Open Address, Cryptographic Hashing

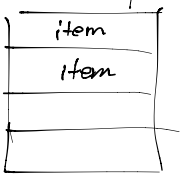
Open Addressing

- ~~Another~~ way ~~to~~ deal with collisions
- No chaining

- Array structure, one item per slot at most:

$m \geq n$

m: # slot, n: # elements



- Probing

Hash function specifies order for slots for probe for a key (for insert / search / delete)

$$h: \underbrace{U}_{\text{universe of keys}} \times \underbrace{\{0, 1, \cdots, m-1\}}_{\text{trial count}} \longrightarrow \{0, 1, \cdots m-1\}$$

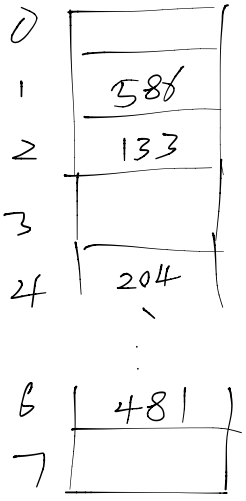
Arbitrary key k:

$h(k,1), h(k,2), \cdots\cdots, h(k, m-1)$

want above to be a permutation of $0, 1, \cdots m-1$

E.g.

Insert operation to a table.



Insert 586

$h(586, 1) = 1$

$h(481, 1) = 6$

Insert 496

$h(496, 1) = 4$ fails

$h(496, 2) = 1$ fails

$h(496, 3) = 3$ succeed

(3 trails to find a empty slot)

- Insert(k, v)

  keep probing until an empty slot is found. Insert item when found.

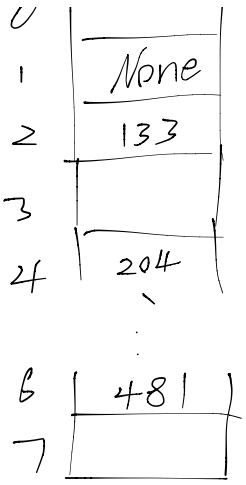
  given $m \geq n$, garanteed to find a slot

- Search(k)

  As long as the slot encounted are occupied by keys $\neq k$, keep probing until either encounter k or find an empty slot. (use the same deterministic probe as if you insert it $h(k, v)$)

- Delete

  Eg. delete (586)



Search(496) → find empty slot
→ failed search Incorrectly!

Replace deleted item with a different flag "Delete Me"
(different from None)

* Insert treats "Delete Me" the same as None, but Search treats it as "keep going" (different from None)

• Probing Strategies

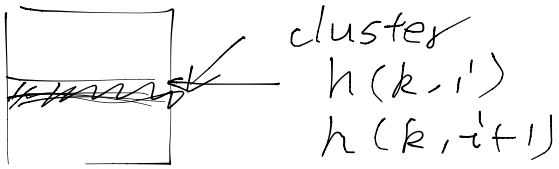
- Linear Probing

$h(k,i) = (h'(k) + i) \bmod m$

$h'(k)$ → ordinary hash function

Satisfy permutation

Problem: clusters: Consecutive groups of occupied slots which keep longer.



$k(k, i+2)$

for $0.01 < \alpha = \frac{n}{m} < 0.99$, will see cluster of size $\Theta(\lg n)$ → search/insert won't work in a const. time.

- Double hashing

  $h(k, i) = (h_1(k) + i * h_2(k)) \mod m$

  if $h_2(k)$ is relatively prime to m

  ↳ guarantee permutation

  e.g. $m = 2^r$, $h_2(k)$ is odd for all k

- Uniform hashing assumption

  (NOT the same as simple uniform hashing)

  Each key is equally likely to have any one of the $m!$ permutations as its probe sequence.

  if $\alpha = \frac{n}{m}$, Cost of operations (e.g. insert) $\leq \frac{1}{1-\alpha}$

  in pratice, resize the table when getting high α

Cryptography hashing

E.g. Password storage, even the sys. admin doesn't know my password

One way: given $h(x) = Q$ it is very hard to find x s.t. $h(x) = Q$

save to file: /etc/passwd: username X 12647

↳ type password x' (may be wrong)

2↳ compare $h(x') \stackrel{?}{=} h(x)$

(don't need x, already saved h:x))