## HASH算法的简单了解

**有趣的比喻**

**=-----------------------------------------------------------------------------------------------------------------------------=**

我们有很多的小猪，每个的体重都不一样，假设体重分布比较平均(我们考虑到公斤级别)，我们按照体重来分,划分成100个小猪圈。

然后把每个小猪，按照体重赶进各自的猪圈里，记录档案。

好了，如果我们要找某个小猪怎么办呢？我们需要每个猪圈，每个小猪的比对吗？

当然不需要了。

我们先看看要找的这个小猪的体重，然后就找到了对应的猪圈了。

在这个猪圈里的小猪的数量就相对很少了。

我们在这个猪圈里就可以相对快的找到我们要找到的那个小猪了。

对应于hash算法。

就是按照hashcode分配不同的猪圈，将hashcode相同的猪放到一个猪圈里。

查找的时候，先找到hashcode对应的猪圈，然后在逐个比较里面的小猪。

所以问题的关键就是建造多少个猪圈比较合适。

如果每个小猪的体重全部不同（考虑到毫克级别)，每个都建一个猪圈，那么我们可以最快速度的找到这头猪。缺点就是，建造那么多猪圈的费用有点太高了。

如果我们按照10公斤级别进行划分，那么建造的猪圈只有几个吧，那么每个圈里的小猪就很多了。我们虽然可以很快的找到猪圈，但从这个猪圈里逐个确定

那头小猪也是很累的。

所以，好的hashcode，可以根据实际情况，根据具体的需求，在时间成本(更多的猪圈，更快的速度)和空间本(更少的猪圈，更低的空间需求)之间平衡。

**=-----------------------------------------------------------------------------------------------------------------------------=**

**什么是hash算法**

介绍

HASH算法是密码学的基础，比较常用的有MD5和SHA。

**特点**

最重要的两条性质，就是不可逆和无冲突。

所谓不可逆，就是当你知道x的HASH值，无法求出x；

所谓无冲突，就是当你知道x，无法求出一个y， 使x与y的HASH值相同。

**原理**

这两条性质在数学上都是不成立的。因为一个函数必然可逆，且由于HASH函数的值域有限，理论上会有无穷多个不同的原始值，它们的hash值都相同。

MD5和SHA做到的，是求逆和求冲突在计算上不可能，也就是正向计算很容易，而反向计算即使穷尽人类所有的计算资源都做不到。

**贡献**

密码学的几个算法（HASH、对称加密、公私钥）是计算机科学领域最伟大的发明之一，它授予了弱小的个人在**强权面前信息的安全**（而且是绝对的安全）。

举个例子，只要你一直使用https与国外站点通讯，并注意对方的公钥没有被篡改，G\*\*W可以断开你的连接，但它永远不可能知道你们的传输内容是什么。

简单举个例子：

若存储4个元素 13 7 14 11

显然，我们可以用数组来存。也就是：a[1] = 13; a[2] = 7; a[3] = 14; a[4] = 11;

当然，我们也可以用Hash来存。下面给出一个简单的Hash存储：

先来确定哪个函数。我们就用h(ki) = ki%5 。

注意：这里的除数取素数（又称质数，如2,3,5,7,11,13,17,19····。质数，又称为因数只有他本身的数。）

对于第一个元素 h(13) = 13%5 = 3; 也就是说13的下标为3；即Hash[3] = 13;

对于第二个元素 h(7) = 7 % 5 = 2; 也就是说7的下标为2; 即Hash[2] = 7;

同理，

Hash[4] = 14;

Hash[1] = 11;

最后的结果：

[1] => 11;

[2] => 7;

[3] => 13;

[4] => 14;

也就是说，通过hash存这一组数，就是以上这个顺序。

Hash关于冲突的解决

已知一组数：26，36，41，38，44，15，68，12，06，51，用“除余法散列构造函数”，用“线性探查法”解决冲突，构造这组数字的散列表。

首先，用除余数法，求得余数结果，顺序为0，10，2，12，5，2，3，12，6，12

a.仔细观察，前五个没有相同的数，所以，前五个数的散列表为

[0] => 26;

[10] => 36;

[2] => 41;

[12] => 38;

[5] => 44;

注：0，10，2，12，5

b.余数结果，到了第六个数2，出现了重复（前五个数里也有2），那这个怎么办呢？

探查 h1 = (2+1)%13 = 3;

因此，将第五个数，放入散列3，即 [3]=>15 .

c.接下来看第七个,68，它的余数是3，但是，我们刚刚把3这个序列给了15，因此，它不能用3这个序列

因此，只能将其插入到4这个序列，即 [4] = 68

d.再到第8个数12，余数是12，但是序号12已经被38占领，所以，这个它不能用

探查 h1 = （12+1）%13 = 0

但是，我们发现 0 这个序号也被使用了

再探查 h2 = (12+2)%13 = 1

1没被占用，所以，结果是 [1]=>12 .

e.再到第九个数6，余数是6，这个序号没有被占用，所以，结果 [6]=>6

f.最后一个数51，余数12，可是，12被占了

探查 h1 = (12+1)%13 = 0

但是，0被占了

再探查 h2 = (12+2)%13 = 1

可是，1也被占了

再探查 h3 = (12+3)%13 = 2

然而，2也被占了，仔细观察，我们发现一直到6都已经被占，只有7空着

所以，结果是 [7]=>51

g.最后排序

[0] => 26;

[1] => 12;

[2] => 41;

[3] => 15;

[4] => 68;

[5] => 44;

[6] => 6;

[7] => 51;

[8] =>

[9] =>

[10] => 36;

[11] =>

[12] => 38;

**通过实验，我们感受一下hash算法的不可逆和无冲突：**

不可逆和无冲突，这两条性质在数学上都是不成立的。因为一个函数必然可逆，且由于HASH函数的值域有限，理论上会有无穷多个不同的原始值，它们的hash值都相同。所以，不可逆。

MD5和SHA做到的，是求逆和求冲突在计算上不可能，也就是正向计算很容易，而反向计算即使穷尽人类所有的计算资源都做不到。