

Lec31-hashing

Sunday, December 1, 2024 4:33 PM

Let's consider a dictionary. How do we look up words in a dictionary?

Binary search works. What about an index in a book?

That jumps us straight to the relevant page.

Dictionary ADT (Abstract Data Type)

- $add(x)$: add key x to S
 - $query(q)$: is key $q \in S$
 - $delete(x)$: remove key x from S
- } insertion only dictionary just needs two ops
 ← static dictionary only needs this
 E.g. English language changes slowly.

Static dictionary implementations

- sorted list
- binary tree
- skiplists

} also work for dynamic case
 $O(\log n)$ search

↳ can we do better? Want $O(1)$ search.

Hashing basics

- Keys from large universe U . e.g. all strings ≤ 100 char Andrew IDs
- e.g. numbers from 1 to 10^{12} . SSN
- e.g. all integers

- Set $S \subseteq U$ of keys we care about.

e.g.

Notice: we can encode most pertinent information as integers.

Ex. $U = \{1, 2, \dots, 10^{12}\}$

$N = |S| = 1,000,000$ (set S much smaller than U)

binary search tree / skiplists take $O(\log N)$ query time

Alt: bitstrings of size 10^{12} . Take $O(1)$ query time (assuming random access model)
↳ takes too much space

Idea: hash function $h: U \rightarrow \{0, \dots, M-1\}$, & use array A of size M .

Store $x \in S$ in $A[h(x)]$.


If lucky or if $M \gg N$, no collisions, so $O(1)$ query time.

Collision resolution: Keep linked list in each entry of A . (separate chaining)

Lookup time for $x = O(\text{length}(A[h(x)]))$.

Intuition: If $M = \Theta(N)$, and h spreads S out evenly in A , then $\text{length}(A[h(x)])$ is small.

Impossibility result: \forall hash function h , if $|U| \geq (N-1)M + 1$, then $\exists S$ of size $|S|=N$ that all hash to same location.

proof. Pigeon-hole principle. Suppose every location in A has only $N-1$ items from U that hash to it. Then $|U| \leq M(N-1)$, a contradiction. Thus, $\exists S$ with $|S|=N$ that all hash to one location. 

Randomness to the Rescue

Def. A randomized algorithm H for constructing hash functions $h: U \rightarrow \{0, \dots, M-1\}$ is universal if $\forall x \neq y \in U$,

$$\Pr_{h \in H} [h(x) = h(y)] \leq \frac{1}{M}.$$

Often, we will also say a set H of hash functions is a universal family of hash functions if the algorithm "choose $h \in H$ uniformly at random" is universal.

Important: By definition \ast any \ast 2 distinct items must collide with probability at most $\frac{1}{M}$.

Intuition: If $h: U \rightarrow [M]$ maps every $x \in U$ uniformly at random, then collision prob. is exactly $\frac{1}{M}$.

Alt interpretation of definition: Count all hash functions in family H .

In some, $h(x) = h(y)$ for a specific $x \neq y$.

In some, $h(x) = h(y)$ for a specific $x \neq y$.
 Then we need $\forall x \neq y \in U, \frac{|\{h \in H \mid h(x) = h(y)\}|}{|H|} \leq \frac{1}{M}$.

Example: $U = \{a, b\}$ $M = 2$
 $h: \{a, b\} \rightarrow \{0, 1\}$

✓

	a	b
h_1	0	0
h_2	0	1

$a \neq b$
 $h_1(a) = h_1(b)$
 $h_2(a) \neq h_2(b)$ } collide half the time $\leq \frac{1}{M}$

✓

	a	b
h_1	0	1
h_2	1	0

$h_1(a) \neq h_1(b)$
 $h_2(a) \neq h_2(b)$ } don't collide

✓

	a	b
h_1	0	0
h_2	1	0
h_3	0	1

$h_1(a) = h_1(b)$
 $h_2(a) \neq h_2(b)$
 $h_3(a) \neq h_3(b)$ } collide $\frac{1}{3}$ of time $\leq \frac{1}{M}$.

✗

	a	b
h_1	0	0
h_2	1	1

$h_1(a) = h_1(b)$
 $h_2(a) = h_2(b)$ } collide $1 \geq \frac{1}{2}$ of time

Ex. $U = \{a, b, c\}$ $M = 2$

✗

	a	b	c
h_1	0	0	1
h_2	1	1	0
h_3	1	0	1

$a \neq b$	$a \neq c$	$b \neq c$
collide	diff	diff
collide	diff	diff
diff	diff	collide
$\frac{2}{3}$	$\frac{0}{3}$	$\frac{1}{3}$

collisions: prob

↳ not universal because of single pair (a, b).

