Topics in TCS

Frequency estimation via sketching

Raphaël Clifford

Frequent items via sketching

We return to the problem of finding frequent items. Our previous definition was: Given a parameter k, find the set of symbols with frequency greater than m/k.

Frequent items via sketching

We return to the problem of finding frequent items. Our previous definition was: Given a parameter k, find the set of symbols with frequency greater than m/k.

The Misra-Gries algorithm is one-pass and runs in $O(k(\log m + \log n))$ bits of space and $O(m \log n)$ time.

Frequent items via sketching

We return to the problem of finding frequent items. Our previous definition was: Given a parameter k, find the set of symbols with frequency greater than m/k.

The MISRA-GRIES algorithm is one-pass and runs in $O(k(\log m + \log n))$ bits of space and $O(m \log n)$ time.

It is deterministic (good✓) but only works in the cash register model.

Frequent items via sketching

We return to the problem of finding frequent items. Our previous definition was: Given a parameter k, find the set of symbols with frequency greater than m/k.

The MISRA-GRIES algorithm is one-pass and runs in $O(k(\log m + \log n))$ bits of space and $O(m \log n)$ time.

It is deterministic (good✓) but only works in the cash register model.

We will change the definition to ask for an estimate of the frequency of occurrence for any token queried.

Frequent items via sketching

We return to the problem of finding frequent items. Our previous definition was: Given a parameter k, find the set of symbols with frequency greater than m/k.

The MISRA-GRIES algorithm is one-pass and runs in $O(k(\log m + \log n))$ bits of space and $O(m \log n)$ time.

It is deterministic (good✓) but only works in the cash register model.

We will change the definition to ask for an estimate of the frequency of occurrence for any token queried.

We will introduce our first randomised *sketching* algorithms.

Frequent items via sketching

We return to the problem of finding frequent items. Our previous definition was: Given a parameter k, find the set of symbols with frequency greater than m/k.

The Misra-Gries algorithm is one-pass and runs in $O(k(\log m + \log n))$ bits of space and $O(m \log n)$ time.

It is deterministic (good✓) but only works in the cash register model.

We will change the definition to ask for an estimate of the frequency of occurrence for any token queried.

We will introduce our first randomised *sketching* algorithms.

Our sketches will be *linear* which will mean we can extend them to the *turnstile model*. We can also combine them easily by adding.

Frequent items via sketching

We return to the problem of finding frequent items. Our previous definition was: Given a parameter k, find the set of symbols with frequency greater than m/k.

The MISRA-GRIES algorithm is one-pass and runs in $O(k(\log m + \log n))$ bits of space and $O(m \log n)$ time.

It is deterministic (good✓) but only works in the cash register model.

We will change the definition to ask for an estimate of the frequency of occurrence for any token queried.

We will introduce our first randomised *sketching* algorithms.

Our sketches will be *linear* which will mean we can extend them to the *turnstile model*. We can also combine them easily by adding.

They will give us an estimate of the frequency for *every* token.

CountSketch

The sketch is a 2D-array C with t rows and k columns. All hash functions are chosen from a pairwise independent family.

CountSketch

The sketch is a 2D-array C with t rows and k columns. All hash functions are chosen from a pairwise independent family.

```
stream ⟨a_1, ..., a_m⟩, a_i ∈ [n]
initialise C[1...t][1...k] = 0
choose hash functions h_1, ... h_t : [n] → [k]
choose hash function g_1, ..., g_t : [n] → {−1, 1}

CountSketch(a_i)
for each j ∈ [t]
      C[j, h_j(a_i)] += c_i g_j(a_i)

return f̂_{a_i} = median{g_j(a_i) C[j, h_j(a_i)]}
```

c_i is the number of instances of a_i. In the turnstile model this can be either positive of negative.

