Lecture 4 1/27/22

Perfect Hashing

Goal: Dict. with $O(n)$ space
$O(1)$ query time determ.

Contrast: last time $O(1)$ expected.

Question: what if build a hash table with random h.f with no collisions.

$C_x = \# y \in S \quad y \neq x$ and $h(y) = h(x)$.

$C \triangleq \sum_{x \in S} C_x = \#$ pairwise col. $x, y \in S$.

Suppose we want $C = 0$. $\Rightarrow$ all buckets have size 0 or 1.

$\mathbb{E}[C]$ when we have m buckets.
$\boxed{(h : U \to [m])}$.

SCU
$n = |S|$
$m =$ table size

$$\mathbb{E}[C] = \mathbb{E}\left[\sum_{x \in S} C_x\right] = \sum_{x \in S} \mathbb{E}[C_x]$$

$$\leq \sum_{x \in S} \frac{n}{m}$$

$$= n^2/m.$$

By Markov bnd: $C \le 4\,\mathbb{E}[C]$ with prob $\ge 1 - 1/4$.

$= 4n^2/m$.

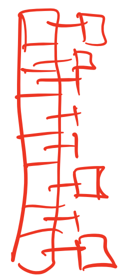
Set $m = 8n^2$ $\Rightarrow$ $C \le 1/2$ with prob $\ge 1 - 1/4$.

$\Rightarrow C = 0$ —— " ——

Algo: repeat until success:

- pick random $h \in \mathcal{H}$; $m = 8n^2$
- compute C
- if $C \ge 1$, repeat.
- if $C = 0$, success.

build hash table H with last h.



Algo has space $O(m+n) = O(n^2)$

q.t. $O(1)$ determ.

Preprocessing: proportional to #repeats:

$$\mathbb{E}[\#\text{repeats}] \le 1 \cdot \frac{3}{4} + 2 \cdot \frac{1}{4} \cdot \frac{3}{4} + 3 \cdot \left(\frac{1}{4}\right)^2 \cdot \frac{3}{4} + \dots$$

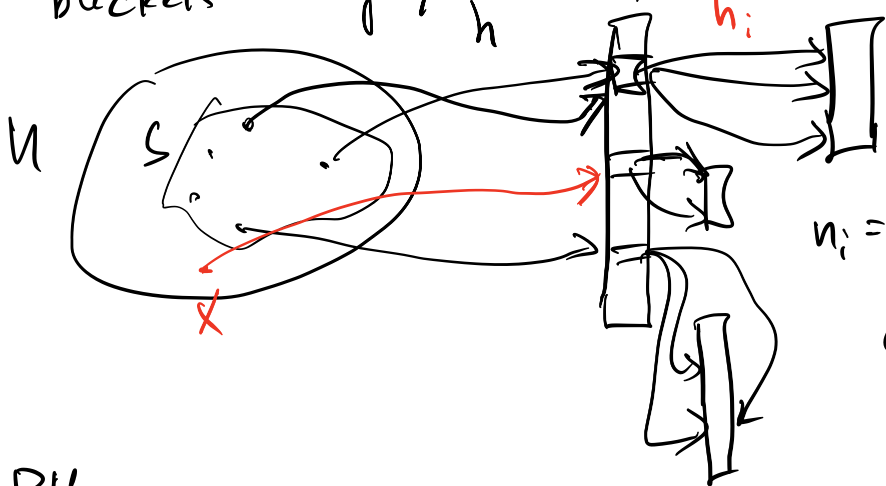
$$= 1 + \frac{1}{4} + \left(\frac{1}{4}\right)^2 + \dots$$

$$= \frac{1}{1 - 1/4} = 4/3.$$

Cor: can obtain $O(n^2)$ space, $O(1)$ det. q.t.

Perfect Hashing

Idea: use standard hashing, $m = 2n$.
+ 2^{nd} level hash where we split non-empty buckets using quadratic-space hash.



$m_i = 8n_i^2$

n_i = # elm's going bucket i.