Ex. $U = \{a, b, c\}$ $M = 3$

	a	b	c
h_0	0	0	0
h_1	0	1	0
h_2	0	0	1

$a \neq b$	$a \neq c$	$b \neq c$
collide	collide	collide
collide	collide	collide

Notice: some hash functions can

	a	b	c
h_0	0	0	0
h_1	0	1	2
h_2	1	2	0
h_3	2	0	1

collision
prob

a + b	a + c	b + c
collide	collide	collide
diff	diff	diff
diff	diff	diff
diff	diff	diff

Note: some hash functions can be super bad while still being universal

$$\frac{1}{4} \quad \frac{1}{4} \quad \frac{1}{4} \leq \frac{1}{3}$$

\Rightarrow universal

Thm If H is universal, then $\forall S \subseteq U$ of size N , $\forall x \in U$, if we construct h at random according to H ($h \in H$ random), the expected number of collisions between x & other elements in S is $\leq \frac{N}{M}$.

proof: By universality, $\Pr(h(x) = h(y) \mid y \neq x) \leq \frac{1}{M}$.

$$\text{let } C_{xy} = \begin{cases} 1 & \text{if } h(x) = h(y) \\ 0 & \text{otherwise} \end{cases}$$

let C_x be r.v. denoting total # collisions for x .

$$C_x = \sum_{\substack{y \in S \\ y \neq x}} C_{xy}$$

$$\mathbb{E}[C_x] = \sum_y \mathbb{E}[C_{xy}] \leq \frac{N}{M}$$

\uparrow linearity of expectation



Thus, the expected lookup time is $O(\frac{N}{M})$.

If, e.s. $M > 2N$, then lookup is $O(1)$.

(assuming h takes $O(1)$ to compute)

Constructing universal hash function

Matrix method:

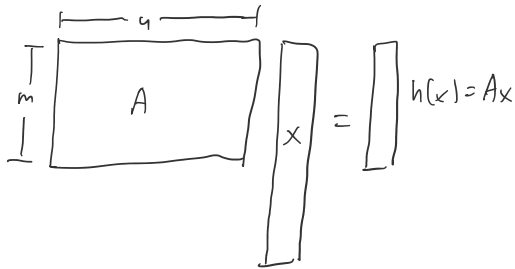
Let keys be u -bits long.

$$(U = \{0, 1\}^u \cong \{1, \dots, 2^u\})$$

Let $M = 2^m$, so indices are m -bits long.

Let $A \in \{0, 1\}^{m \times u}$ matrix where each entry is i.i.d. uniform.

Then let $h(x) = Ax$ where all additions are mod-2.



Claim: For $x \neq y$, $\Pr[h(x) = h(y)] = \frac{1}{M}$.

proof. Some bit $x_i \neq y_i$ since $x \neq y$. WLOG, say $x_i = 0$, $y_i = 1$.

Let's imagine setting all of A except col i .

Consider row j of A , and entries $[Ax]_j$, $[Ay]_j$.

Flipping A_{ji} will keep $[Ax]_j$ the same, while flipping $[Ay]_j$.

So some setting of A_{ji} causes $[Ax]_j = [Ay]_j$ and some setting $[Ax]_j \neq [Ay]_j$.

\Rightarrow then every entry has $\frac{1}{2}$ prob to be equal.

$\Rightarrow \Pr[Ax = Ay] = \left(\frac{1}{2}\right)^m$



Some Other methods involve finite field arithmetic for other primes.

We now can get $O(1)$ expected runtime for hash table.

Can we do better if S is static?

Perfect Hashing

Method 1: (quadratic space)

Theorem: If H is universal and $M = N^2$, then $\Pr_{h \in H}(\text{no collisions in } S) \geq \frac{1}{2}$.

proof. $\binom{N}{2}$ pairs (x, y) to check.

proof. $\binom{N}{2}$ pairs (x, y) to check.

$\Pr(h(x)=h(y)) \leq \frac{1}{M}$ by universality.

$$\Rightarrow \Pr(\text{any collision}) \leq \frac{\binom{N}{2}}{M} \leq \frac{1}{2} \quad (\text{union bound})$$



Recall birthday problem. If 22 people in room, $\frac{1}{2}$ chance of shared birthday, since only 365 days, and $\binom{22}{2}$ pairs. Here we just ensure lots of open slots.

