字符串相关算法

Jiezhong Qiu

September 8, 2011

*Hash Table*是一种高效的数据结构，一般用于判重或判等价。简要地说，Hash是信息的一种压缩，把一个字符串，一棵树等复杂的信息压缩成少量的数字，从而方便比较和存储。

常用的Hash函数

- RK函数

*Hash Table*是一种高效的数据结构，一般用于判重或判等价。简要地说，Hash是信息的一种压缩，把一个字符串，一棵树等复杂的信息压缩成少量的数字，从而方便比较和存储。

常用的Hash函数

- RK函数
- ELF

*Hash Table*是一种高效的数据结构，一般用于判重或判等价。简要地说，Hash是信息的一种压缩，把一个字符串，一棵树等复杂的信息压缩成少量的数字，从而方便比较和存储。

常用的Hash函数

- RK函数
- ELF

RK函数

一个字符串可以被看成一个p进制数，我们将这个数字模M的值作为它的$Hash$值

$$Hash(S) = \sum_{i=1}^{len(S)} S[i] * p^{len(S)-i} \mod M$$

RK函数

一个字符串可以被看成一个p进制数，我们将这个数字模M的值作为它的$Hash$值

$$Hash(S) = \sum_{i=1}^{len(S)} S[i] * p^{len(S)-i} \mod M$$

Hint

- p要选比字符集大的数字
- p, M最好都选质数
- 如何处理$Hash$冲突
- 推荐用链表实现

Equal squares (*Ural* 1486)

题目大意

在一个N×M的字符矩阵中找到两个相同的子正方形矩阵（可以相交），并使找到的两个子正方形矩阵的边长尽量大

Equal squares (*Ural* 1486)

题目大意

在一个N×M的字符矩阵中找到两个相同的子正方形矩阵（可以相交），并使找到的两个子正方形矩阵的边长尽量大

算法分析

- 二分

Equal squares (*Ural* 1486)

题目大意

在一个N×M的字符矩阵中找到两个相同的子正方形矩阵（可以相交），并使找到的两个子正方形矩阵的边长尽量大

算法分析

- 二分
- 双重*Hash*

BZOJ1240 不稳定匹配

题目大意

有两个串A和B，每次可以进行的操作有：

- *INSERT* i j：在A中第i个字符前插入j
- *DELETE* i：删除A中第i个字符
- *REVERSE* i j：把A中i到j之间的内容反转
- *QUERY* i j：求A从第i个字符开始，B从第j个字符开始能匹配的最大长度，即询问A从第i个字符开始的后缀与B从第j个字符开始的后缀的LCP长度

BZOJ1240 不稳定匹配

题目大意

有两个串A和B，每次可以进行的操作有：

- *INSERT* i j：在A中第i个字符前插入j
- *DELETE* i：删除A中第i个字符
- *REVERSE* i j：把A中i到j之间的内容反转
- *QUERY* i j：求A从第i个字符开始，B从第j个字符开始能匹配的最大长度，即询问A从第i个字符开始的后缀与B从第j个字符开始的后缀的LCP长度

算法分析—带标记的Splay—作业题

- 翻转标记的处理，记录正的和反的Hash值

BZOJ1240 不稳定匹配

题目大意

有两个串A和B，每次可以进行的操作有：

- *INSERT* i j：在A中第i个字符前插入j
- *DELETE* i：删除A中第i个字符
- *REVERSE* i j：把A中i到j之间的内容反转
- *QUERY* i j：求A从第i个字符开始，B从第j个字符开始能匹配的最大长度，即询问A从第i个字符开始的后缀与B从第j个字符开始的后缀的LCP长度

算法分析—带标记的Splay—作业题

- 翻转标记的处理，记录正的和反的*Hash*值
- 询问的处理，二分答案+check

比较两棵树是否相同

题目描述

比较两棵无根树是否相同，儿子是无序的

有根树的做法

solution

- 树是递归定义的数据结构

有根树的做法

solution

- 树是递归定义的数据结构
- 假设我们已经得到了结点u的所有儿子(d个儿子)的$Hash$值$H(v_1), H(v_2), \ldots, H(v_d)$

有根树的做法

solution

- 树是递归定义的数据结构
- 假设我们已经得到了结点u的所有儿子(d个儿子)的$Hash$值$H(v_1), H(v_2), \ldots, H(v_d)$
- 由于儿子无序，我们将每个儿子的$Hash$值排序，记为$Hash(1), Hash(2), \ldots, Hash(d)$

有根树的做法

solution

- 树是递归定义的数据结构
- 假设我们已经得到了结点u的所有儿子(d个儿子)的$Hash$值$H(v_1), H(v_2), \ldots, H(v_d)$
- 由于儿子无序，我们将每个儿子的$Hash$值排序，记为$Hash(1), Hash(2), \ldots, Hash(d)$
- 直接当成字符串用RK函数$Hash$?

