The turnstile streaming model / Linear Sketching
Sparse recovery

Def: A datastructure $DS(\sigma)$ computed from stream σ is a <u>sketch</u> if there is a combining alg

$$Comb(DS(\sigma_1), DS(\sigma_2)) = DS(\sigma_1 \circ \sigma_2)$$

that can be computed in small space.

A sketching alg Sk is a <u>linear sketch</u> if

maintain $Sk(\sigma) \in$ vectorspace of dim $l = l(n)$

$A \cdot x$ implicitly & $Sk(\sigma)$ is a linear function the frequency vector $(f_1, f_2 \dots f_n)$, where

f_i = # occurences of elt i in σ.

Turnstile token (j, c)

elt $\in [n]$

update for elt j

$$f_j \leftarrow f_j + c$$

$$c \in \mathbb{Z}$$

Streaming s-sparse recovery / detect.

- if σ has $\leq s$ values

  ($\Rightarrow f$ is s.t. $|\mathrm{supp} f| \leq s$)

  output f (we only care abt the s-elements of supp)

- ow. output "not s-sparse"

$$\begin{array}{cccccccccc} & 1 & 2 & 3 & 4 & & & & n \\ f: & 0 & 0 & 2 & 4 & 0 & 0 & 1 & 0 \end{array}$$

freq. vect of stream

$(3, 2)$ $(4, 4)$ $(7, 1)$

pos i count/freq of i $\forall\, i \in \mathrm{supp}(f)$

$O(s \log n)$ space needed.

Eg:

updates $(2, 3)$ $f_2 = 0$

$f_2 = 3$

$(2, -3)$ $f_2 = 3 - 3 = 0$

$(2, 1)$ $f_2 = 1$

$(2, -2)$ $f_3 = -1$

Notation : sparsity $= s = \ell_0(f)$

Space : $\mathrm{poly}(s, \log n)$; $s \ll n$

Special case $s = 1$

$$f = (0 \quad 0 \quad 0 \ldots \lambda \quad 0 0 0)$$

$$f_i = \lambda.$$

Promiss: f is 1-sparse.

Alg: maintain $\ell = \sum f_i$ ← sum of freq.

lin sketches. $z = \sum j f_j$ ← sum of all cts.

if 1-sparsity $\ell = \lambda$, $z = \lambda i$ $\Big\}$ $i = z/\ell$

Output z/ℓ if $\ell \neq 0$

0 ow.

No promiss case:

Use "fingerprint": map [big] -> [small]

Pr[colision] is small

Let Field $|F| = n^3$

Idea: maintain $p(x) = \sum_{i=1}^{n} f_i x^i$ implicitly (freq)

Alg for 1-sparse recovery & detection

- Iniatiaze (ℓ, z, p)

  ℓ: $\sum f_i$, z: $\sum f_i i$, p: fingerprint.

- Pick $r \xleftarrow{uar} F^*$ → $\mathbb{Z}$ mod p prime, $p \approx \Theta(n^3)$

On token (j, c) ($j \in [n]$) (update $f_j \leftarrow f_j + c$)

$\ell = \ell + c$

$z = z + jc$

$j \le n$

poly NOT explicit.

$p = p + c\, r^j$ ← updates just the coeff at r^j

fingerprint

coefficients = frequencies

i.e. $p(r) = \sum f_i r^i$ is maintained implicitly

$p(x) = \sum f_i x^i$

Output:

If $\ell = z = p = 0$ out $f = \bar{0}$

Else if $z/\ell \notin [n]$ or

$$p = \ell r^{z/\ell}$$

then "non-1-sparse"

Eg: $(7, 2)$

$p = 0 + 2r^7$

$(3, -1)$

$p = 2r^7 - r^3$

$(7, -3)$

$p = 2r^7 - r^3 - 3r^7$

$= (2-3)r^7 - r^3$

sequence of updates

Else output $f = \ell e_{z/\ell}$

$\rightarrow (z/\ell, \ell)$ $\quad O(\log n)$ bits.

n is size of domain "$[n]$"

assumption poly(n)

Analysis:

If f is 1-sparse

then ℓ, z, p are in the correct form.

& will not output $\ell = 0$ or $z = 0$

nor $p = 0$

so won't output $f_0 = 0$

Also $z/\ell \in \mathbb{Z}$ ✓.