Algo PH:

Pick a random h.f. $h: U \to [m]$, $m = 2n$.

for $i = 1..m$:

- build 2^{nd} level hash table of size $m_i = 8n_i^2$, (using Cor)
  where $n_i = |S \cap h^{-1}(i)|$ = # elm's in bucket i.
- store $S \cap h^{-1}(i)$ in the 2^{nd} level h.t.

Query time: look-up $h(x)$, then look-up hash table corresponding to $i = h(x)$.

Obs 1: query time is $O(1)$ det.

Question: what about space?

Claim: in expectation, size of 2^{nd} level hash tables is $O(n)$.

$n_i + n_i(n_i-1) = n_i^2$

pf: $\sum_{i=1}^{m} m_i = \sum_{i=1}^{m} 8n_i^2$.

$$\mathbb{E}_h\left[\sum_{i=1}^{m} n_i^2\right] = \mathbb{E}_h\left[\sum_{\substack{i=1 \\ n_i \geq 1}}^{m} n_i + n_i(n_i-1)\right]$$

$$= \mathbb{E}\sum_{i=1}^{m} n_i + \mathbb{E}\left[\sum_{\substack{i=1 \\ n_i \geq 1}}^{m} n_i(n_i-1)\right]$$

$$= n + \mathbb{E}[C]$$

$$= n + n^2/m = n + n/2 = O(n).$$

Conclusion: PH algo obtains $O(1)$ det. q.t.

$O(n)$ space exp.

$O(n)$ preproc. exp.

Remark: can have $O(n)$ det. space bound, by repeating 1^{st} lev hashing until $\sum n_i^2 \leq O(n)$.

Remark: dynamic Dict.: can we obtain $O(1)$ det. query/update time?

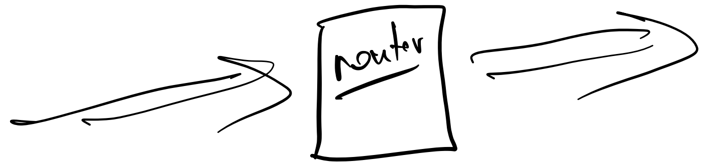
(with $O(n)$ space).

OPEN.

Streaming & Sketching

Mot 2: more eff. read data in a linear fashion.

Mot 1:



Goal: store statistic on data, as streams to be used later.

Distinct elm count.

Stream: elements from universe $[n] = \{1, \dots n\}$.

E.g.: $[n]$ = all possible IPs.

Problem: report how many diff elm's we have seen in the stream.

Stream length: m.

Goal: store as little info as possible

Solution 1: store entire stream: $O(m)$ space.

Sol 2: store a table $T[1..n]$,

$T[i] = 1$ iff we've seen i.

$O(n)$ bits.

Can we obtain space $\ll \min\{n, m\}$?

No unless we allow approx + random.

Algo: approx & random. [Flajolet–Martin]

- pick a random hash func. $h : [n] \to [0,1]$.
- store a reg. z, init $z = 1$.
- when see i in the stream:

$$z = \min\{z, h(i)\}.$$

- Est: $\frac{1}{z} - 1$.

Note: z = min hash function value $h(i)$ among i's in the stream.

Claim: $\mathbb{E}[z] = \frac{1}{d+1}$, d = # distinct elm's.

Pf: z = min of d random #'s $a_1 .. a_d \in [0,1]$.

$a_1 = h(1^{st}\ \text{elm})$

Experiment: pick $a \in [0,1]$.

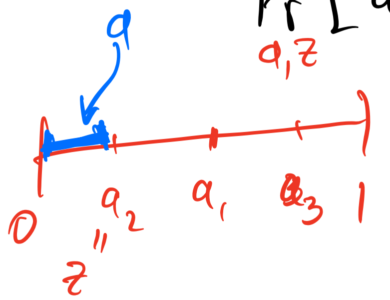
1) z

$1) = \mathbb{E}_z\left[\Pr_a[a < z]\right]$

$\Pr[a < z] =$

2) Prob that a is min $a_1, a_2, \ldots, a_d, a$

$= \frac{1}{d+1}$



$\Rightarrow \mathbb{E}[z] = \frac{1}{d+1}$. $\boxtimes$.

$\mathbb{E}[1/z] \neq 1/\mathbb{E}[z]$.