散列思想

散列表的英文叫“Hash Table”，我们平时也叫它“哈希表”或者“Hash 表”

散列表用的是数组支持按照下标随机访问数据的特性，所以散列表其实就是数组的一种扩展，由数组演化而来。可以说，如果没有数组，就没有散列表。

散列函数，就是一个函数。我们可以把它定义成hash（key），其中key表示元素的键值，hash（key）的值表示经过散列函数计算得到的散列值。

散列函数的伪代码：

```
int hash(String key) {
  // 获取后两位字符
  string lastTwoChars = key.substr(length-2, length);
  // 将后两位字符转换为整数
  int hashValue = convert lastTwoChas to int-type;
  return hashValue;
}
```

构造散列函数的基本要求：

1. 散列函数计算得到的散列值是一个非负整数；

2. 如果key1 = key2，那hash（key1）==hash（key2）；

3. 如果key1 ！= key2，那hash（key1） ！= hash（key2）。

但是几乎不可能找到完全避免冲突的散列函数

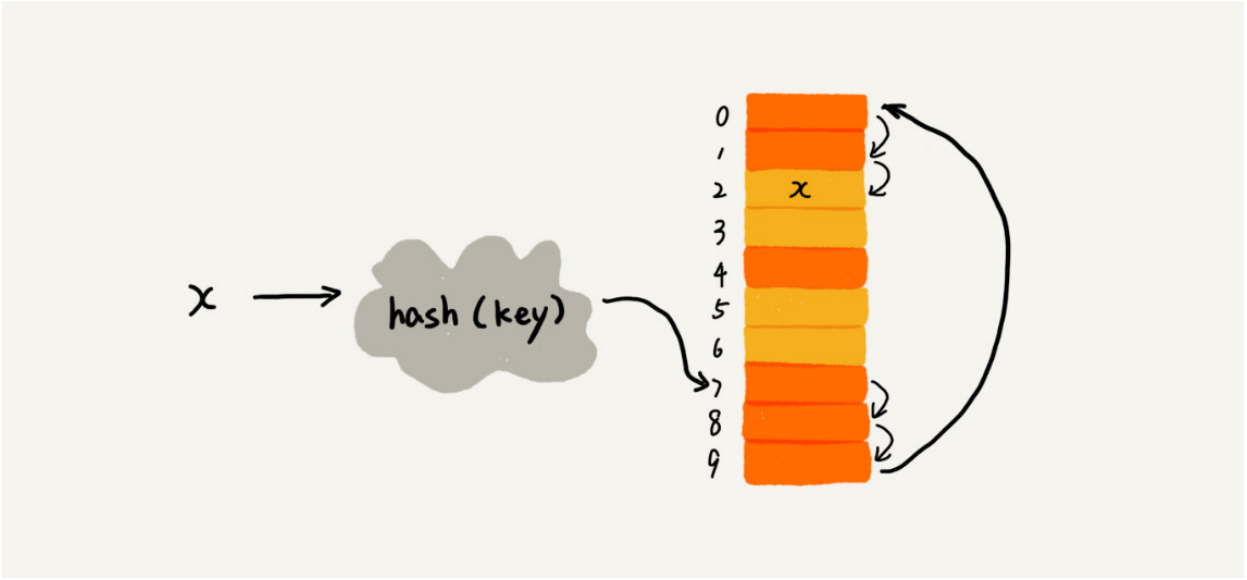
解决散列冲突的两种方法：

1.开放寻址法

线性探测：

当我们往散列表中插入数据时，如果某个数据经过散列函数散列之后，储存位置已经被

占用了，我们就从当前位置开始，依次往后查找，看是否有空闲位置，直到找到为止。



二次探测：

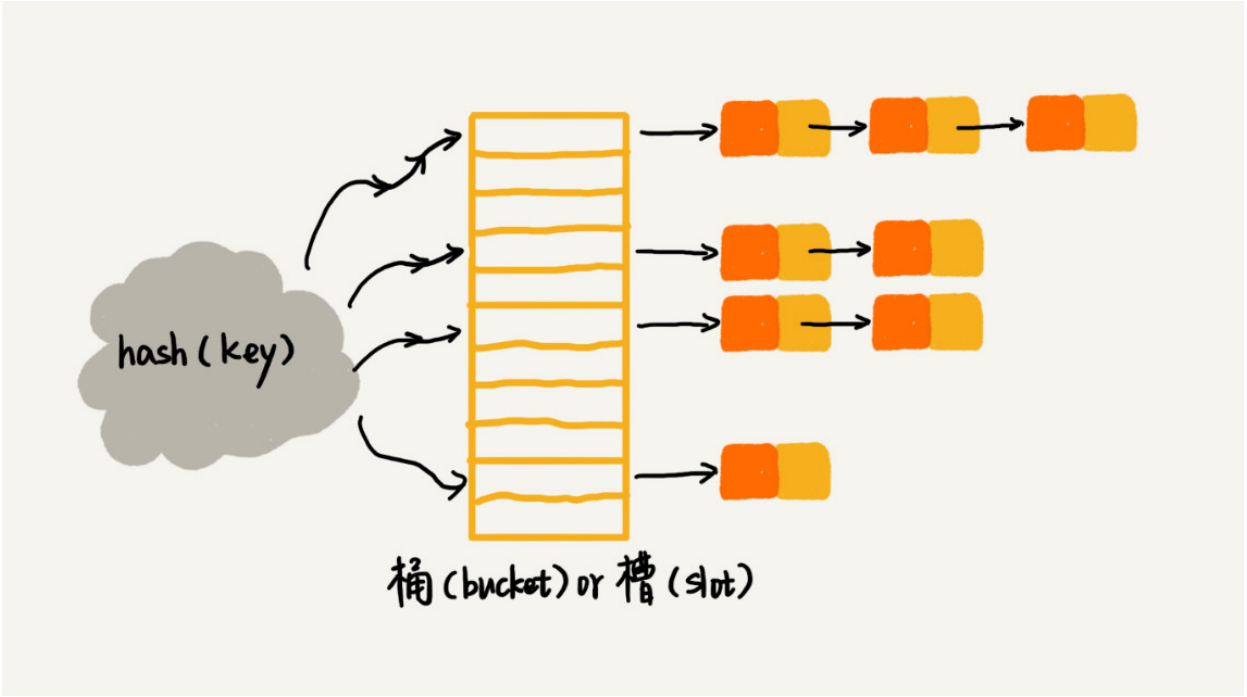
二次探测探测的步长就变成了原来的“二次方”，也就是说，它探测的下标序列就是hash(key)+0, hash(key)+1^2, hash(key)+22……

