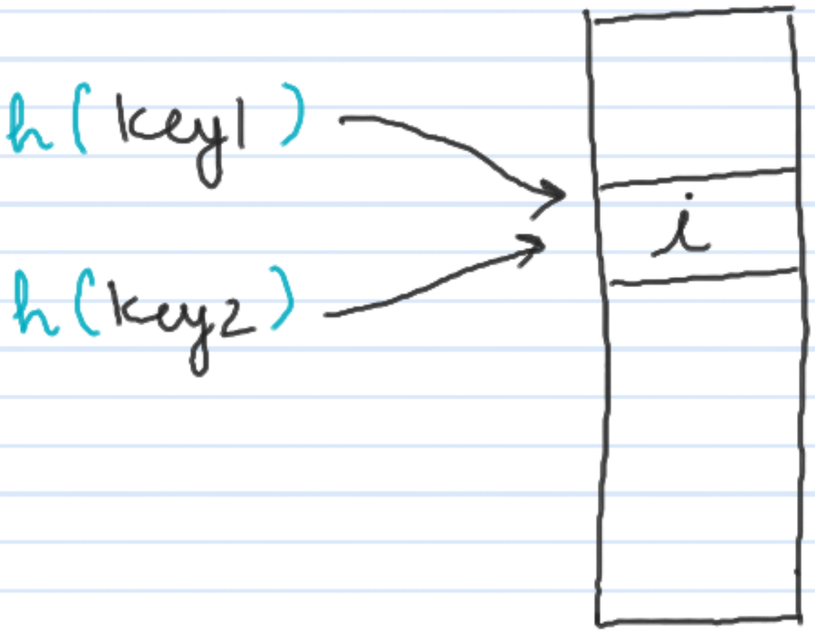
CS2700 : Programming and Data Structures

Hashing : collision resolution

- Chaining
- Open Addressing
    - Linear Probing
    - Quadratic Probing
    - Double Hashing

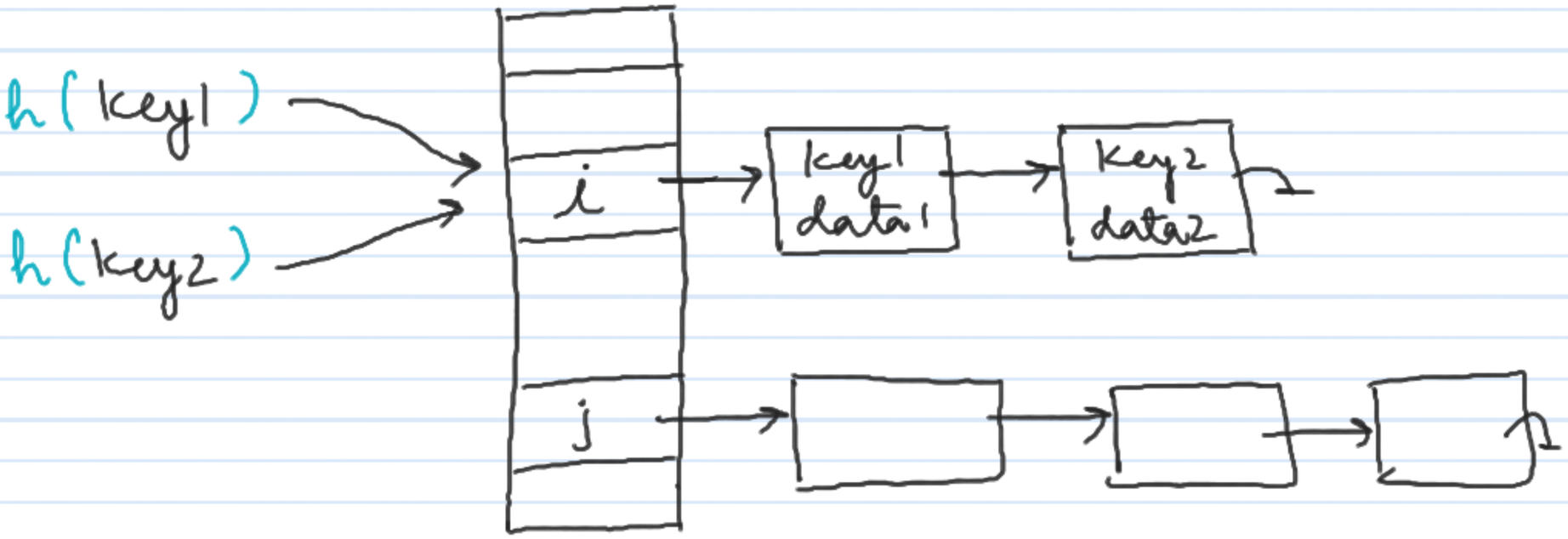
Collision Resolution : Separate Chaining