\Rightarrow Using quadratic space, just choose hash functions until we get a perfect one, as each hash function has at least $\frac{1}{2}$ chance of being perfect.

Method 2 (linear space) (2-level scheme)

Idea: First use universal hash function to $M=N$ size array.

Let L_i be the load in array bin i .

Then use 2nd-level universal hash function on each bin mapping to array of size L_i^2 . (Method 1)

Thm. If $h \in \mathcal{H}$ is universal, $\Pr\left[\sum_i (L_i)^2 > 4N\right] < \frac{1}{2}$.

proof.

$$\mathbb{E}\left[\sum_i (L_i)^2\right] = \mathbb{E}\left[\sum_x \sum_y C_{xy}\right]$$

$$C_{xy} = \begin{cases} 1 & \text{if } h(x)=h(y) \\ 0 & \text{else} \end{cases} \quad (\text{even if } x=y)$$

If $L_i=3$ with elements (a, b, c) ,

$$\text{then we get collisions } \begin{pmatrix} aa & ab & ac \\ ab & bb & bc \\ ac & bc & cc \end{pmatrix} = 9$$

$$= N + \sum_x \sum_{y \neq x} \mathbb{E}[C_{xy}]$$

$$\leq N + \frac{N(N-1)}{M} < 2N \quad (\text{since } N=M)$$

By Markov's inequality, $\Pr\left(\sum_i (L_i)^2 > 4N\right) < \frac{1}{2}$.



Thus, we just try $h \in \mathcal{H}$ repeatedly to get one with few enough collisions that squaring the bin loads is still linear.

Set-similarity

Let $A, B \subseteq U$ be two subsets. Let $n = |A \cup B|$.

Then Jaccard Similarity $J = J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ is a measure of set-similarity

MinHash (idealized)

- Let $h_i: U \rightarrow [k]$ be ind. unif. random hash functions, $i \in [k]$
- Let $\delta_i = \begin{cases} 1, & \min_{a \in A} h_i(a) = \min_{b \in B} h_i(b) \\ 0, & \text{else.} \end{cases}$ ↳ using oracle actual random hash functions
- Then output $\hat{J} = \frac{1}{k} \sum_{i=1}^k \delta_i$ as estimate for J .

Claim: If $k > \frac{k_n^2}{\delta}$, and $k > \frac{2}{\epsilon^2 \delta}$, then

$$\text{Prob}(|\hat{J} - J| > \epsilon) < \delta.$$

proof. Recall that if h_i is a universal hash function, all of $h_i(x)$, $x \in A \cup B$ are distinct with probability at least $1 - \frac{\delta}{2k}$. (signatures)

By union bound, all h_i have no collisions w.p. at least $1 - \frac{\delta}{2}$.

Then $h_i(a) = h_i(b)$ only if $a = b$.

$$\Rightarrow \min_{a \in A} h_i(a) = \min_{b \in B} h_i(b) \text{ only if } a = b \in A \cap B.$$

Clearly, the converse also holds, so $\mathbb{E} \delta_i = J$, and $\mathbb{E} \hat{J} = J$, $\text{Var}(\delta_i) \leq J$.

$$\text{Then } \text{Prob}(|\hat{J} - J| \geq \epsilon) \leq \frac{\text{Var}(\hat{J})}{\epsilon^2} \leq \frac{1}{\epsilon^2} \cdot \frac{1}{k} \cdot J \leq \frac{\delta J}{2} \leq \frac{\delta}{2}. \quad \square$$

Thus, we need $O\left(\frac{1}{\epsilon^2 \delta} \log \frac{k_n^2}{\delta}\right) = O\left(\frac{1}{\epsilon^2} \log \frac{n}{\epsilon \delta}\right)$
not tight at all.

As2e: Can do better in idealized setting $O\left(\frac{1}{\epsilon^2} \left(\log \log n + \log \frac{1}{\epsilon}\right)\right)$
[Yu, Weber, TKDE, 2020] for ϵ additive error

As, 2c: We assume hash functions were fully random

Unfortunately, cannot just use 2- or 4-wise hash families, but need a stronger cond., i.e. that any item is equally likely to be the minimum.

A new criterion called **min-wise independence** [Indyk, 1999]

↳ almost achievable for l -wise ind. hash functions for l large enough.