数据结构

（C语言版）（第2版）

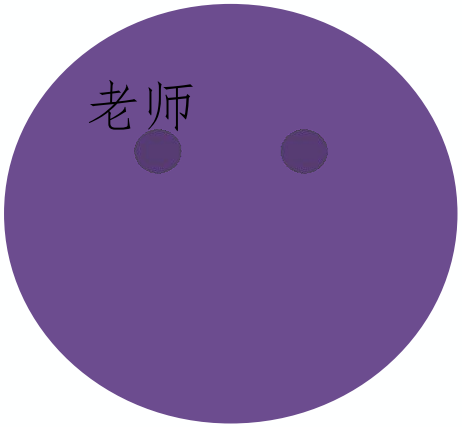
查找

哈希表的查找

主讲教师：汪红松

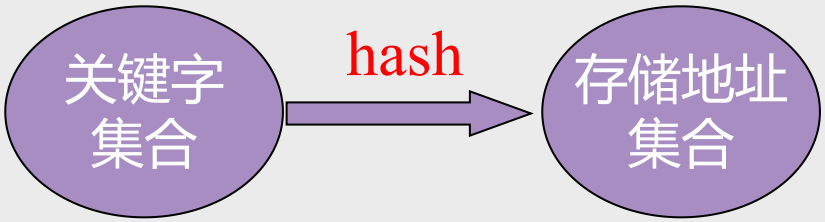
教 学 内 容 Contents

一、哈希表的查找
二、冲突
三、哈希函数的构造方法
四、处理冲突的方法



一、哈希表的查找

- 基本思想：记录的存储位置与关键字之间存在对应关系，Loc(i)=H(keyi) ⟶ 哈希函数。



- 优点：查找速度极快O(1),查找效率与元素个数n无关。

一、哈希表的查找

1.如何查找

地址	…	9	…	11	…	14	…	23	24	25	…	39	…
内容		9		11		14		23		25		39	

根据哈希函数H（k）=k

查找key=9,则访问H(9)=9号地址，若内容为9则成功；

若查不到，则返回一个特殊值，如空指针或空记录。

一、哈希表的查找

2.有关术语

哈希方法(杂凑法)

选取某个函数，依该函数按关键字计算元素的存储位置，并按此存放；

查找时，由同一个函数对给定值k计算地址，将k与地址单元中元素关键码进行比，确定查找是否成功。

哈希函数(杂凑函数)：

哈希方法中使用的转换函数

一、哈希表的查找

2.有关术语

哈希表(杂凑表)：按上述思想构造的表

地址	…	9	…	11	…	14	…	23	24	25	…	39	…
内容		9		11		14		23		25		39	

冲 突：不同的关键码映射到同一个哈希地址

$key1 \neq key2$，但$H(key1)=H(key2)$

同义词：具有相同函数值的两个关键字。

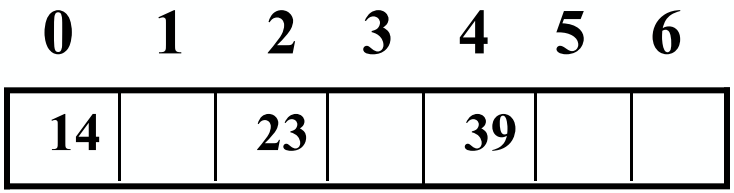
二、冲突

1.冲突现象举例

（14，23，39，9，25，11）

哈希函数：H(k)=k mod 7

6个元素用7个地址应该足够!



H(14)=14%7=0

H(25)=25%7=4

H(11)=11%7=4

同义词

有冲突

二、冲突

2.如何减少冲突

冲突是不可能避免的

构造好的哈希函数；

制定一个好的解决冲突方案。

三、哈希函数的构造方法

根据元素集合的特性构造

地址空间尽量小

均匀

1. 直接定址法
2. 数字分析法
3. 平方取中法
4. 折叠法
5. 除留余数法
6. 随机数法

三、哈希函数的构造方法

1.直接定址法

$Hash(key) = a \cdot key + b$　(a、b为常数)