- We would like to store data corresp. to key1 and key2 (and possibly more keys) in the index i

all such keys have same hash value

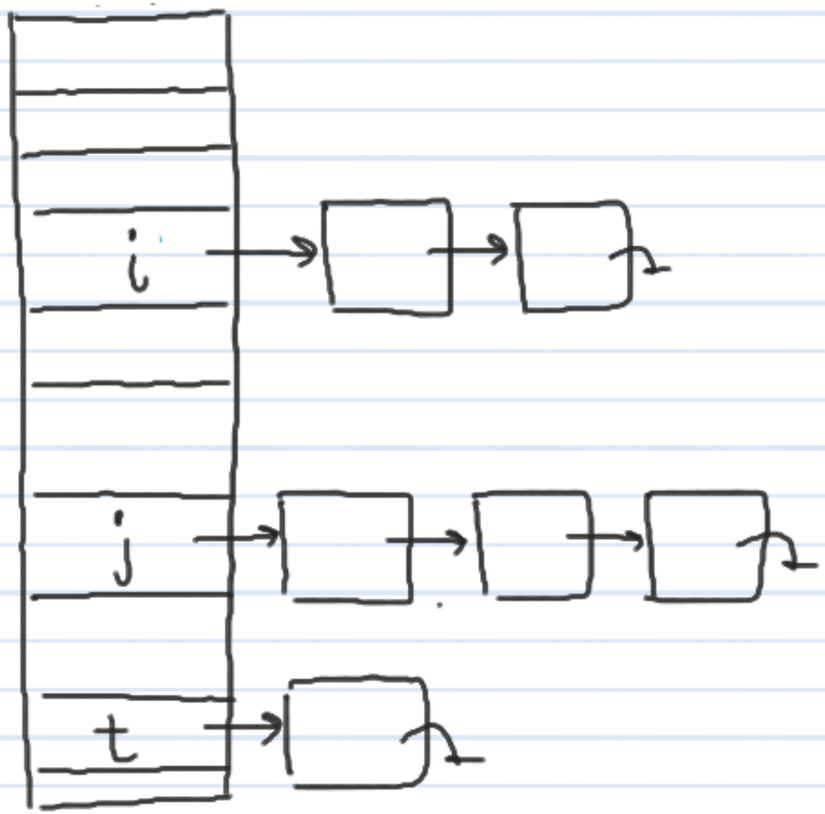
Collision Resolution : Separate Chaining



- We would like to store data corresp. to key1 and key2 (and possibly more keys) in the index i

all such keys have same hash value

Collision Resolution : Separate Chaining



Insert (key, data)

```
i = h(key)
insert into list A[i]
```

Search (key)

```
i = h(key)
find in list A[i]
```

Delete (key)

```
i = h(key)
delete from list A[i]
```

Note the importance of storing keys along with data.

DISADVANTAGES OF CHAINING ?

LOAD FACTOR OF A HASH TABLE

n : # of keys in hash table

m : size of hash table (# of slots)

