Hash

calabash_boy

2022 年 4 月 28 日

AC.NOWCODER.COM

Hash 定义

定义

Hash 是一种单射函数，可以将万物**单向**映射成一个整数值。

Hash 定义

定义

Hash 是一种单射函数，可以将万物**单向**映射成一个整数值。
字符串 Hash 就是指将一个字符串映射成一个整数值的方法，通常用来快速比较两个字符串是否相等。
约定：$H(S)$ 表示一个字符串 S 通过 Hash 算法 (H) 映射成的整数值。

牛客竞赛
AC.NOWCODER.COM

Hash 定义

定义

Hash 是一种单射函数，可以将万物**单向**映射成一个整数值。
字符串 Hash 就是指将一个字符串映射成一个整数值的方法，通常用来快速比较两个字符串是否相等。
约定：$H(S)$ 表示一个字符串 S 通过 Hash 算法 (H) 映射成的整数值。
例如：$S = abcde$，$H(S) = 12345$，$H'(S) = 54321$.

牛客竞赛 AC.NOWCODER.COM

Hash 定义

定义

Hash 是一种单射函数，可以将万物**单向**映射成一个整数值。
字符串 Hash 就是指将一个字符串映射成一个整数值的方法，通常用来快速比较两个字符串是否相等。
约定：$H(S)$ 表示一个字符串 S 通过 Hash 算法 (H) 映射成的整数值。
例如：$S = abcde$，$H(S) = 12345$，$H'(S) = 54321$.

性质

- 必要性：若字符串 $S = T$，则一定有 $H(S) = H(T)$。
- 非充分性：若 $H(S) = H(T)$，不一定有 $S = T$

牛客竞赛 AC.NOWCODER.COM

Hash 检测

定义

Hash 检测：通过 $H(S)$ 和 $H(T)$ 是否相等，来判断 S 和 T 是否相等的方法。
Hash 冲突：$H(S) = H(T)$，但 $S \neq T$，即为发生了 Hash 冲突。
Hash 检测时发生 Hash 冲突的概率是衡量 Hash 算法好坏的重要指标

牛客竞赛
AC.NOWCODER.COM

设计 Hash 算法

XJB Hash

充分发挥主观能动性，自行发明创造一些 Hash 算法。
例如：定义 $H(S) = |S| * (S[1] + S[|S|])$

设计 Hash 算法

XJB Hash

充分发挥主观能动性，自行发明创造一些 Hash 算法。
例如：定义 $H(S) = |S| * (S[1] + S[|S|])$

优缺点

优点：获得一些成就感。在某些特殊场景，可能会适用。
缺点：值域的利用率低，且冲突概率很高，beach == bitch？

牛客竞赛 AC.NOWCODER.COM

设计 Hash 算法

多项式 Hash

将字符串看作是某个进制（Base）下的数字串，

$$H(S) = \sum_{i=1}^{i<=|S|=n} S[i] * Base^{1+n-i} = \mathbf{H(S[1,\ |S|\ -\ 1])\ *\ Base\ +\ S[|S|]}$$

$$= S[1] * Base^{n} + S[2] * Base^{n-1} + \cdots + S[n] * Base^{0}$$

牛客竞赛
AC.NOWCODER.COM

设计 Hash 算法

多项式 Hash

将字符串看作是某个进制（Base）下的数字串，

$$H(S) = \sum_{i=1}^{i<=|S|=n} S[i] * Base^{1+n-i} = \textbf{H(S[1, |S| - 1]) * Base + S[|S|]}$$
$$= S[1] * Base^{n} + S[2] * Base^{n-1} + \cdots + S[n] * Base^{0}$$

例如: 字符集 $\Sigma = \{a, b, \cdots, o\}$，字符串就可以看成 16 进制数字串。
其中‘a’对应 1,'b' 对应 2,...,'j' 对应 A(10),...,'o' 对应 F(15)。

牛客竞赛 AC.NOWCODER.COM

设计 Hash 算法

多项式 Hash

将字符串看作是某个进制（Base）下的数字串，

$$H(S) = \sum_{i=1}^{i<=|S|=n} S[i] * Base^{1+n-i} = \mathbf{H(S[1,\ |S|\ -\ 1])\ *\ Base\ +\ S[|S|]}$$
$$= S[1] * Base^{n} + S[2] * Base^{n-1} + \cdots + S[n] * Base^{0}$$

例如: 字符集 $\Sigma = \{a, b, \cdots, o\}$，字符串就可以看成 16 进制数字串。
其中 ‘a’ 对应 1,’b’ 对应 2,...,’j’ 对应 A(10),...,’o’ 对应 F(15)。
若 $S = adenoo$，则 $H(S) = 145EFF_{(16)} = 1335039_{(10)}$