优点：

以关键码key的某个线性函数值为哈希地址，不会产生冲突。

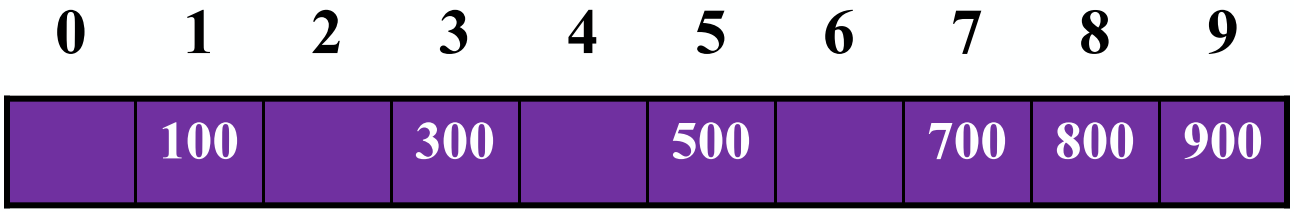
缺点：

要占用连续地址空间，空间效率低。

三、哈希函数的构造方法

1.直接定址法

例：{100，300，500，700，800，900}，
哈希函数Hash(key)=key/100



三、哈希函数的构造方法

2.除留余数法（最常用重点掌握）

Hash(key)=key mod p　(p是一个整数)

