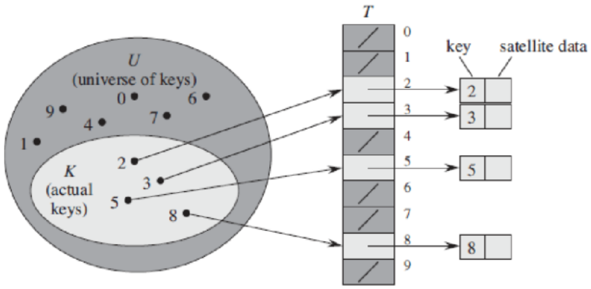
Hashtable

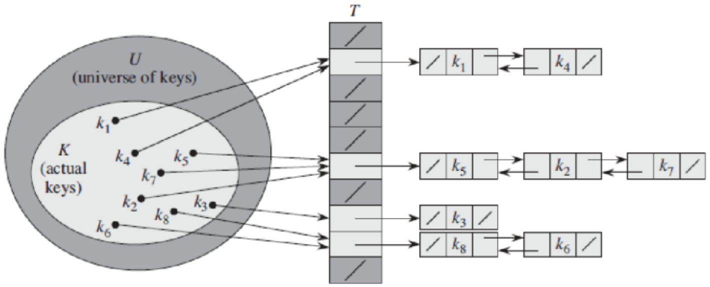
问题1：dictionary

- 什么是dictionary？
- 你如何理解它的两种实现？
  - direct-address table
  - hash table

many algorithms need only the ability to insert elements into, delete elements from, and test membership in a set. We call a dynamic set that supports these operations a **dictionary**. (p250, TC)

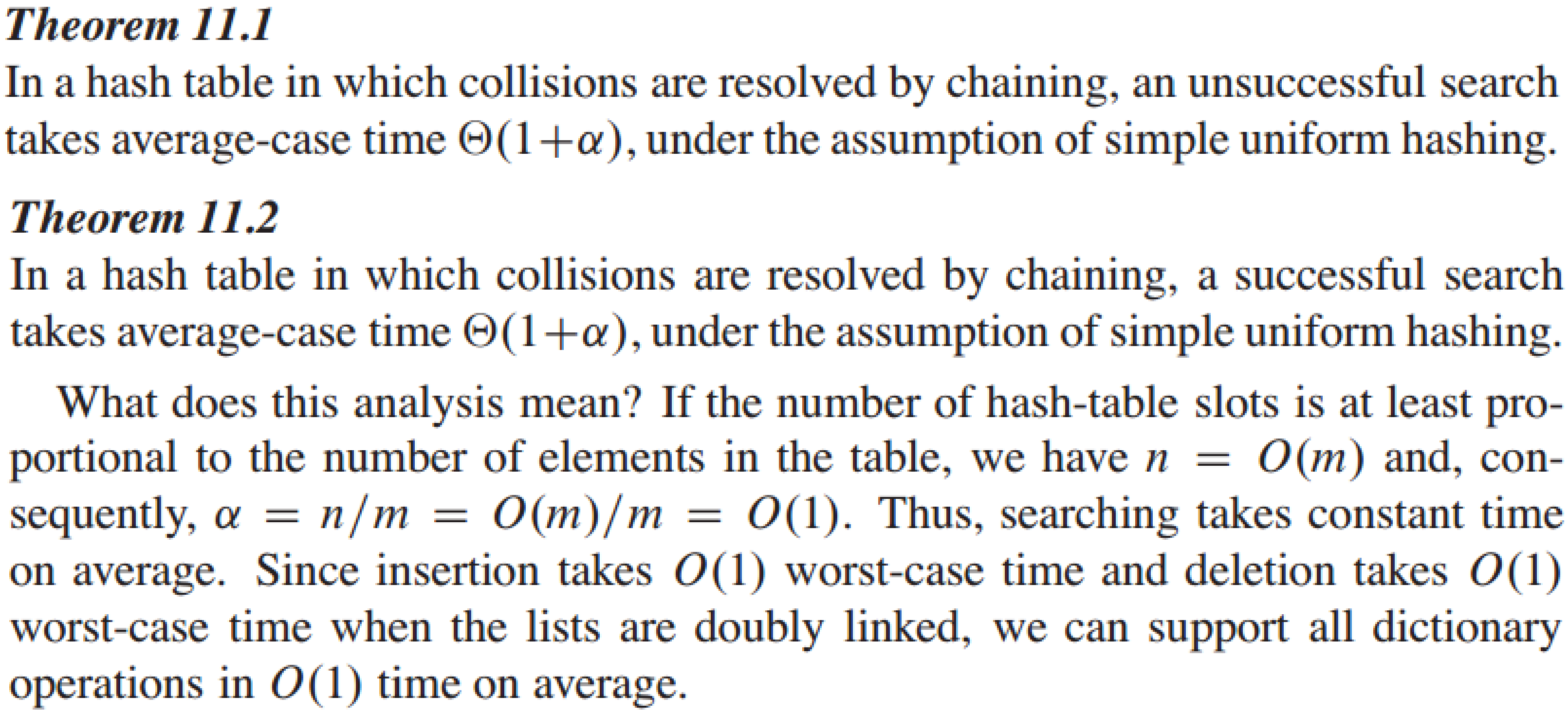
- 你能分析它们的存储空间和插入/删除/查找时间吗？
- 因此，你能对比它们的优缺点吗？