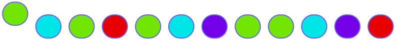
CountSketch - worked example

	1	2	3
h_1			
h_2			

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
      C[j, h_j(a_i)] += c_i g_j(a_i)
```

CountSketch - worked example

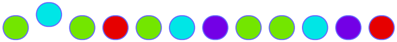


	1	2	3
h_1		+	
h_2	+		

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i g_j(a_i)
```

CountSketch - worked example

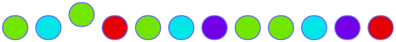


	1	2	3
h_1		$+$	$-$
h_2	$+$	$+$	

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
      C[j, h_j(a_i)] += c_i g_j(a_i)
```

CountSketch - worked example

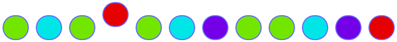


	1	2	3
h_1		$++$	$-$
h_2	$++$	$+$	

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

CountSketch(a_i)
`for each` $j \in [t]$
$\quad C[j, h_j(a_i)] \mathrel{+}= c_i g_j(a_i)$

CountSketch - worked example

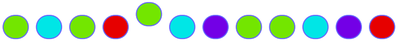


	1	**2**	**3**
h_1		$++-$	$-$
h_2	$++$	$+$	$+$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i g_j(a_i)
```

COUNTSKETCH - worked example

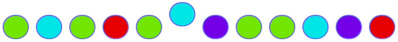


	1	2	3
h_1		$++-+$	$-$
h_2	$+++$	$+$	$+$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

COUNTSKETCH(a_i)
for each $j \in [t]$
$\quad C[j, h_j(a_i)] \mathrel{+}= c_i g_j(a_i)$

CountSketch - worked example

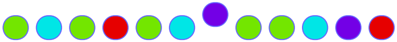


	1	2	3
h_1		$++-+$	$--$
h_2	$+++$	$++$	$+$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
      C[j, h_j(a_i)] += c_i g_j(a_i)
```

CountSketch - worked example

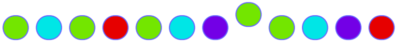


	1	2	3
h_1	$+$	$++-+$	$--$
h_2	$+++$	$++$	$+-$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
      C[j, h_j(a_i)] += c_i g_j(a_i)
```

CountSketch - worked example

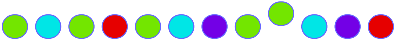


	1	2	3
h_1	$+$	$++-++$	$--$
h_2	$++++$	$++$	$+-$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i g_j(a_i)
```

CountSketch - worked example

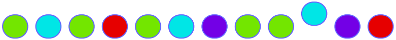


	1	**2**	**3**
h_1	$+$	$++-++$ $+$	$--$
h_2	$+++++$	$++$	$+-$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i g_j(a_i)
```

CountSketch - worked example

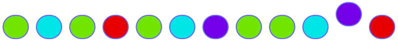


	1	2	3
h_1	$+$	$++-++$ $+$	$---$
h_2	$+++++$	$+++$	$+-$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

```
CountSketch(a_i)
for each j ∈ [t]
      C[j, h_j(a_i)] += c_i g_j(a_i)
```

CountSketch - worked example

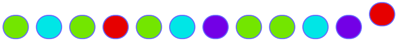


	1	2	3
h_1	$++$	$++-++$ $+$	$---$
h_2	$+++++$	$+++$	$+--$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

CountSketch(a_i)
for each $j \in [t]$
    $C[j, h_j(a_i)] \mathrel{+}= c_i g_j(a_i)$

CountSketch - worked example



	1	2	3
h_1	$++$	$++-++$ $+-$	$---$
h_2	$+++++$	$+++$	$+--+$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

CountSketch(a_i)
for each $j \in [t]$
    $C[j, h_j(a_i)] \mathrel{+}= c_i g_j(a_i)$

CountSketch - worked example

	1	2	3
h_1	$++$	$++-++$ $+-$	$---$
h_2	$+++++$	$+++$	$+--+$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

return $\hat{f}_{a_i} =$ median$\{g_j(a_i)C[j, h_j(a_i)]\}$

CountSketch - worked example

	1	2	3
h_1	$++$	$++-++$ $+-$	$---$
h_2	$+++++$	$+++$	$+--+$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

`return` $\hat{f}_{a_i} =$ `median`$\{g_j(a_i) C[j, h_j(a_i)]\}$

$\hat{f} = \text{median}(g_1()C[1, h_1()], g_2()C[2, h_2()]) = \text{median}(1 \cdot 3, 1 \cdot 5)$