关键：

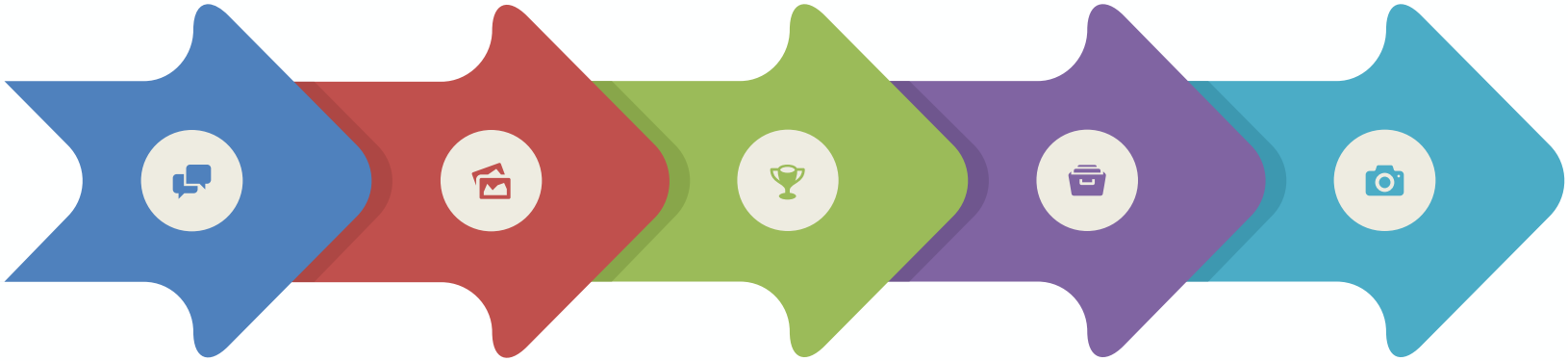
如何选取合适的p？

技巧：

设表长为m，取p≤m且为质数。

三、哈希函数的构造方法

3.构造哈希函数考虑的因素



① 执行速度（即计算哈希函数所需时间）；

② 关键字的长度；

③ 哈希表的大小；

④ 关键字的分布情况；

⑤ 查找频率。

四、处理冲突的方法

1.开放定址法

2.链地址法

四、处理冲突的方法

1.开放定址法（开地址法）

基本思想：有冲突时就去寻找下一个空的哈希地址，只要哈希表足够大，空的哈希地址总能找到，并将数据元素存入。

线性探测法

二次探测法

伪随机探测法

四、处理冲突的方法

1.开放定址法（开地址法）

（1）线性探测法

$H_i=(Hash(key)+d_i) \bmod m \quad (1 \le i < m)$

其中：m为哈希表长度

d_i 为增量序列 1，2，…m-1，且$d_i=i$

一旦冲突，就找下一个空地址存入

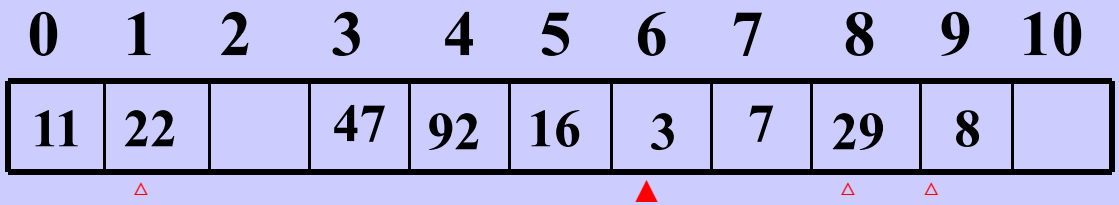
四、处理冲突的方法

1.开放定址法（开地址法()1）线性探测法

关键码集为 {47，7，29，11，16，92，22，8，3}，

设：哈希表表长为m=11；

哈希函数为Hash(key)=key mod 11 。



① 47、7、11、16、92没有冲突。

② Hash(29)=7，有冲突，由H_1=(Hash(29)+1) mod 11=8，哈希地址8为空，因此将29存入。

③ 3 连续移动了两次。

四、处理冲突的方法

1.开放定址法（开地址法）

（1）线性探测法

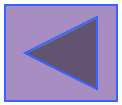