问题1： dictionary (续)

- 你理解这段话了吗？



Theorem 11.1

In a hash table in which collisions are resolved by chaining, an unsuccessful search takes average-case time $\Theta(1+\alpha)$, under the assumption of simple uniform hashing.

Theorem 11.2

In a hash table in which collisions are resolved by chaining, a successful search takes average-case time $\Theta(1+\alpha)$, under the assumption of simple uniform hashing.

What does this analysis mean? If the number of hash-table slots is at least proportional to the number of elements in the table, we have $n = O(m)$ and, consequently, $\alpha = n/m = O(m)/m = O(1)$. Thus, searching takes constant time on average. Since insertion takes $O(1)$ worst-case time and deletion takes $O(1)$ worst-case time when the lists are doubly linked, we can support all dictionary operations in $O(1)$ time on average.

如何做到那个“if”？

问题2： hash function

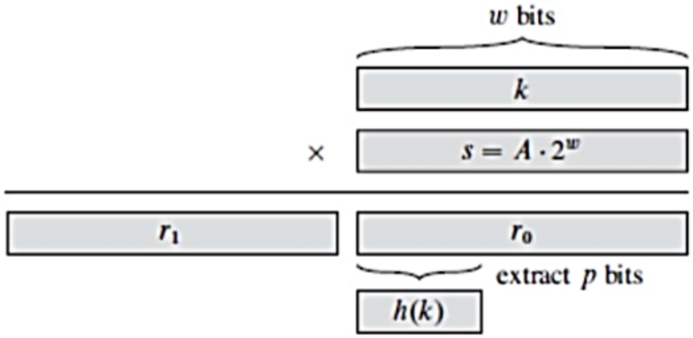
- 你如何理解一个好的hash function应有的这些要素？
  - Satisfies (approximately) the assumption of simple uniform hashing.
  - Derives the hash value in a way that we expect to be independent o f any patterns that might exist in the data.
- 你如何理解simple uniform hashing？ 它对hash table为什么至关重要？

问题2： hash function (续)

- 你理解这两种hash function了吗？

$$h(k) = k \bmod m$$

$$h(k) = \lfloor m\,(kA \bmod 1) \rfloor$$



- 这些hash function在实际中能确保是 simple uniform hashing吗？ 如果不能，怎么办？
  - universal hashing: to choose the hash function randomly in a way that is independent of the keys that are actually going to be