CountSketch - worked example

	1	2	3
h_1	$++$	$++-++$ $+-$	$---$
h_2	$+++++$	$+++$	$+--+$

	h_1, g_1	h_2, g_2
$\bigcirc$	$2, +$	$1, +$
$\bigcirc$	$3, -$	$2, +$
$\bigcirc$	$1, +$	$3, -$
$\bigcirc$	$2, -$	$3, +$

$$\texttt{return } \hat{f}_{a_i} = \texttt{median}\{g_j(a_i)C[j, h_j(a_i)]\}$$

$$\hat{f}_{\bigcirc} = \text{median}(g_1(\bigcirc)C[1, h_1(\bigcirc)], g_2(\bigcirc)C[2, h_2(\bigcirc)]) = \text{median}(1 \cdot 3, 1 \cdot 5)$$

$$\hat{f}_{\bigcirc} = \text{median}(g_1(\bigcirc)C[1, h_1(\bigcirc)], g_2(\bigcirc)C[2, h_2(\bigcirc)]) = \text{median}(-1 \cdot -3, 1 \cdot 3)$$

CountSketch - worked example

	1	2	3
h_1	$++$	$++-++$ $+-$	$---$
h_2	$+++++$	$+++$	$+--+$

	h_1, g_1	h_2, g_2
	$2, +$	$1, +$
	$3, -$	$2, +$
	$1, +$	$3, -$
	$2, -$	$3, +$

$$\boxed{\texttt{return}\ \hat{f}_{a_i} = \texttt{median}\{g_j(a_i)C[j, h_j(a_i)]\}}$$

$$\hat{f} = \text{median}(g_1()C[1, h_1()], g_2()C[2, h_2()]) = \text{median}(1 \cdot 3, 1 \cdot 5)$$

$$\hat{f} = \text{median}(g_1()C[1, h_1()], g_2()C[2, h_2()]) = \text{median}(-1 \cdot -3, 1 \cdot 3)$$

$$\hat{f} = \text{median}(g_1()C[1, h_1()], g_2()C[2, h_2()]) = \text{median}(1 \cdot 2, -1 \cdot 0)$$

CountSketch - worked example

	1	2	3
h_1	$++$	$++-++$ $+-$	$---$
h_2	$+++++$	$+++$	$+--+$

	h_1, g_1	h_2, g_2
$\bullet$	$2, +$	$1, +$
$\bullet$	$3, -$	$2, +$
$\bullet$	$1, +$	$3, -$
$\bullet$	$2, -$	$3, +$

`return` $\hat{f}_{a_i} =$ `median`$\{g_j(a_i)C[j, h_j(a_i)]\}$

$$\hat{f}_{\bullet} = \text{median}(g_1(\bullet)C[1, h_1(\bullet)], g_2(\bullet)C[2, h_2(\bullet)]) = \text{median}(1 \cdot 3, 1 \cdot 5)$$

$$\hat{f}_{\bullet} = \text{median}(g_1(\bullet)C[1, h_1(\bullet)], g_2(\bullet)C[2, h_2(\bullet)]) = \text{median}(-1 \cdot -3, 1 \cdot 3)$$

$$\hat{f}_{\bullet} = \text{median}(g_1(\bullet)C[1, h_1(\bullet)], g_2(\bullet)C[2, h_2(\bullet)]) = \text{median}(1 \cdot 2, -1 \cdot 0)$$

$$\hat{f}_{\bullet} = \text{median}(g_1(\bullet)C[1, h_1(\bullet)], g_2(\bullet)C[2, h_2(\bullet)]) = \text{median}(-1 \cdot 3, 1 \cdot 0)$$

CountSketch - Analysis I

To start, let us look just at an arbitrary row of C. We will show that for each row CountSketch gives an unbiased estimate. Define $C[x] = C[1, x]$.

CountSketch - Analysis I

To start, let us look just at an arbitrary row of C. We will show that for each row CountSketch gives an unbiased estimate. Define $C[x] = C[1, x]$.

Let $X = \hat{f}_a$ be the output for query a.

CountSketch - Analysis I

To start, let us look just at an arbitrary row of C. We will show that for each row CountSketch gives an unbiased estimate. Define $C[x] = C[1, x]$.

Let $X = \hat{f}_a$ be the output for query a.

For each token j, define indicator r.v. $Y_j = 1$ iff $h(j) = h(a)$.

CountSketch - Analysis I

To start, let us look just at an arbitrary row of C. We will show that for each row CountSketch gives an unbiased estimate. Define $C[x] = C[1, x]$.

Let $X = \hat{f}_a$ be the output for query a.

For each token j, define indicator r.v. $Y_j = 1$ iff $h(j) = h(a)$.

Token j contributes $f_j \cdot g(j)$ to $C[h(a)]$ iff $h(j) = h(a)$.