优点：

只要哈希表未被填满，保证能找到一个空地址单元存放有冲突的元素。

缺点：

可能使第i个哈希地址的同义词存入第i+1个地址，这样本应存入第i+1个哈希地址的元素变成了第i+2个哈希地址的同义词，……，产生“聚集”现象，降低查找效率。

解决方案：二次探测法



四、处理冲突的方法

1.开放定址法（开地址法）(2）二次探测法

关键码集为 {47，7，29，11，16，92，22，8，3}，

设：　哈希函数为Hash(key)=key mod 11

$H_i=(Hash(key)\pm d_i)\ mod\ m$

其中：m为哈希表长度，m要求是某个4k+3的质数；

d_i为增量序列 1^2，-1^2，2^2，-2^2，…，q^2

0	1	2	3	4	5	6	7	8	9	10
11	22	3	47	92	16		7	29	8	

△ ▲ △ △

Hash(3)=3，哈希地址冲突，由

$H_1=(Hash(3)+1^2)\ mod\ 11=4$，仍然冲突；

$H_2=(Hash(3)-1^2)\ mod\ 11=2$，找到空的哈希地址，存入。

四、处理冲突的方法

1.开放定址法（开地址法）

（3）伪随机探测法

$H_i=(Hash(key)+d_i) \bmod m \quad (1 \le i < m)$

其中：m为哈希表长度

d_i 为随机数

四、处理冲突的方法

1.开放定址法（开地址法）

step1

取数据元素的关键字key，计算其哈希函数值（地址）。若该地址对应的存储 空间还没有被占用，则将该元素存入；否则执行step2解决冲突。

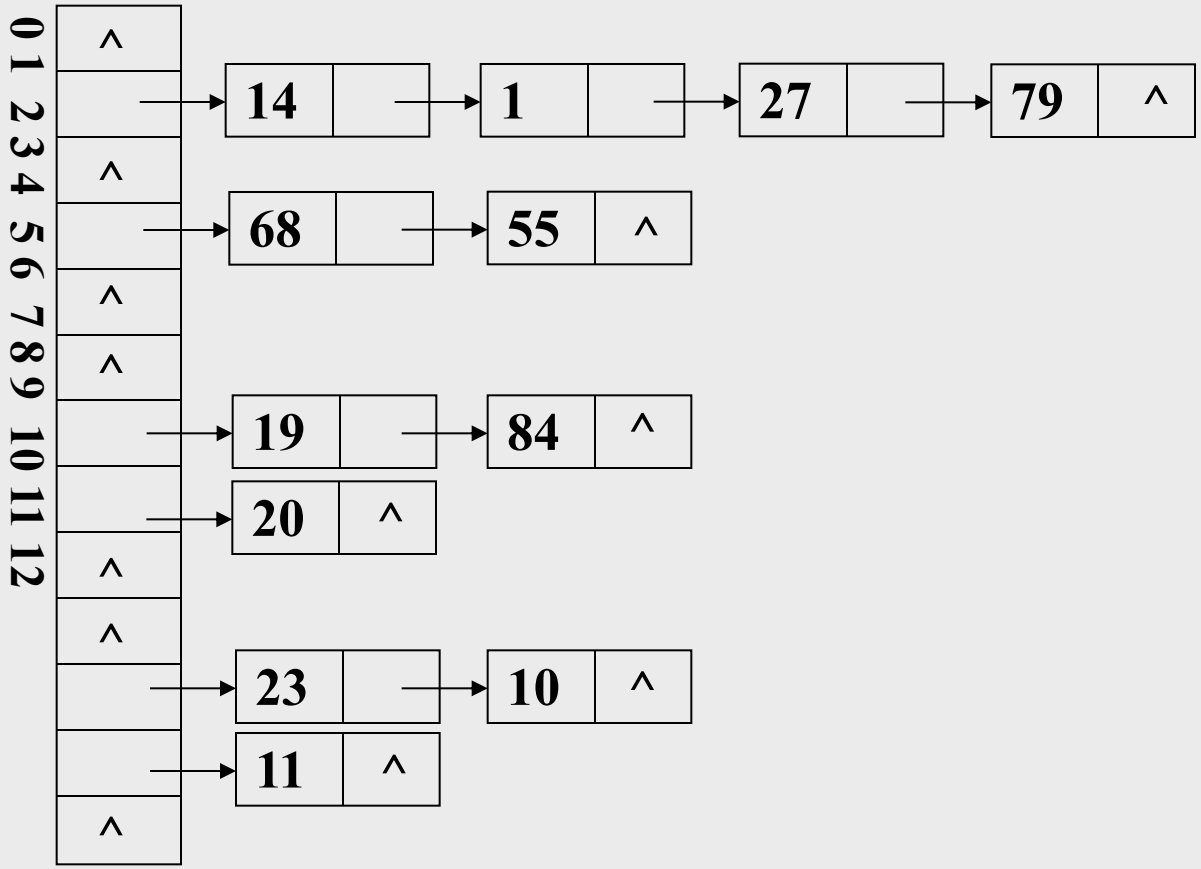
step2