牛客竞赛
AC.NOWCODER.COM

设计 Hash 算法

多项式 Hash

将字符串看作是某个进制（Base）下的数字串，

$$H(S) = \sum_{i=1}^{i<=|S|=n} S[i] * Base^{1+n-i} = \mathbf{H(S[1,\ |S|\ -\ 1])\ *\ Base\ +\ S[|S|]}$$
$$= S[1] * Base^{n} + S[2] * Base^{n-1} + \cdots + S[n] * Base^{0}$$

例如: 字符集 $\Sigma = \{a, b, \cdots, o\}$，字符串就可以看成 16 进制数字串。
其中 ‘a’ 对应 1,’b’ 对应 2,...,’j’ 对应 A(10),...,’o’ 对应 F(15)。
若 $S = adenoo$，则 $H(S) = 145EFF_{(16)} = 1335039_{(10)}$

优缺点

优点：字符串和 Hash 值一一对应，不会发生 Hash 冲突。
缺点：数字范围过大，难以用原始数据结构存储和比较。

设计 Hash 算法

多项式取模 Hash(模哈)

模哈是为了解决多项式 Hash 的缺点，在效率和冲突率之间进行的折衷：将多项式 Hash 的值对一个较大的质数取模。

$$
\begin{aligned}
H'(S) = H(S)\ \%\mathbf{Mod} &= \Big(\sum_{i=1}^{i<=|S|=n} S[i] * Base^{n-i}\Big)\ \%\mathbf{Mod} \\
&= \big(S[1] * Base^{n-1} + S[2] * Base^{n-2} + \cdots + S[n] * Base^{0}\big)\ \%\mathbf{Mod}
\end{aligned}
$$

设计 Hash 算法

多项式取模 Hash(模哈)

模哈是为了解决多项式 Hash 的缺点，在效率和冲突率之间进行的折衷：将多项式 Hash 的值对一个较大的质数取模。

$$H'(S) = H(S)\ \mathbf{\%Mod} = \mathbf{(}\sum_{i=1}^{i<=|S|=n} S[i] * Base^{n-i}\mathbf{)}\ \mathbf{\%Mod}$$

$$= \mathbf{(}S[1] * Base^{n-1} + S[2] * Base^{n-2} + \cdots + S[n] * Base^{0}\mathbf{)}\ \mathbf{\%Mod}$$

优缺点

优点： 使得多项式 Hash 可以用原始数据结构 (uint/ulong) 存储和比较。
缺点： 会有小概率发生 Hash 冲突。

设计 Hash 算法

模哈冲突概率

模哈冲突是指：$H(S) \neq H(T)$ 但 $H(S)\%\mathbf{Mod} = H(T)\%\ \mathbf{Mod}$
模运算可以看作是一个均匀随机散列，即每个 $H(S)$ 会被随机的映射成 $[0, Mod-1]$ 内的整数。

设计 Hash 算法

模哈冲突概率

模哈冲突是指：$H(S) \neq H(T)$ 但 $H(S)\%\textbf{Mod} = H(T)\%\ \textbf{Mod}$
模运算可以看作是一个均匀随机散列，即每个 $H(S)$ 会被随机的映射成 $[0, Mod-1]$ 内的整数。

生日悖论

有 n 个人，每个人的生日可以认为是 $[1, 365]$ 范围内的随机整数。
若 $n > 365$，则一定有两个人生日相同。
若 $n \leq 365$，则没有人生日相同的概率为：$\frac{A(365,n)}{365^n}$。
当 $n = 23$ 时，上述结果约为 0.46。即有超过 50% 的概率有人生日相同。

牛客竞赛 AC.NOWCODER.COM

设计 Hash 算法

模哈冲突概率

模哈冲突是指：$H(S) \neq H(T)$ 但 $H(S)\%\mathrm{Mod} = H(T)\%\ \mathrm{Mod}$
模运算可以看作是一个均匀随机散列，即每个 $H(S)$ 会被随机的映射成 $[0, Mod-1]$ 内的整数。

生日悖论

有 n 个人，每个人的生日可以认为是 $[1, 365]$ 范围内的随机整数。
若 $n > 365$，则一定有两个人生日相同。
若 $n \leq 365$，则没有人生日相同的概率为：$\frac{A(365,n)}{365^n}$。
当 $n = 23$ 时，上述结果约为 0.46。即有超过 50% 的概率有人生日相同。
即当随即检验次数超过 $\sqrt{Mod}$ 时，就会有较大概率发生错误。
因此，在模哈中使用的 Mod 最好超过 Hash 检验次数的平方。

Hash 的三种姿势

Hash 模数

优秀的 Hash 模数首先应满足：足够大

Hash 的三种姿势

Hash 模数

优秀的 Hash 模数首先应满足：足够大

自然溢出：使用 *ULL* 保存 Hash 值，利用硬件特性，使 Hash 值自然溢出，即实现模数为 2^{64}，但容易构造 Hash 冲突。（详见 BZOJ3097）