If f is > 1-sparse

$$\Pr\left[\text{Outputing incorrect "1-sparse" answer}\right]$$

$$= \Pr_{r \leftarrow F}\left[p(r) = \ell r^{z/\ell}\right]$$

$p(x) \neq \ell x^{z/\ell}$ as polynomials

p is at least 2-sparse

$$(0\,0\,1\,1\,0\,0\,0) \neq (0\,0\,0\,1\,0\,0)$$

$$x^3 + x^4 \qquad\qquad \ell x^{r/\ell}$$

$$\Pr_{r \leftarrow F}\left[\left(p - \ell x^{z/\ell}\right)(r) = 0\right]$$

poly of deg $\leq n$.

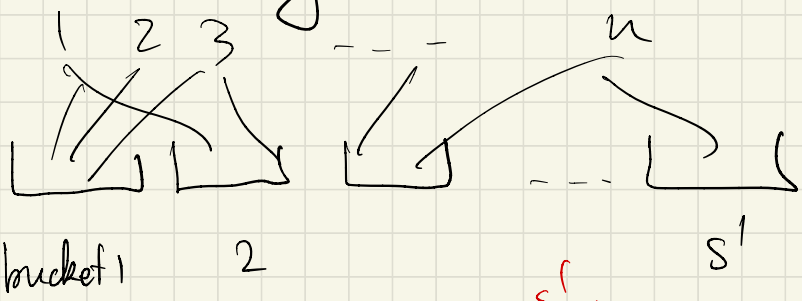
$$\leq \frac{n}{|F|} = \frac{1}{n^2}$$

✓

General s-sparse recovery & detection

2 10 7 2 n ---- j

Use hashing $h: [n] \to [s']$



h is chosen u.a.r from a 2-universal family

$s' \ll n$

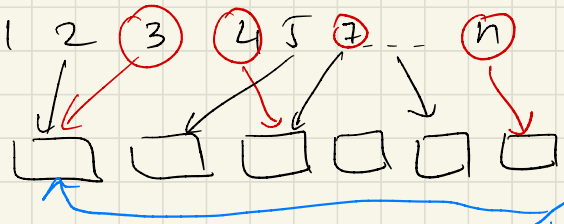
We'll use a stream per bucket (in parallel)

Point: If stream s-sparse then two distinct elements of supp(f) are hashed into different buckets

So we'll be using Alg for 1-sparse recovery.

Eg: $\mathrm{Supp}(f) = \{3, 4, 7, n\}$

$h : [n] \to [6]$ from 2-universal family



We'd like that a fixed elt of supp is hashed in a diff bucket than everyone else in supp

Use 2-universal hash family

$H = \{ h : [n] \to [2s] \}$ is 2-universal

if $\forall\, x \neq y \in [n]$

$$\Pr_{h \leftarrow H}[h(x) = h(y)] \leq \frac{1}{2s}$$

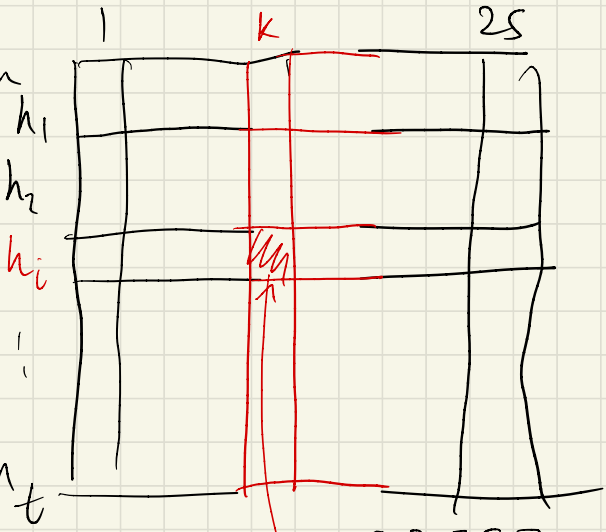
Pick Eg A uar matrix
b uar vect
$\{f_{A,B}(x) = Ax + b\}_{A,b}$ is 2-univ.

$f_{ab}(x) = ax + b$
$a, b \xleftarrow{uar} \mathbb{F}_2^n$
$g_{ab}(x) =$ last k bits of $f_{ab}(x)$
$\{g_{ab}\}$ is 2-univ

Goal: Run many 1-sparse recovery instances in parallel. (2s per hash fnct)

Alg for s-sparse recovery:

Maintain



$t = O(\log s/\delta)$

hash independent function

$D[i][k]$

Stream for bucket k on hash h_i

↳ waiting for some elt $x \in supp(f)$ s.t. $h_i(x) = k$

$|h_i^{-1}(h_i(x)) \cap supp| = 1$

Init $D[t][2s] = 0$

On token (j, c)

Look at where j is mapped to by each h_i?

