February 28, 2023 1:50 PM

Problem Given set S of n numbers in $\{0, 1, \ldots, U-1\}$, build data structure s.t. we can quickly answer membership queries: given y, is $y \in S$?

("Dictionary")

e.g. sorted array
$\Rightarrow$ $O(\log n)$ query time
$O(n \log n)$ preprocessing time
$O(n)$ space

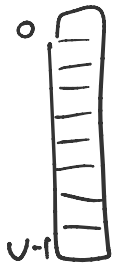
balanced search tree
$\Rightarrow$ $O(\log n)$ insert/delete time ← "dynamic"

("predecessor/successor search")

better?

Method 0.

bit vector



$O(1)$ time
$O(U)$ space

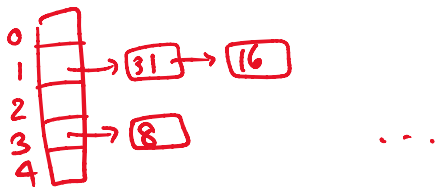
Method 1. hash table with chaining

Pick a hash fn $h: \{0, \ldots, U-1\} \to \{0, \ldots, m-1\}$ for some $m \ll U$.

Store array $A[0, \ldots, m-1]$
where $A[i]$ = list of all $x \in S$ with $h(x) = i$ ← "bucket"

e.g. $h(x) = x \bmod m$

$m = 5$



...

query(y):
search the list $A[h(y)]$

if input is rand, unif. distributed,
each bucket will have $\sim \frac{n}{m}$ elems
"on average"

Set $m \approx n \Rightarrow$ $O(n)$ space
$O(1)$ average query time
$O(n)$ preproc. time

but can't assume input is random!

Method 2. hashing with random hash fn
chosen a universal family
[Carter, Wegman '77]

Ex Assume U is prime.
Pick rand. $a \in \{1, .., U-1\}$
rand $b \in \{0, .., U-1\}$.

Define $h_{a,b} : \{0,.., U-1\} \to \{0,.., m-1\}$:

$$h_{a,b}(x) = \Big((ax+b) \bmod U \Big) \bmod m$$

computable in $O(1)$ time

Property For any fixed $x, y \in \{0,.., U-1\}$ $(x \neq y)$,

$$\Pr_{a,b}\Big[h_{a,b}(x) = h_{a,b}(y) \Big] \leq O\left(\frac{1}{m}\right).$$

(called universality)

$h_{a,b}(x) = h_{a,b}(y)$ ⇔ we say x, y collide

More strongly, for any fixed $x, y \in \{0,.., U-1\}$, $(x \neq y)$
$i, j \in \{0,.., m-1\}$,

$$\Pr_{a,b}\Big[h_{a,b}(x) = i \wedge h_{a,b}(y) = j \Big] \leq O\left(\frac{1}{m^2}\right).$$

(called strong 2-universality)

Pf: Fix $i', j' \in \{0,.., U-1\}$.

Pf: Fix $i', j' \in \{0, \ldots, U-1\}$.

$$\Pr_{a,b}\left[\begin{array}{l} (ax+b) \bmod U = i' \\ \wedge\ (ay+b) \bmod U = j' \end{array} \right]$$

$$= \Pr_{a,b}\left[\begin{array}{l} ax+b \equiv i' \pmod U \\ \wedge\ ay+b \equiv j' \end{array} \right]$$

2×2 linear system of eqns; soln is uniq.

$$= \frac{1}{U(U-1)}$$

$$\Leftarrow \begin{cases} a = (i'-j')(x-y)^{-1} \\ b = i' - ax \end{cases}$$

$$\Rightarrow \Pr_{a,b}\left[h_{a,b}(x) = i \wedge h_{a,b}(y) = j \right]$$

$$= \sum_{\substack{i', j' \in \{0, \ldots, U-1\} \\ i' \bmod m = i \\ j' \bmod m = j}} \frac{1}{U(U-1)}$$

$$\leq \lceil U/m \rceil \lceil U/m \rceil \cdot \frac{1}{U(U-1)}$$

$$\leq O\left(\frac{1}{m^2}\right). \qquad \square$$

Query time analysis:

for a fixed query y,

$$E[\text{query time}] \approx E[\#\text{ collisions with } y]$$

$$= E\left[\left| \{ x \in S - \{y\} : h_{a,b}(x) = h_{a,b}(y) \} \right| \right]$$

$$= E\left[\sum_{x \in S - \{y\}} 1_{[h_{a,b}(x) = h_{a,b}(y)]} \right]$$

(indicator fn: $1_E = \begin{cases} 1 & \text{if } E \text{ true} \\ 0 & \text{else} \end{cases}$)

($E(1_E) = 1 \cdot \Pr(E) + 0 \cdot \Pr(\bar{E})$)

$$= \sum_{x \in S - \{y\}} E\left[1_{[h_{a,b}(x) = h_{a,b}(y)]} \right]$$

by linearity of expectation

$$= \sum_{x \in S-\{y\}} \Pr\left[h_{a,b}(x) = h_{a,b}(y)\right]$$

$$\leq \sum_{x \in S-\{y\}} O\left(\frac{1}{m}\right)$$ by universality

$$= O\left(\frac{n}{m}\right)$$

Set $m \approx n \Rightarrow$

- $O(n)$ space
- $O(1)$ expected query time
- $O(n)$ preproc time
- $O(1)$ insert/delete expected time assuming update seq is indep of rand. choices

can we get good worst-case query time?

Assume static...

Alternative analysis:

$$E[\text{total \# colliding pairs } (x,y) \in S \times S]$$

$$= E\left[\sum_{y \in S} (\text{\# collisions with } y) \right]$$

$$= E\left[\sum_{y \in S} O\left(\frac{n}{m}\right) \right]$$

$$= O\left(n \cdot \frac{n}{m}\right).$$

Set $m = cn^2 \Rightarrow E[\text{total \# colliding pairs}] = O\left(\frac{1}{c}\right)$

$\Rightarrow Pr[\text{total \# colliding pairs} \geq 1]$

$\leq O\left(\frac{1}{c}\right)$.

by Markov's ineq

$\Rightarrow$ $O(n^2)$ space

$O(1)$ worst-case query time

perfect hashing

$O(n)$ expected preproc time

by repeating $O(c)$ expected # times.

Can space be reduced back to $O(n)$?

yes ... to be cont'd ...