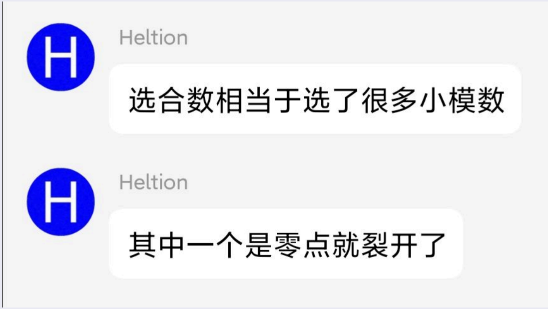
Hash 的三种姿势

Hash 模数

优秀的 Hash 模数首先应满足：**足够大**

自然溢出：使用 *ULL* 保存 Hash 值，利用硬件特性，使 Hash 值自然溢出，即实现模数为 2^{64}，但容易构造 Hash 冲突。（详见 BZOJ3097）

优秀的 Hash 模数还应是一个**质数**



Hash 的三种姿势

Hash 模数

质数模哈（单模）：选取 10^9 到 10^{10} 范围的大质数作为 Hash 模数。但也有广为人知的方法构造冲突。

Hash 的三种姿势

Hash 模数

质数模哈（单模）：选取 10^9 到 10^{10} 范围的大质数作为 Hash 模数。但也有广为人知的方法构造冲突。

双模（多模）：进行多次不同质数的单模哈希，有效降低冲突概率。在不泄露模数的前提下，没有已知方法可以构造冲突。

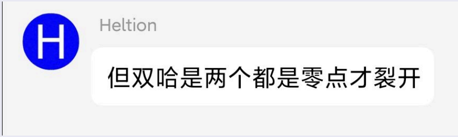
牛客竞赛 AC.NOWCODER.COM

Hash 的三种姿势

Hash 模数

质数模哈（单模）：选取 10^9 到 10^{10} 范围的大质数作为 Hash 模数。但也有广为人知的方法构造冲突。

双模（多模）：进行多次不同质数的单模哈希，有效降低冲突概率。在不泄露模数的前提下，没有已知方法可以构造冲突。



一些比较好的 Hash 素数

牛客竞赛
AC.NOWCODER.COM

例题 1

子串比较

给出一个字符串 S，$|S| \leq 100,000$。
共由 $Q \leq 100,000$ 次询问：$S[l_1, r_1]$ 与 $S[l_2, r_2]$ 是否相等。

牛客竞赛 AC.NOWCODER.COM

例题 1

子串比较

给出一个字符串 S，$|S| \leq 100,000$。
共由 $Q \leq 100,000$ 次询问：$S[l_1, r_1]$ 与 $S[l_2, r_2]$ 是否相等。

题解

需要设计一种数据结构，快速求出子串的 Hash 值。

子串 Hash

子串 Hash

$$H(S[l,r]) = (S[l] * Base^{r-l} + S[l+1] * Base^{(r-l-1)} + \cdots + S[r])\ \%\mathbf{Mod}$$

牛客竞赛
AC.NOWCODER.COM

子串 Hash

子串 Hash

$$H(S[l,r]) = \left(S[l] * Base^{r-l} + S[l+1] * Base^{(r-l-1)} + \cdots + S[r]\right) \ \%Mod$$

令 F(i) = H(Prefix[i])

$$F(l-1) = \left(S[1] * Base^{l-2} + S[2] * Base^{l-3} + \cdots + S[l-1]\right) \ \%Mod$$

$$F(r) = \left(S[1] * Base^{r-1} + S[2] * Base^{r-2} + \cdots + S[r]\right) \ \%Mod$$

子串 Hash

子串 Hash

$$H(S[l,r]) = (S[l]*Base^{r-l} + S[l+1]*Base^{(r-l-1)} + \cdots + S[r])\ \%\mathbf{Mod}$$

令 F(i) = H(Prefix[i])

$$F(l-1) = (S[1]*Base^{l-2} + S[2]*Base^{l-3} + \cdots + S[l-1])\ \%\mathbf{Mod}$$
$$F(r) = (S[1]*Base^{r-1} + S[2]*Base^{r-2} + \cdots + S[r])\ \%\mathbf{Mod}$$

因此 $H(S[l,r]) = F(r) - F(l-1)*Base^{r-l+1}$
所以只需要求出每个前缀的 Hash 值 $F(i)$，就可以快速求出子串 Hash 值。