有根树的做法

solution

- 树是递归定义的数据结构
- 假设我们已经得到了结点u的所有儿子(d个儿子)的$Hash$值$H(v_1), H(v_2), \ldots, H(v_d)$
- 由于儿子无序，我们将每个儿子的$Hash$值排序，记为$Hash(1), Hash(2), \ldots, Hash(d)$
- 直接当成字符串用RK函数$Hash$?
- 反例

有根树的做法

solution

- 树是递归定义的数据结构
- 假设我们已经得到了结点u的所有儿子(d个儿子)的$Hash$值$H(v_1), H(v_2), \ldots, H(v_d)$
- 由于儿子无序，我们将每个儿子的$Hash$值排序，记为$Hash(1), Hash(2), \ldots, Hash(d)$
- 直接当成字符串用RK函数$Hash$?
- 反例
- 在字符串后加一个$size(u)$再$Hash$，即$Hash$字符串$Hash(1), Hash(2), \ldots, Hash(d), size(u)$

无根树的做法

solution

- 树的*dfs*

无根树的做法

solution

- 树的*dfs*
- 在*dfs*(*u*)的过程中同时维护*u*的每个儿子的*Hash*值，以及非*u*子树部分的*Hash*值
- 复杂度*O*(*nlogn*)
- *spoj TREEISO*作业题，要求本周完成

*sdoi*2011 *calc*

题目描述

- 给定y, z, P，计算满足$y^x \equiv z(mod\ P)$的最小整数x
- $1 \leq y, z, P \leq 10^9$，P为质数

*sdoi*2011 *calc*

题目描述

- 给定y, z, P，计算满足$y^x \equiv z(mod\ P)$的最小整数x
- $1 \leq y, z, P \leq 10^9$，P为质数

算法分析—离散对数

- 欧拉定理$y^{\phi(P)} \equiv 1(mod\ P)$

*sdoi*2011 *calc*

题目描述

- 给定y, z, P，计算满足$y^x \equiv z(mod\ P)$的最小整数x
- $1 \leq y, z, P \leq 10^9$，P为质数

算法分析—离散对数

- 欧拉定理$y^{\phi(P)} \equiv 1(mod\ P)$
- $y^{x+\phi(P)} \equiv z(mod\ P)$

*sdoi*2011 *calc*

题目描述

- 给定y, z, P，计算满足$y^x \equiv z(mod\ P)$的最小整数x
- $1 \leq y, z, P \leq 10^9$，P为质数

算法分析—离散对数

- 欧拉定理$y^{\phi(P)} \equiv 1(mod\ P)$
- $y^{x+\phi(P)} \equiv z(mod\ P)$
- 若$[0, \phi(P))$内无解，由上式的周期性必然不会有解

*sdoi*2011 *calc*

题目描述

- 给定y, z, P，计算满足$y^x \equiv z(mod\ P)$的最小整数x
- $1 \leq y, z, P \leq 10^9$，P为质数

算法分析—离散对数

- 欧拉定理$y^{\phi(P)} \equiv 1(mod\ P)$
- $y^{x+\phi(P)} \equiv z(mod\ P)$
- 若$[0, \phi(P))$内无解，由上式的周期性必然不会有解
- 设$x = A\sqrt{P} + B$，$A, B \leq \sqrt{P}$，$y^x \equiv y^{A\sqrt{P}} * y^B$

*sdoi*2011 *calc*

题目描述

- 给定y, z, P，计算满足$y^x \equiv z(mod\ P)$的最小整数x
- $1 \leq y, z, P \leq 10^9$，P为质数

算法分析—离散对数

- 欧拉定理$y^{\phi(P)} \equiv 1(mod\ P)$
- $y^{x+\phi(P)} \equiv z(mod\ P)$
- 若$[0, \phi(P))$内无解，由上式的周期性必然不会有解
- 设$x = A\sqrt{P} + B$，$A, B \leq \sqrt{P}$，$y^x \equiv y^{A\sqrt{P}} * y^B$
- 我们把y^B映射到*Hash*表中，然后枚举A，用*Extend kmp*解出y^B的值，检查*Hash*表中是否有该数字即可

