿子和增加儿子的最小值）
  - 旋转时重新统计最小值、子树大小

Splay如何选取区间？

- 将区间左端点的前驱旋转到根、将区间右段点的后继旋转到根的右节点，那么根的右节点的左节点对应的子树就是区间的全部
  - 可能需要哨兵
- 删除操作也可以这样，或者用其他类似方法

操作

- Add 找到区间直接打加合标记并修改最小值即可
- Reverse 找到区间直接打翻转标记
- Transpose 首先对区间长度取模，实际是将一段区间插入另一位置，仍然找到对应子树摘除、插入即可
- Insert Delete Minimum 传统操作

谢谢大家

祝大家考试周顺利，RP++

Q&A