6.2 应用hash技术处理字符串的实验范例

吴永辉

Email： yhwu@fudan.edu.cn

WeChat： 13817360465

ICPC Asia Programming Contest 1st Training Committee – Chair

- 字符串的hash是通过某种字符串hash函数将不同的字符串映射到不同的数字，配合其他数据结构或STL，进行判重，统计，查询等操作。
- 一个常用的字符串hash函数是$hash[i]=(hash[i-1]*p+idx(s[i]))\%mod$，即$hash[i]$是字符串的前$i$个字符组成的前缀的hash值，而$idx(s)$为字符$s$的一个自定义索引，例如，$idx(‘a’)=1$，$idx(‘b’)=2$，……，$idx(‘z’)=26$。
- 例如，$p=7$，$mod=91$，把字符串"abc"映射为一个整数：hash[0]=$idx(‘a’)\%91=1$，字符串"a"被映射为1；$hash[1]=(hash[0]*p+idx(‘b’))\%mod=9$，表示字符串"ab"被映射为9；$hash[2]=(hash[1]*p+idx(‘c’))\%mod=66$，所以，字符串"abc"被映射成66。

- 基于字符串hash函数，可以求字符串的任何一个子串的hash值：$hash[l..r]=((hash[r]-hash[l-1]*p^{r-l+1})\ \%mod+mod)\ \%mod$。
- 如上例，对于字符串"abab"，$hash[2]=(hash[1]*p+idx(\text{‘a’}))\%mod=64$，表示字符串"aba"被映射为64；$hash[3]=(hash[2]*p+idx(\text{‘b’}))\%mod=86$，即字符串"abab"被映射为86。则$hash[2..3]=(\ (hash[3]-hash[1]*p^2)\%mod+mod)\%mod\ =9=hash[1]$，即字符串"abab"的第一个"ab"子串和第二个"ab"子串所对应的hash值相同，都是9。

- *p*和*mod*取值要合适，否则可能会出现不同字符串有相同的hash值。一般*p*和*mod*要取素数，*p*取一个6位到8位的较大的素数，*mod*取一个大素数，比如10^9+7，或者10^9+9。

6.2.1 Power Strings

- 试题来源：**Waterloo local 2002.07.01**
- 在线测试：**POJ 2406**

- 给出两个字符串*a*和*b*，定义*a**b*是它们的串联。例如，如果*a*="abc"，*b*="def"，则*a***b*="abcdef"。如果把串联视为相乘，非负整数指数则定义为：*a*^0=""（空串），而*a*^(*n*+1)=*a**(*a*^*n*)。

- 输入
- 每个测试用例是在一行中给出一个可打印字符的字符串*s*。*s*的长度将至少为1，并且不会超过1000000（一百万）个字符。在最后一个测试用例后面，给出包含句点的一行。
- 输出
- 对于每个*s*，请输出大的*n*，使得对某个字符串*a*，*s=a^n*。
- 提示
- 本题海量输入，为避免超时，请使用scanf替代cin。

试题解析

设字符串 s 的长度为 $L=strlen(s+1)$，字符串 s 的下标从 1 开始。

首先计算出字符串 s 中每个前缀的 hash 函数值，即 $hash[i]=(hash[i-1]*k+s[i])\%\ mod$（$1\le i\le L$）；然后按照长度递增的顺序枚举 s 中可能存在的相邻子串。若 $L\%x==0$，则说明 s 中可能存在长度为 x 且满足相乘关系的相邻子串，即对于等长子串 $s_{1..x}, s_{x+1..2x}, \ldots, s_{(n-1)*x+1..L}$，如果 $hash[x]=hash[x+1..2x]=\ldots\ldots=hash[(n-1)*x+1..L]$，其中，子串 $s_{i-x+1..i}$ 的 hash 值为 $((hash[i]-(hash[i-x]*k^x)\%mod+mod)\%mod$，$n=\frac{L}{x}$；则相乘关系成立，即 s 为连续 n 个子串 a，$s=a^n$。

由于此时子串长度 x 是最小的，因此次幂 $n=\frac{L}{x}$ 为最大，n 即为问题的解。

6.2.2 Stammering Aliens

- 试题来源：**ACM 2009 South Western European Regional Contest**
- 在线测试：**HDOJ 4080，UVA 4513**

- Ellie Arroway博士与一个外星文明建立了联系。然而，所有破解外星人讯息的努力都失败了，因为他们遇上了一群口吃的外星人。Ellie的团队发现，在每一条足够长的讯息中，最重要的单词都会以连续字符的顺序出现一定次数的重复，甚至出现在其他单词的中间；而且，有时讯息会以一种模糊的方式缩写；例如，如果外星人要说bab两次，他们可能会发送讯息babab，该讯息已被缩写，在第一个单词中第二个b被重用为第二个单词中的第一个b。
- 因此，一条讯息可能包含重复的相同单词一遍又一遍。现在，Ellie向您，S.R. Hadden，寻求帮助，以确定一条讯息的要点。