扩展kmp的定义

定义

函数$A_i = |S$与$S(i,n)$的最长公共前缀$|$，这里$2 \leq i \leq n$

$O(n)$的时间复杂度

复杂度分析,$k + a_k$单调递增

代码

$j \Leftarrow 0$
while $T[1+j] = T[2+j] do$ **do**
  $j \Leftarrow j+1$
end while
$A_2 \Leftarrow j, k \Leftarrow 2$
for $i = 3 \rightarrow m$ **do**
  $Len \Leftarrow k + A_k - 1, L \Leftarrow A_{i-k+1}$
  if $L < Len - i + 1$ **then**
    $A_i \Leftarrow L$
  else
    $j \Leftarrow max(0, Len - i + 1)$
    while $T[i+j] = T[1+j]$ **do**
      $j \Leftarrow j+1$
    end while
  end if
end for

经典问题

最长回文串

经典问题

最长回文串

- 二分+*Hash*
- 利用前缀*Hash*和后缀*Hash*可以$O(1)$得到某一段正或反的*Hash*值

经典问题

最长回文串

- 二分+*Hash*
- 利用前缀*Hash*和后缀*Hash*可以$O(1)$得到某一段正或反的*Hash*值
- 类似*Extend kmp*的方法

经典问题

最长回文串

- 二分+*Hash*
- 利用前缀*Hash*和后缀*Hash*可以$O(1)$得到某一段正或反的*Hash*值
- 类似*Extend kmp*的方法
- 小技巧，插空格法

经典问题

最长回文串

- 二分+*Hash*
- 利用前缀*Hash*和后缀*Hash*可以$O(1)$得到某一段正或反的*Hash*值
- 类似*Extend kmp*的方法
- 小技巧，插空格法

最长重复子串

经典问题

最长回文串

- 二分+*Hash*
- 利用前缀*Hash*和后缀*Hash*可以$O(1)$得到某一段正或反的*Hash*值
- 类似*Extend kmp*的方法
- 小技巧，插空格法

最长重复子串

- 分治+*Extend kmp*
- 复杂度$O(nlogn)$

简介

字母树（又叫单词查找树、*Trie*树，*Trie Tree*），构造简单，并能很好地处理和“串”相关的检索问题。

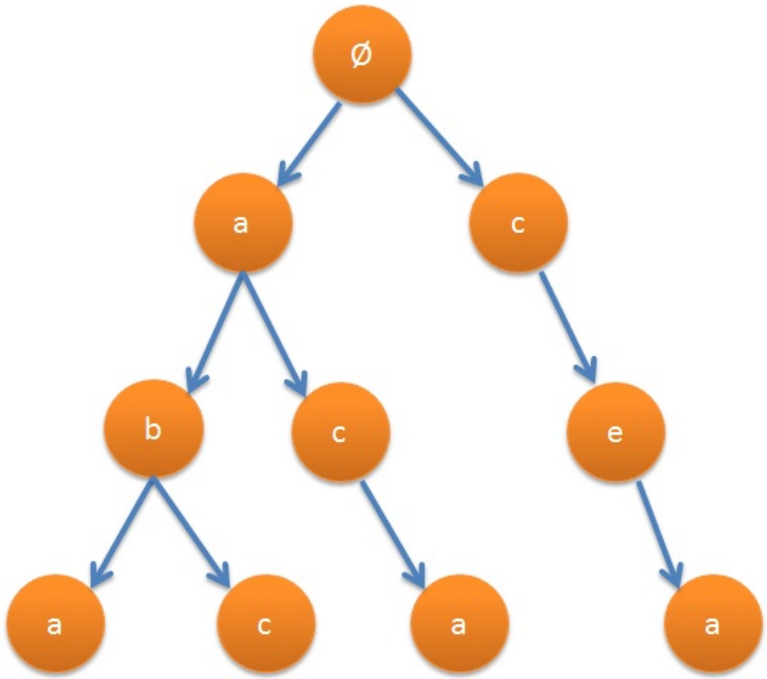

Figure: 字母树图示

其他

Hint

- 串的排序—字母树的*dfs*

其他

Hint

- 串的排序—字母树的*dfs*
- 字母树的存储

其他

Hint

- 串的排序—字母树的*dfs*
- 字母树的存储
- 压缩字母树，后缀树(不要求掌握)