双重散列：

使用一组散列函数hash1(key)，hash2(key)，hash3(key).....，先用第一个散列函数，如果计算得到的储存位置已经被占用，再用第二个散列函数，依次类推，直到找到空间的储存位置。

2.链表法

链表法是一种更加常用的散列冲突解决方法，相比开放寻址法，他要简单的多。



总结:

一、散列表的由来?

1.散列表来源于数组，它借助散列函数对数组这种数据结构进行扩展，利用的是数组支持按照下标随机访问元素的特性。

2.需要存储在散列表中的数据我们称为键，将键转化为数组下标的方法称为散列函数，散列函数的计算结果称为散列值。

3.将数据存储在散列值对应的数组下标位置。

二、如何设计散列函数?

总结3点设计散列函数的基本要求

1.散列函数计算得到的散列值是一个非负整数。

2.若key1=key2，则hash(key1)=hash(key2)

3.若key≠key2，则hash(key1)≠hash(key2)

正是由于第3点要求，所以产生了几乎无法避免的散列冲突问题。

三、散列冲突的解放方法?

1.常用的散列冲突解决方法有2类：开放寻址法（open addressing）和链表法（chaining）

2.开放寻址法

①核心思想：如果出现散列冲突，就重新探测一个空闲位置，将其插入。

②线性探测法（Linear Probing）：

插入数据：当我们往散列表中插入数据时，如果某个数据经过散列函数之后，存储的位置已经被占用了，我们就从当前位置开始，依次往后查找，看是否有空闲位置，直到找到为止。

查找数据：我们通过散列函数求出要查找元素的键值对应的散列值，然后比较数组中下标为散列值的元素和要查找的元素是否相等，若相等，则说明就是我们要查找的元素；否则，就顺序往后依次查找。如果遍历到数组的空闲位置还未找到，就说明要查找的元素并没有在散列表中。

删除数据：为了不让查找算法失效，可以将删除的元素特殊标记为deleted，当线性探测查找的时候，遇到标记为deleted的空间，并不是停下来，而是继续往下探测。

结论：最坏时间复杂度为O(n)

③二次探测（Quadratic probing）：线性探测每次探测的步长为1，即在数组中一个一个探测，而二次探测的步长变为原来的平方。

④双重散列（Double hashing）：使用一组散列函数，直到找到空闲位置为止。

⑤线性探测法的性能描述：

用“装载因子”来表示空位多少，公式：散列表装载因子=填入表中的个数/散列表的长度。

装载因子越大，说明空闲位置越少，冲突越多，散列表的性能会下降。

3.链表法（更常用）

插入数据：当插入的时候，我们需要通过散列函数计算出对应的散列槽位，将其插入到对应的链表中即可，所以插入的时间复杂度为O(1)。

查找或删除数据：当查找、删除一个元素时，通过散列函数计算对应的槽，然后遍历链表查找或删除。对于散列比较均匀的散列函数，链表的节点个数k=n/m，其中n表示散列表中数据的个数，m表示散列表中槽的个数，所以是时间复杂度为O(k)。

四、思考

1.Word文档中单词拼写检查功能是如何实现的？

字符串占用内存大小为8字节，20万单词占用内存大小不超过20MB，所以用散列表存储20万英文词典单词，然后对每个编辑进文档的单词进行查找，若未找到，则提示拼写错误。

2.假设我们有10万条URL访问日志，如何按照访问次数给URL排序？
字符串占用内存大小为8字节，10万条URL访问日志占用内存不超过10MB，通过散列表统计url访问次数，然后用TreeMap存储散列表的元素值（作为key）和数组下标值（作为value）
3.有两个字符串数组，每个数组大约有10万条字符串，如何快速找出两个数组中相同的字符串？
分别将2个数组的字符串通过散列函数映射到散列表，散列表中的元素值为次数。注意，先存储的数组中的相同元素值不进行次数累加。最后，统计散列表中元素值大于等于2的散列值对应的字符串就是两个数组中相同的字符串。

极客时间文档：https://time.geekbang.org/column/article/64233