例题 2

回文串

给出一个字符串 S，每次操作可以删除最末尾的一个字符，最少进行多少次操作可以得到一个回文串。

例题 2

回文串

给出一个字符串 S，每次操作可以删除最末尾的一个字符，最少进行多少次操作可以得到一个回文串。

题解

题目等价于求最长的回文前缀。对于回文串，正串和反串是相同的，可以利用 Hash 判定。枚举答案长度进行检测即可。

例题 2

回文串

给出一个字符串 S，每次操作可以删除最末尾的一个字符，最少进行多少次操作可以得到一个回文串。

题解

题目等价于求最长的回文前缀。对于回文串，正串和反串是相同的，可以利用 Hash 判定。枚举答案长度进行检测即可。

不能二分答案。

例题 3

子串字典序比较 1

给出一个正整数数组 A，长度不超过 $100,000$，以及 $Q \leq 100,000$ 次询问：

子串 $A[l_1, r_1]$ 和 $A[l_2, r_2]$ 的字典序大小关系。

例题 3

子串字典序比较 1

给出一个正整数数组 A，长度不超过 $100,000$，以及 $Q \leq 100,000$ 次询问：
子串 $A[l_1, r_1]$ 和 $A[l_2, r_2]$ 的字典序大小关系。

题解

判定两个串的字典序等价于求解两个串的**最长公共前缀 (LCP)**，所以本题要求快速求出两个子串的 LCP。

例题 3

子串字典序比较 1

给出一个正整数数组 A，长度不超过 $100,000$，以及 $Q \leq 100,000$ 次询问：
子串 $A[l_1, r_1]$ 和 $A[l_2, r_2]$ 的字典序大小关系。

题解

判定两个串的字典序等价于求解两个串的**最长公共前缀 (LCP)**，所以本题要求快速求出两个子串的 LCP。
方法 1：后缀数组 SA，复杂度 $O(n \log n)$ 日后再讲。

例题 3

子串字典序比较 1

给出一个正整数数组 A，长度不超过 $100,000$，以及 $Q \leq 100,000$ 次询问：
子串 $A[l_1, r_1]$ 和 $A[l_2, r_2]$ 的字典序大小关系。

题解

判定两个串的字典序等价于求解两个串的**最长公共前缀 (LCP)**，所以本题要求快速求出两个子串的 LCP。
方法 1：后缀数组 SA，复杂度 $O(n \log n)$ 日后再讲。

方法 2：LCP 满足二分性，问题转化为判定两个子串是否相等，可以用 Hash 解决。复杂度 $O(n \log n)$。

例题 4

子串字典序比较 2

给出一个正整数数组 A，长度不超过 $100,000$，以及 $Q \leq 100,000$ 次操作：

询问：子串 $A[l_1, r_1]$ 和 $A[l_2, r_2]$ 的字典序大小关系。

修改：将 $A[x]$ 的值替换为 y。

例题 4

子串字典序比较 2

给出一个正整数数组 A，长度不超过 $100,000$，以及 $Q \leq 100,000$ 次操作：

询问：子串 $A[l_1, r_1]$ 和 $A[l_2, r_2]$ 的字典序大小关系。

修改：将 $A[x]$ 的值替换为 y。

题解

询问依然使用二分 $+$Hash。

由于两串拼接，其 Hash 值可以快速和并。因此可以使用线段树维护 Hash 值。

复杂度 $O(n \log^2 n)$

AC.NOWCODER.COM

例题 5

子串字典序比较 3

给出一个正整数数组 A，长度不超过 $100,000$，以及 $Q \leq 100,000$ 次操作：

询问：子串 $A[l_1, r_1]$ 和 $A[l_2, r_2]$ 的字典序大小关系。

修改：将区间 $[l, r]$ 位置的数字值增加 1。

牛客竞赛
AC.NOWCODER.COM

例题 5

子串字典序比较 3

给出一个正整数数组 A，长度不超过 $100,000$，以及 $Q \leq 100,000$ 次操作：

询问：子串 $A[l_1, r_1]$ 和 $A[l_2, r_2]$ 的字典序大小关系。

修改：将区间 $[l, r]$ 位置的数字值增加 1。

题解

询问依然使用二分 +Hash。

修改操作对线段树节点的影响为：加上 $Base^p + Base^{p+1} + \cdots + Base^q$。

提前求出 Base 等比数列的前缀和，使用区间修改线段树即可。

例题 6

CF580E

给出一个数组 A，进行 Q 次操作：
询问：p 是否是区间 $[L, R]$ 的周期。
修改：将区间 $[L, R]$ 的数字赋值为 k。

例题 6

CF580E

给出一个数组 A，进行 Q 次操作：
询问： p 是否是区间 $[L, R]$ 的周期。
修改： 将区间 $[L, R]$ 的数字赋值为 k。

题解

由上节课可以知道：询问等价于询问 $R - L + 1 - p$ 是否为 $[L, R]$ 的 Border，即是否有 $A[L, R - p] == A[L + p, R]$。可以利用 Hash 解决。修改操作可以利用线段树维护 Hash。