其他

Hint

- 串的排序—字母树的*dfs*
- 字母树的存储
- 压缩字母树，后缀树(不要求掌握)
- $LCP \rightarrow LCA$

其他

Hint

- 串的排序—字母树的*dfs*
- 字母树的存储
- 压缩字母树，后缀树(不要求掌握)
- $LCP \rightarrow LCA$
- $Trie \rightarrow AC$自动机

介绍

Aho – Corasick automation，该算法在1975年产生于贝尔实验室，是著名的多模匹配算法之一。

要搞懂AC自动机，先得有Trie和KMP的基础知识。AC自动机算法分为3步：构造Trie树，构造失败指针和模式匹配过程。

介绍

Aho – Corasick automation，该算法在1975年产生于贝尔实验室，是著名的多模匹配算法之一。

要搞懂AC自动机，先得有Trie和KMP的基础知识。AC自动机算法分为3步：构造Trie树，构造失败指针和模式匹配过程。

in fact

- *AC*自动机是在*Trie*上构建的*kmp*
- *kmp*是一条链时的*AC*自动机

1.构造Trie树

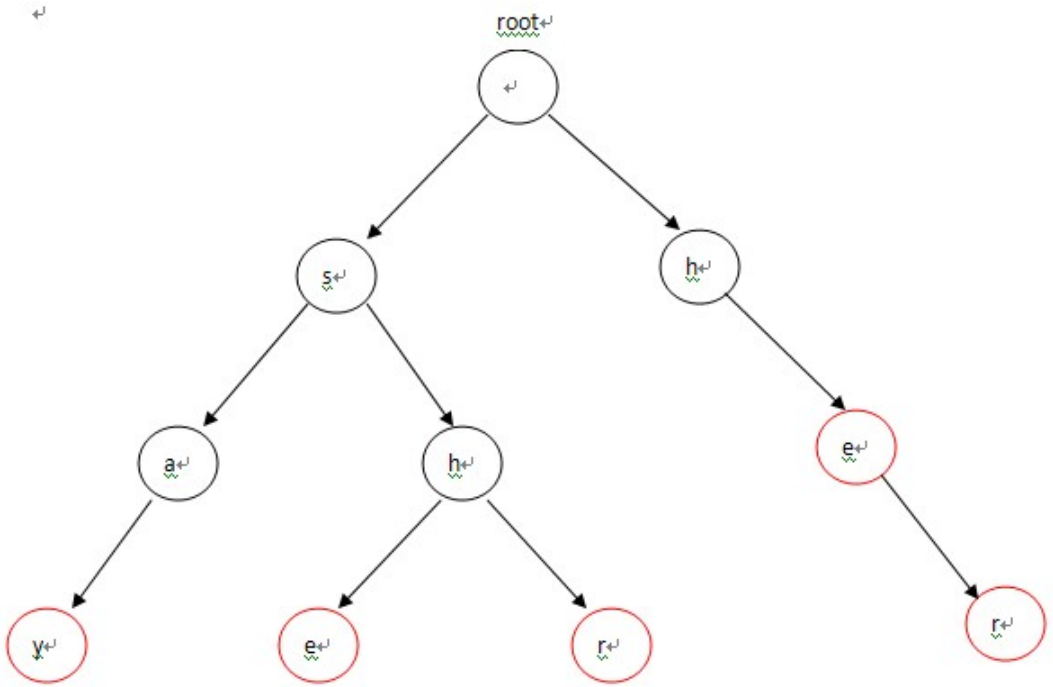

Figure: 详见上一部分内容

2.构造失败指针

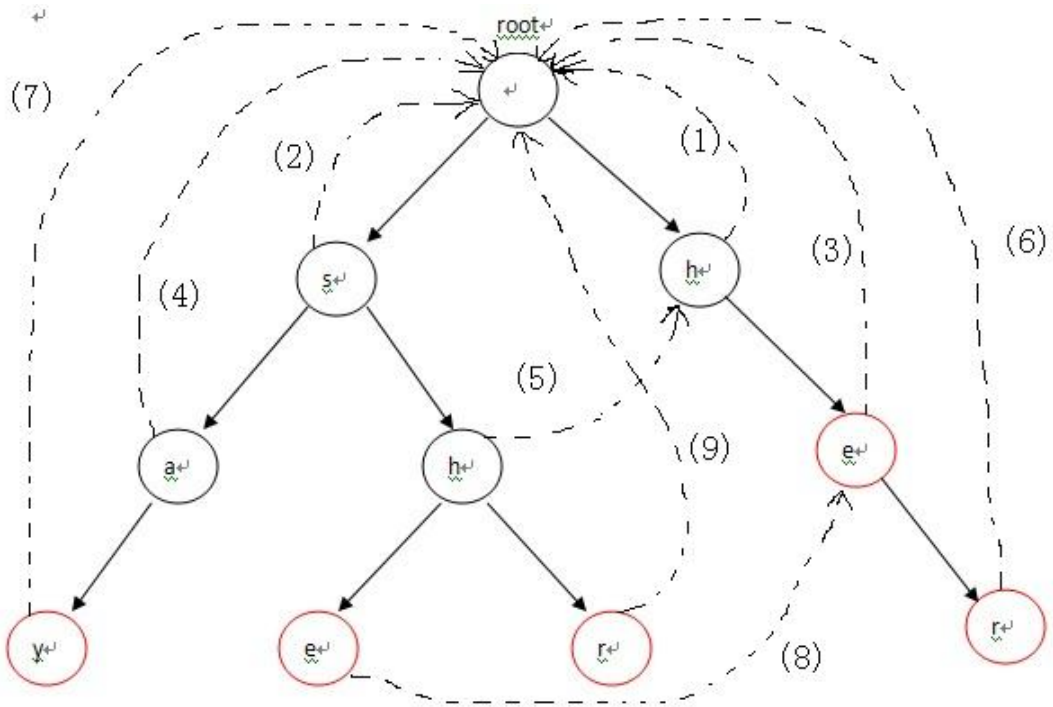

Figure: 详见代码

3.模式匹配

模式匹配的过程就是沿着*Trie*匹配，若无法匹配了就沿着失败指针向上跳的过程。

*Ural*1158 *Censored*!

题目描述

- 有一些和谐词汇和一个文本串
- 找到文本中第一个禁忌词汇出现的位置。

*Ural*1269 *Obscene Words Filter*

题目描述

- 有一些和谐词汇，你的字母集大小为M
- 现在你要说一句长为N的话