问题3：
probability calculations in hashing

- 你会计算这些期望值吗？
  - expected number of items per location

n/k

  - expected number of empty locations

$$k\left(1-\frac{1}{k}\right)^{n}$$

  - expected number of collisions

$$\begin{aligned} E(\text{collisions}) &= n - E(\text{occupied locations}) \\ &= n - k + E(\text{empty locations}), \end{aligned}$$

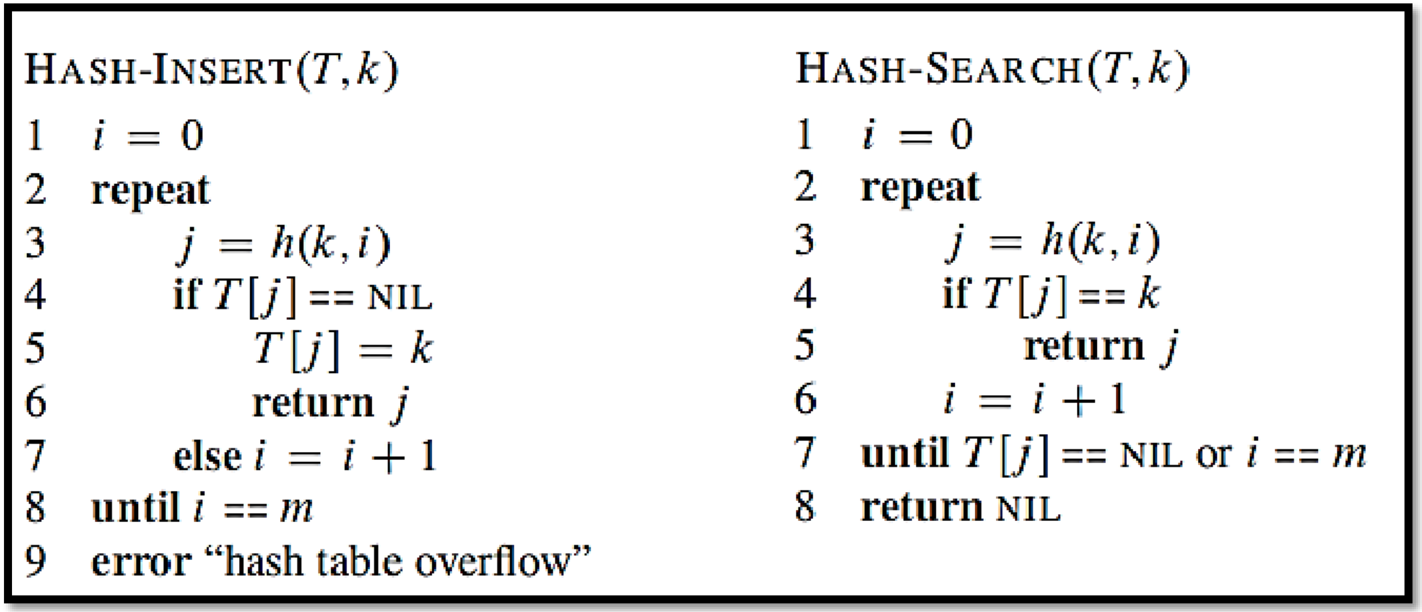
$$n-k+k\left(1-\frac{1}{k}\right)^{n}$$

  - expected time until all locations have at least one item

$$\begin{aligned} E(X) &= \sum_{j=1}^{k} E(X_j) \\ &= \sum_{j=1}^{k} \frac{k}{k-j+1} \end{aligned}$$

问题4： collision resolution

- 你理解open addressing了吗？ 它与chaining的本质区别是什么？ 因此，它有哪些相对的优缺点？



```
HASH-INSERT$(T, k)$
$i = 0$
repeat
    $j = h(k, i)$
    if $T[j] ==$ NIL
        $T[j] = k$
        return $j$
    else $i = i + 1$
until $i == m$
error "hash table overflow"
```