- 给出一个整数*m*和一个表示讯息的字符串*s*，请您查找至少出现*m*次的*s*的最长子字符串。例如，在讯息baaaababababbababbab中，长度为5个单词的babab包含3次，即在位置5、7和12处（其中下标索引从零开始），出现3次或更多次的子字符串不会比5更长（请参见样例输入中的第1个样例）；而且，在这条讯息中，没有子串出现11次或更多次（请参见第2个样例）。如果存在多个解决方案，则首选出现最右的子字符串（请参见第3个样例）。

- 输入
- 输入包含若干测试用例。每个测试用例在第一行给出一个整数*m*（*m*≥1），表示最小重复次数；接下来的一行给出一个长度介于*m*和40000之间（包括*m*和40000）的字符串*s*。在*s*中，所有字符都是从“a”到“z”的小写字符。最后一个测试用例由*m*=0标识，程序不用处理。

- 输出
- 对每个测试用例输出一行。如果无解，则输出none；否则，在一行中输出两个用空格分隔的整数，第一个整数表示至少出现*m*次的子串的最大长度；第二个整数表示此子字符串的最右起始位置。

试题解析

- 设字符串前*i*个字符组成的前缀的hash值：
- x_i=(x_{i-1}*26+*s*[*i*]对应的序号值))% *mod*1，
- y_i=(y_{i-1}*26+*s*[*i*]对应的序号值))% *mod*2。
- 注‘a’的序号值为0，…，’z’的序号值为25
- 所有的x_i存储在长度为*mod*1的哈希表*hash*1[]中；所有的y_i存储在长度为*mod*2的哈希表*hash*2[]中。

- 我们使用二分法计算至少出现*m*次的子串的最大长度和其起始位置，搜索区间为子串长度，初始时为整个子串的长度。然后不断按照下述方法二分：
- 计算中间指针*mid*和长度为*mid*的前缀的hash值。若在hash表中这个hash值的个数不小于*m*，则说明目标子串的长度不小于*mid*，将目标子串的最大长度*ans*暂调整为*mid*，记下子串的起始位置*pos*，继续搜索右区间；否则说明目标子串长度小于*mid*，搜索左区间。
- 这个搜索过程一直进行到区间不存在为止。

现在，问题的关键变成，怎么判断字串 s 中是否存在长度为 len 且出现次数不小于 m 的子串？如果有，怎么计算其起始位置？

首先计算 x_{len} 和 y_{len}。然后搜索哈希表 $hash1[]$ 和 $hash2[]$，看这两个哈希表是否分别存在值为 x_{len} 和 y_{len} 的哈希元素。如果存在，则设对应子串的起始位置为 0，出现次数+1；如果不存在，则出现次数设 1，x_{len} 和 y_{len} 分别置入 $hash1$ 和 $hash2$ 表。

然后搜索后缀 $s_{len..n}$ 中的每个字符，计算子串 $s_{i-len+1..i}$ 的 hash 值 $x_{i-len+1..i}$ 和 $y_{i-len+1..i}$（len ≤i≤），看哈希表 hash1[]和 hash2[]中是否存在 $x_{i-len+1..i}$ 和 $y_{i-len+1..i}$ 的元素值：如果存在，则设对应子串的起始位置为 $i - len + 1$，出现次数+1；如果不存在，则出现次数设 1，x_{len} 和 y_{len} 分别置入 hash1 和 hash2 表。

最后搜索 hash1[]表，在所有哈希值对应子串的出现次数不小于 m 的元素中，找出其中起始位置最大的一个元素。如果所有哈希值对应子串的出现次数都小于 m，则说明字串 s 中不存在长度为 len 且出现次数不小于 m 的子串，搜索失败。

6.2.3 String

- 试题来源：**2013 Asia Regional Changchun**
- 在线测试：**HDOJ 4821，UVA 6711**

- 给定一个字符串*S*和两个整数*L*和*M*，我们称*S*的一个子串是“可恢复的”，当且仅当
- （i）子串的长度为*M***L*;
- （ii）这一子串通过串联*S*的*M*个“多样化”子串来构造：其中每个子串的长度*L*；而且这些子串不能有两个完全一样的串。
- 如果*S*的两个子串是从*S*的不同部分切下来的，则它们被认为是“不同的”。例如，字符串"aa"有3个不同的子串"aa"，"a"和"a"。
- 请您计算*S*的不同的“可恢复”子字符串的数量。

- 输入
- 输入包含多个测试用例，以EOF结束。
- 每个测试用例的第一行给出两个用空格分隔的整数*M*和*L*。
- 每个测试用例的第二行给出一个字符串*S*，它只包含小写字母。
- *S*的长度不大于10^5，而且1≤*M**L≤*S*的长度。

- 输出
- 对每个测试用例，在一行中输出答案。

试题解析

- 通过字符串的hash，求出任意一个长度为*L*的子串的hash值。枚举字符串起始位置，从0枚举到*L*-1。然后，在这个位置开始，每*L*个字符作为一块，首先将前*M*块插入到*map*中，同时记录不相同字符串的个数，如果不相同字符串的个数是*M*，则满足要求。然后，将这个区间向右移，删掉第1块，加入第*M*+1块，同样记录不相同字符串的个数。