α : LOAD FACTOR

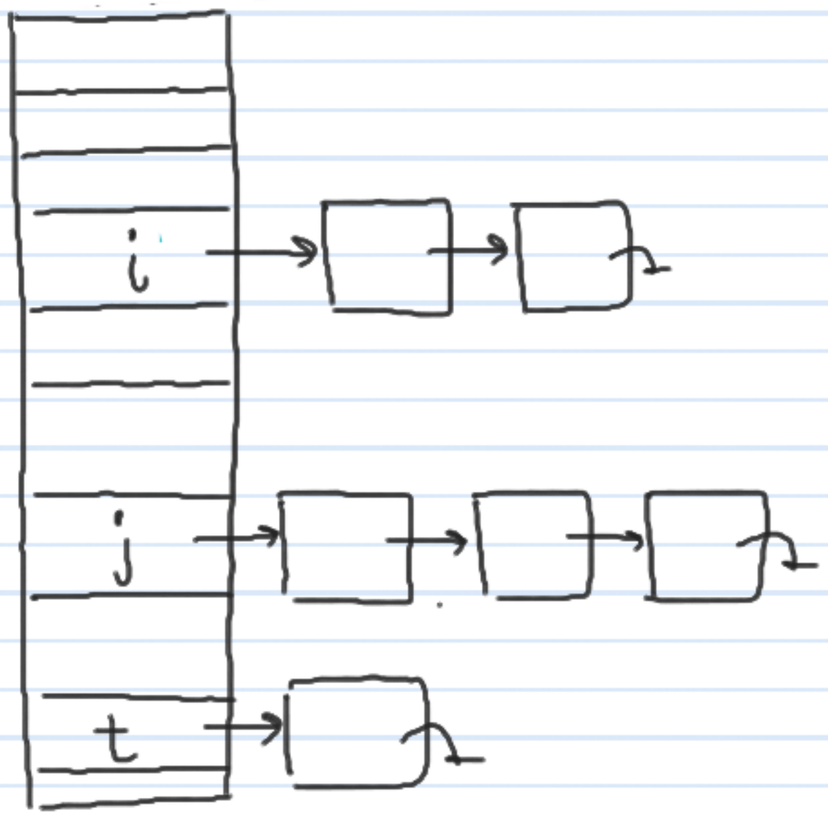
$\alpha = n/m$

Measure of how full the hash table is

In separate chaining α denotes average length of linked list

caution : bad hash function can lead to large α

Collision Resolution : Separate Chaining



Insert (key, data)
    i = h(key)
    insert into list A[i]
} O(1)

Search (key)
    i = h(key)
    find in list A[i]
} O(1+α)

Delete (key)
    i = h(key)
    delete from list A[i]
} O(1+α)

If α is bounded by a constant average time is constant, note that worst case may still be bad!

Collision Resolution : OPEN ADDRESSING.

- Chaining needs two data structures
  (array, aux DS for handling collisions

  this DS can be list, BST, or even a Hash table again!

- Typically needs pointers and cache inefficient.
- Can we do everything within the array?

Collision Resolution : OPEN ADDRESSING.

- Chaining needs two data structures
  (array, aux DS for handling collisions

  this DS can be list, BST, or even a Hash table again!

- Typically needs pointers and cache inefficient.
- Can we do everything within the array?

Collision Resolution : OPEN ADDRESSING.

- In chaining hash value for a key is fixed (not open)
- When hash value can be "open" (not fixed) it is called open addressing.

How to search with "open" hash values

OPEN ADDRESSING :

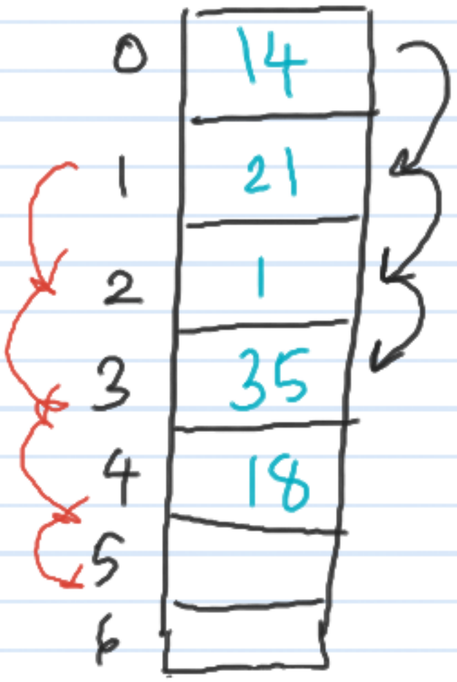
- Hashed items reside in the array itself (Hash table)
- Hash value gives an index (home address)
- Multiple probes may be needed to find key.

OPEN ADDRESSING : Linear Probing.