根据选择的冲突处理方法，计算关键字key的下一个存储地址。若下一个存储地址仍被占用，则继续执行step2，直到找 到能用的存储地址为止。

四、处理冲突的方法

2.链地址法(拉链法）

基本思想：相同哈希地址的记录链成一单链表，m个哈希地址就设m个单链表，然后用用一个数组将m个单链表的表头指针存储起来，形成一个动态的结构。



四、处理冲突的方法

2.链地址法(拉链法）

（1）链地址法建立哈希表步骤

step1

取数据元素的关键字key，计算其哈希函数值（地址）。若该地址对应的链表为空，则将该元素插入此链表；否则执行step2解决冲突。

step2

根据选择的冲突处理方法，计算关键字key的下一个存储地址。若该地址对应的链表为不为空，则利用链表的前插法或后插法将该元素插入此链表。

四、处理冲突的方法

2.链地址法(拉链法）

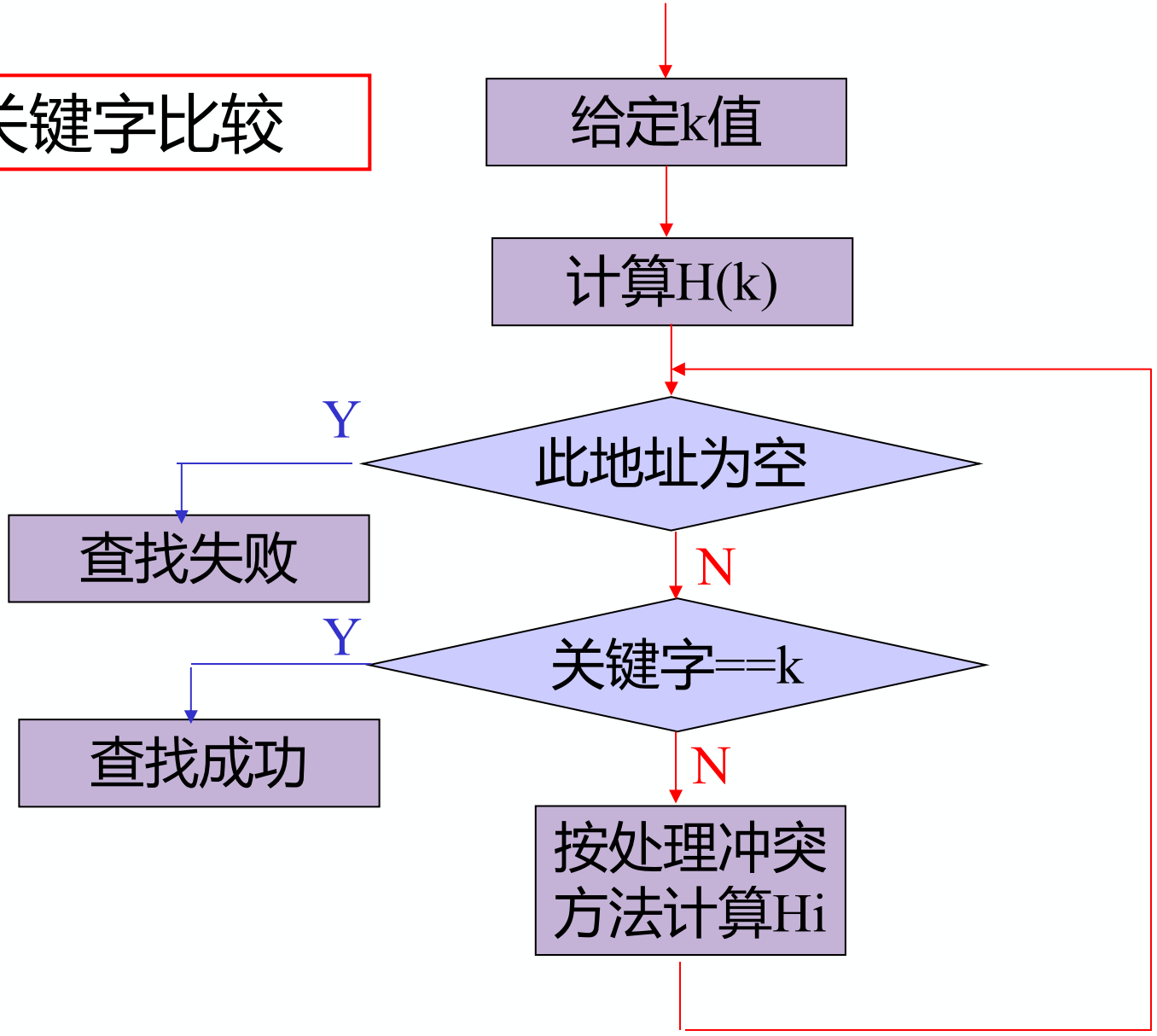
（2）链地址法的优点

非同义词不会冲突，无“聚集”现象；

链表上结点空间动态申请，更适合于表长不确定的情况。

一、哈希表的查找

给定值与关键字比较



一、哈希表的查找

ASL=(1*6+2+3*3+4+9)/12=2.5

已知一组关键字(19,14,23,1,68,20,84,27,55,11,10,79)

哈希函数为：H(key)=key MOD 13, 哈希表长为m=16，

设每个记录的查找概率相等。

(1) 用线性探测再散列处理冲突，即Hi=(H(key)+di) MOD m。

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	14	1	68	27	55	19	20	84	79	23	11	10			
	1	2	1	4	3	1	1	3	9	1	1	3			

