哈希表

前面讨论的各种数据结构，记录在结构中的位置是随机的，和记录的关键字之间不存在确定的关系；因此在查找记录时需要进行一系列和关键字的比较

理想的情况下是希望不经过任何比较，一次存取便能得到所查的记录；就必须在记录的存储位置和关键字之间建立一个对应的关系f；

因为查找时，只要根据这个对应关系f找到给定值K的像f(K)；若结构中存在关键字和K相等的记录，则必定在f(K)的存储位置上；

在此，我们称这个对应关系f为哈希(Hash)函数，按这个思想建立的表为哈希表；

这一映像过程称为哈希造表或散列，所得存储位置称为哈希地址或散列地址

特点

哈希函数是一个映像，因此哈希函数的设定很灵活

冲突

对不同关键字可能得到同一个哈希地址，即key1 != key2，但是f(key1) == f(key2)， 这种现象称为冲突

具有相同函数值的关键字对该哈希函数来说称作同义词(synonym)

然而在一般情况下，冲突只可能尽可能地减少，而不能完全避免

哈希函数的构造方法

若对于关键字集合中的任一关键字，经哈希函数映像到地址集合中任何一个地址的概率是相等的，则称此类哈希函数为均匀的(Uniform)哈希函数

常用的构造哈希函数的方法有：

直接定址法

取关键字或关键字某个线性函数值为哈希地址

H(key) = key 或 H(key) = a * key + b

其中a和b为常数

数字分析法

假设关键字是以r为基数的数，并且哈希表中可能出现的关键字都是事先知道的，则可取关键字的若干位组成哈希地址

平方取中法

取关键字平方后的中间几位为哈希地址

折叠法

将关键字分割成几位位数相同的几部分，然后取着几部分的叠加和（舍去进位）作为哈希地址

除留余数法

取关键字被某个不大于哈希表表长的m的数p除后所得余数为哈希地址

H(key) = key MOD p， p <= m

处理冲突的方法

均匀的哈希函数可以减少冲突，但不能避免

通常用的处理冲突的方法有下列几种：

开放定址法

Hi = (H(key) + di) MOD m，i = 1，2，...，k（k <= m - 1）

其中H(key)为哈希函数；m为哈希表表长；di为增量序列；

di有三种取法：

1. 线性探测再散列；di = 1，2，3，...，m - 1
2. 二次探测再散列；di = 1^2，-1^2，2^2，-2^2，...，+-k^2
3. 随机探测再散列；di = 伪随机数序列

两个第一个哈希地址不同的记录争夺同一个后继哈希地址的现象叫做“二次聚集”，即在处理同义词的冲突过程中又添加了非同义词的冲突

再哈希法

Hi = RHi(key)， i = 1，2，...，k

RHi均是不同的哈希函数，即在同义词产生地址冲突时计算另一个哈希函数地址，直到冲突不再发生；

这种方法不易产生聚集，但增加了计算的时间

链地址法

将所有关键字为同义词的记录存储在同一线性链表中

建立公共溢出区

假设哈希函数的值域为[0，m - 1]，则设向量HashTable[0, m - 1]为基本表，每个分量存放一条记录

另设向量OverTable[0..v]为溢出表

所有关键字和基本表中关键字为同义词的记录，不管他们由哈希函数得到的哈希地址是什么，一旦发生冲突，都填入溢出表

哈希表的查找

在哈希表上进行查找的过程和哈希造表的过程基本一致

给定K值，根据造表时设定的哈希函数求得哈希地址，若表中此位置没有记录，则查找不成功；

否则比较关键字，若和给定值相等，则查找成功；

否则根据造表时设定的处理冲突的方法找“下一地址”，直到哈希表中某个位置为空或者表中所填记录的关键字等于给定值为止