When collision occurs scan forward for available empty slot

Example :

$h(key) = key \% 7;$



insert 18, 14, 21, 1, 35,

find (35)

how many probes are needed?

find (8)

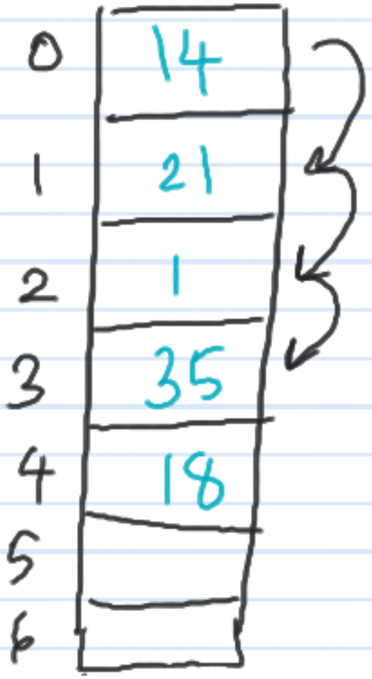
when to stop? How many probes?

OPEN ADDRESSING : Linear Probing.

When collision occurs scan forward for available empty slot

Example :

$h(key) = key \% 7;$

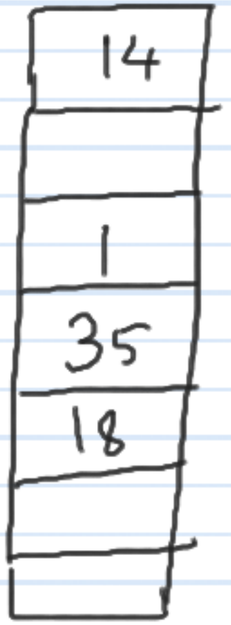


insert 18, 14, 21, 1, 35,

delete (21)

find (35)

↳ if our stopping criteria is empty slot there is an issue!

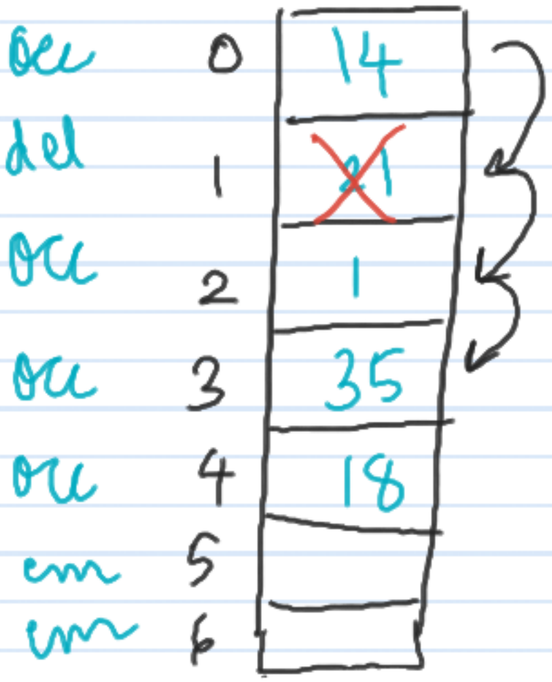


OPEN ADDRESSING : Linear Probing.

Deletion : lazy deletion

↳ simply mark the entry as deleted

Example :



3 states for every slot :