H(19)=6

H(14)=1

H(23)=10

H(1)=1 冲突，H1=(1+1) MOD16=2

H(68)=3

H(20)=7

H(27)=1 冲突，H1=(1+1)MOD16=2

冲突，H2=(1+2)MOD16=3

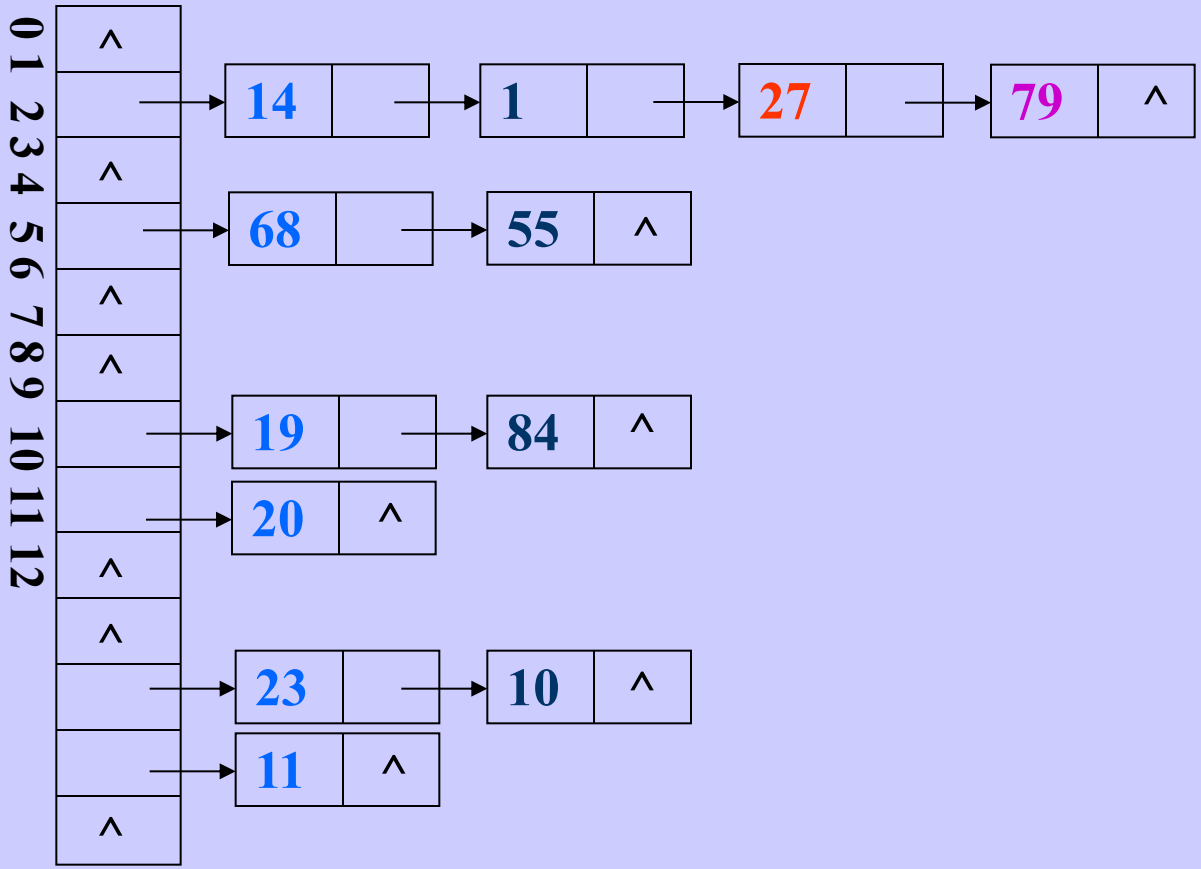
冲突，H3=(1+3)MOD16=4

一、哈希表的查找

ASL=(1*6+2*4+3+4)/12=1.75

(2) 用链地址法处理冲突

关键字(19,14,23,1,68,20,84,27,55,11,10,79)



一、哈希表的查找

哈希表的查找效率分析

使用平均查找长度ASL来衡量查找算法，

ASL取决于：

- ✓ 哈希函数
- ✓ 处理冲突的方法
- ✓ 哈希表的装填因子

O(1)?

α 越大，表中记录数越多，说明表装得越满，发生冲突的可能性就越大，查找时比较次数就越多。

$$\alpha = \frac{\text{表中填入的记录数}}{\text{哈希表的长度}}$$

一、哈希表的查找

哈希表的查找效率分析

ASL与装填因子α有关！既不是严格的O(1)，也不是O(n)

$$ASL \approx 1 + \frac{\alpha}{2}$$ （拉链法）

$$ASL \approx \frac{1}{2}\left(1 + \frac{1}{1-\alpha}\right)$$ （线性探测法）

$$ASL \approx -\frac{1}{\alpha}\ln(1-\alpha)$$ （随机探测法）

几点结论

对哈希表技术具有很好的平均性能，优于一些传统的技术。

链地址法优于开地址法。

除留余数法作哈希函数优于其它类型函数。

小结

1. 散列表的基本概念
2. 散列函数的构造及冲突的处理方法
3. 散列查找的方法和效率