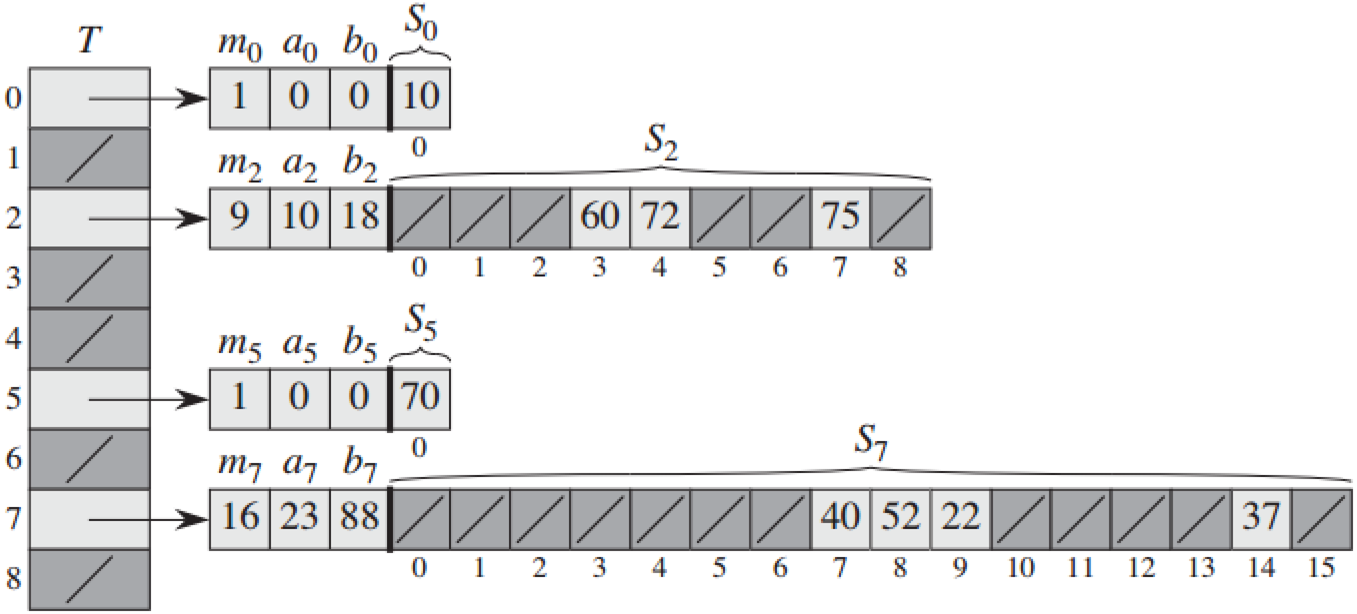
```
HASH-SEARCH$(T, k)$
$i = 0$
repeat
    $j = h(k, i)$
    if $T[j] == k$
        return $j$
    $i = i + 1$
until $T[j] ==$ NIL or $i == m$
return NIL
```

问题4：collision resolution (续)

- 一个好的h函数应该具有哪些特点？
  - The probe sequence is a permutation of <0, 1, …, m-1>.
  - uniform hashing: The probe sequence of each key is equally likely to be any of the m! permutations of <0, 1, …, m-1>.
- 你理解这些h函数了吗？它们为什么不是最好的h函数？
  - linear probing $h(k, i) = (h'(k) + i) \bmod m$
  - quadratic probing $h(k, i) = (h'(k) + c_1 i + c_2 i^2) \bmod m$
  - double hashing $h(k, i) = (h_1(k) + i h_2(k)) \bmod m$
- 你理解这些具体原因了吗？
  - linear probing: primary clustering
  - quadratic probing: secondary clustering

问题4： collision resolution (续)

- 你理解perfect hashing了吗？ 它与chaining的本质区别是什么？ 因此，它有哪些相对的优缺点？


T
0
1
2
3
4
5
6
7
8
m_0 a_0 b_0 S_0
1 0 0 10
0
m_2 a_2 b_2 S_2
9 10 18 60 72 75
0 1 2 3 4 5 6 7 8
m_5 a_5 b_5 S_5
1 0 0 70
0
m_7 a_7 b_7 S_7
16 23 88 40 52 22 37
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15