For each hash i update Δs at $D[i][h_i(j)]$

this stream should take token $(h_i(j), c)$ in 1-sparse subroutine

(as if $D[i][h_i(j)]$ datastructure is for 1-sparse streams)

Output:

For $\forall i$ $\forall k$

if $D[i][k]$ outpts 1-sparse & (i, λ_i) then

record it in $A[i] = \lambda_i$ (overwrite if needed)

If $|\text{supp } A| > s$ output "not s-sparse"

O.W. output $f = \sum_{i \in \text{Supp}(A)} A[i] e_i$

<u>Analysis</u>:

Alg is correct if: ① $\forall i \in \text{Supp}(f)$ $\exists$ hash that hashes i in its own bucket (without other supp elts)

② no other bucket that comes up positive is a false positive.

produces an 1-sparse output

For fixed $i \in Supp(f)$, for the j th hash func.

$$\Pr_{h_j}\left[h_j(i) = h_j(k) \text{ for some other } k \in Supp(f),\ k \neq i \right] \leq \sum_{\substack{k \neq i \\ k \in Supp}} \Pr\left[h_j(i) = h_j(k)\right]$$

$$\leq \frac{s-1}{2s} < \frac{1}{2}$$

by 2-univ. of Hash family.

For fixed i

$$\Pr\left[\text{all } t \text{ of the hash fncs cause some colision in the supp}\right]$$

$$\leq \left(\frac{1}{2}\right)^t \leq \delta/s \quad \text{for } t = \log s/\delta$$

So by union bound

$$\Pr\left[\text{Some } i \text{ in the supp is failed by all hashes}\right]$$

$$\leq s \cdot \delta/s = \delta$$

So every $i \in \mathrm{Supp}(f)$ is split by some hash j w.p. $1-\delta$.

$\uparrow$ So we succeed in getting the correct 1-sparse streams w.p. $1-\delta$.

$\Pr[$ Some non 1-sparse stream outputs a false positive 1-sparse answer $(i, \lambda_i)]$

by union bound

$$\leq \underbrace{2s \cdot t}_{\text{many 1-sparse instances / streams}} \cdot \frac{1}{n^2} = o(1)$$

So we get correct answer w.p. $1-\delta-o(1)$

Thm: Can do s-sparse recovery in the turnstile model wp $1-\delta'$ using

$$O(s \log \tfrac{s}{\delta} \log n) \text{ bits of space}$$

Can be viewed as maintaining

$$A \cdot x$$

random matrix

freq.

1-sparse rec.: $(1, r, r^2, \dots r^n) \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{pmatrix}$

$$= \sum f_i r^i$$

s-sparse: take multiple random r's.

$$\begin{pmatrix} r_1 & r_1^2 & \cdots & r_1^n \\ r_2 & r_2^2 & \cdots & r_2^n \\ \vdots & & & \\ r_{2st} & r_{2st}^2 & \cdots & r_{2st}^n \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_n \end{pmatrix} = \begin{pmatrix} \sum f_i r_1^i \\ \sum f_i r_2^i \\ \vdots \\ \sum f_i r_{2st}^n \end{pmatrix}$$

implicitly maintaining a Vandermonde matrix (random)

linear sketch: $V \cdot f$ is maintained implicitly in small space.

One can use same ideas as above to get "ℓ_0 sampling"

ℓ_0-sampling: sample a unif. rand elt in the supp(f)

$\forall i \in \text{supp} \qquad \Pr[\text{output} = i] = \frac{1}{|\text{supp}(f)|}$

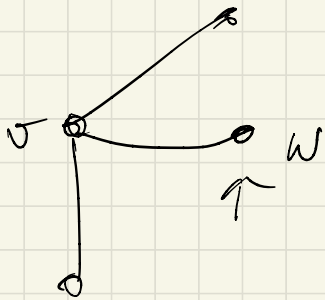
Thm: Jayaram & Woodruff.

Can do perfect ℓ_0-sampling in small space using linear sketch.

$$\Pr_A[\text{sketch } A \text{ gives correct answer}] \geq 1 - \delta$$

$\uparrow$ $1/\text{poly}\, n$

Eg: Sample a unif rand neighbor of a vtx v



adj vector of v : 0 0 1 0 1 0 1 1 0 0

ℓ_0 sampling : output unif rand elt of supp adj(v)

We'll use ℓ_0-sampling as a subroutine in graph sketching for connectivity & bipartiteness testing.