- 求有几种方案使得句子中不包含任何和谐词汇

spoj NSUBSTR2

题目描述

- 有小写字母串T和若干询问串S以及常数A, B
- 每次询问一个串S在T中出现了几次
- 每次询问串S在T中出现了几次，令每次回答完询问后的答案为ans，则动态地将第$((A * ans + B) mod\ 26 + 1)$个字母添加到T串末尾。

spoj NSUBSTR2

题目描述

- 有小写字母串T和若干询问串S以及常数A, B
- 每次询问一个串S在T中出现了几次
- 每次询问串S在T中出现了几次，令每次回答完询问后的答案为ans，则动态地将第$((A * ans + B) mod\ 26 + 1)$个字母添加到T串末尾。

Hint

- 失败指针组成的一棵树
- 没有动态修改如何解决

spoj NSUBSTR2

算法分析

- 将所有询问串S读入，建AC自动机
- 用T串匹配该AC自动机时，每次经过一个点都将其沿着失败指针到根的路径的权值集体加1，这样某个串末尾结点的权值就是该结点的答案

spoj NSUBSTR2

算法分析

- 将所有询问串S读入，建AC自动机
- 用T串匹配该AC自动机时，每次经过一个点都将其沿着失败指针到根的路径的权值集体加1，这样某个串末尾结点的权值就是该结点的答案
- 如何实现动态操作呢，注意到AC自动机的失败指针形成的是一棵树，所以可以用动态树来维护
- 也可以先勾出失败指针树的dfs序，利用线段树维护dfs序

spoj NSUBSTR2

算法分析

- 将所有询问串S读入，建AC自动机
- 用T串匹配该AC自动机时，每次经过一个点都将其沿着失败指针到根的路径的权值集体加1，这样某个串末尾结点的权值就是该结点的答案
- 如何实现动态操作呢，注意到AC自动机的失败指针形成的是一棵树，所以可以用动态树来维护
- 也可以先勾出失败指针树的dfs序，利用线段树维护dfs序

作业

[NOI2011]阿狸的打字机
BZOJ可以提交