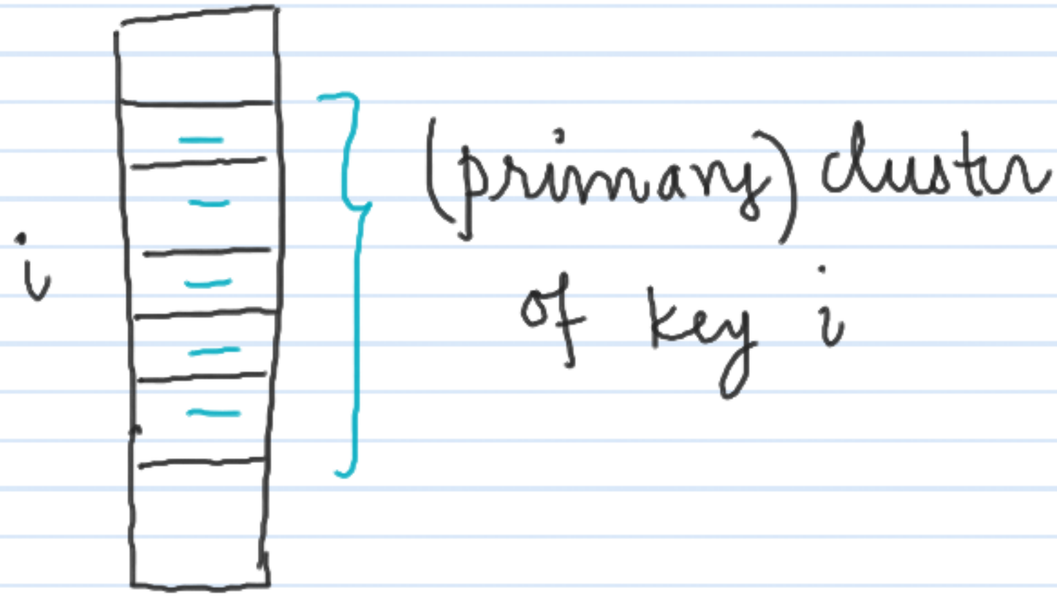
- Occupied
- deleted
- empty

} needs table size aux space.

deleted and empty are treated diff by insert and find.

OPEN ADDRESSING : Linear probing (issue of clustering)

CLUSTER : A set of consecutive locations that are occupied.



- Large clusters slow down search and insert times
- clusters of 2 keys can merge forming an even larger cluster

Can we leave gaps?

OPEN ADDRESSING : QUADRATIC PROBING.

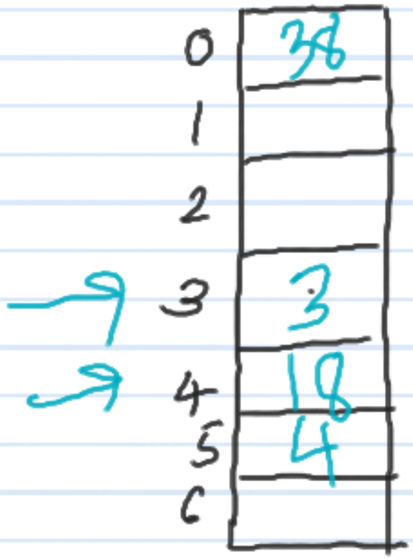
collision resolution function is quadratic

Linear probing : $h(key), h(key)+1, h(key)+2, \ldots$

Quadratic probing : $h(key), h(key)+1^2, h(key)+2^2,$
$h(key)+3^2, \ldots$

Example : $h(key) = key \% 7$

Insert 3, 18, 4, 38



OPEN ADDRESSING : QUADRATIC PROBING.

collision resolution function is quadratic

Linear probing : $h(key), h(key)+1, h(key)+2, \ldots$

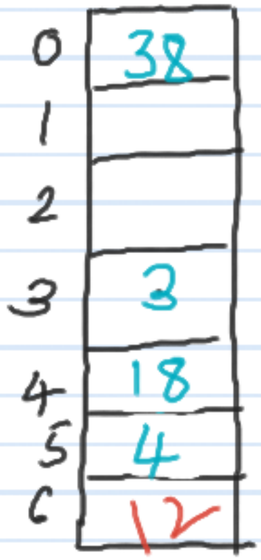
Quadratic probing : $h(key), h(key)+1^2, h(key)+2^2, h(key)+3^2, \ldots$

Example : $h(key) = key \% 7$

Insert 3, 18, 4, 38

probes : 3, 3+1, 3+4 = 0 (successful.)

Insert 12, 10 → what happens?



OPEN ADDRESSING : QUADRATIC PROBING.

Even if table is not full, our probes may not find a slot!

- does our probe finding terminate always? - no in general (even when table has slots)

Claim: If Table size is an odd prime > 3, and table is at least half empty, then quadratic probing is guaranteed to terminate.

why? $h(key) + i^2 = h(key) + j^2$

OPEN ADDRESSING : QUADRATIC PROBING.

SECONDARY CLUSTERS

h(key 1) → h
h(key 2) → h

} Same probes will be used
h, h+1², h+2², ...

These are called secondary clusters

↳ analogue of primary clusters in linear probing.

Can we generate different probe sequences?