CountSketch - Analysis I

To start, let us look just at an arbitrary row of C. We will show that for each row CountSketch gives an unbiased estimate. Define $C[x] = C[1, x]$.

Let $X = \hat{f}_a$ be the output for query a.

For each token j, define indicator r.v. $Y_j = 1$ iff $h(j) = h(a)$.

Token j contributes $f_j \cdot g(j)$ to $C[h(a)]$ iff $h(j) = h(a)$.

Therefore

$$X = g(a) \sum_{j=1}^{n} f_j g(j) Y_j = f_a + \sum_{j \in [n] \setminus \{a\}} f_j g(a) g(j) Y_j$$

CountSketch - Analysis I

To start, let us look just at an arbitrary row of C. We will show that for each row CountSketch gives an unbiased estimate. Define $C[x] = C[1, x]$.

Let $X = \hat{f}_a$ be the output for query a.

For each token j, define indicator r.v. $Y_j = 1$ iff $h(j) = h(a)$.

Token j contributes $f_j \cdot g(j)$ to $C[h(a)]$ iff $h(j) = h(a)$.

Therefore

$$X = g(a)\sum_{j=1}^{n} f_j g(j) Y_j = f_a + \sum_{j \in [n]\setminus\{a\}} f_j g(a) g(j) Y_j$$

As g and h are independent and g is from a pairwise independent family,

$$\mathbb{E}[g(a)g(j)Y_j] = \mathbb{E}(g(a)) \cdot \mathbb{E}(g(j)) \cdot \mathbb{E}(Y_j) = 0 \cdot 0 \cdot \mathbb{E}(Y_j) = 0$$

CountSketch - Analysis I

To start, let us look just at an arbitrary row of C. We will show that for each row CountSketch gives an unbiased estimate. Define $C[x] = C[1, x]$.

Let $X = \hat{f}_a$ be the output for query a.

For each token j, define indicator r.v. $Y_j = 1$ iff $h(j) = h(a)$.

Token j contributes $f_j \cdot g(j)$ to $C[h(a)]$ iff $h(j) = h(a)$.

Therefore

$$X = g(a) \sum_{j=1}^{n} f_j g(j) Y_j = f_a + \sum_{j \in [n] \setminus \{a\}} f_j g(a) g(j) Y_j$$

As g and h are independent and g is from a pairwise independent family,

$$\mathbb{E}[g(a) g(j) Y_j] = \mathbb{E}(g(a)) \cdot \mathbb{E}(g(j)) \cdot \mathbb{E}(Y_j) = 0 \cdot 0 \cdot \mathbb{E}(Y_j) = 0$$

By linearity of expectation

$$\mathbb{E}(X) = f_a + \sum_{j \in [n] \setminus \{a\}} f_j \mathbb{E}[g(a) g(j) Y_j] = f_a$$

CountSketch - Analysis IIa

We will now derive the variance of our estimator $X = \hat{f}$. Recall $Y_j = 1$ iff $h(j) = h(a)$.

CountSketch - Analysis IIa

We will now derive the variance of our estimator $X = \hat{f}$. Recall $Y_j = 1$ iff $h(j) = h(a)$.

$$\mathrm{var}(X) = 0 + \mathrm{var}\left[g(a) \sum_{j \in [n] \setminus \{a\}} f_j \cdot g(j) Y_j \right]$$

CountSketch - Analysis IIa

We will now derive the variance of our estimator $X = \hat{f}$. Recall $Y_j = 1$ iff $h(j) = h(a)$.

$$\begin{aligned}
\operatorname{var}(X) &= 0 + \operatorname{var}\left[g(a)\sum_{j\in[n]\setminus\{a\}} f_j \cdot g(j) Y_j\right] \\
&= \mathbb{E}\left[g(a)^2 \sum_{j\in[n]\setminus\{a\}} f_j^2 Y_j^2 + \sum_{\substack{j\in[n]\setminus\{a\} \\ i\neq j}} f_i f_j g(i) g(j) Y_i Y_j\right] - \\
&\qquad \left[\sum_{j\in[n]\setminus\{a\}} f_j \mathbb{E}[g(a) g(j) Y_j]\right]^2
\end{aligned}$$

CountSketch - Analysis IIa

We will now derive the variance of our estimator $X = \hat{f}$. Recall $Y_j = 1$ iff $h(j) = h(a)$.

$$\begin{aligned}
\mathrm{var}(X) &= 0 + \mathrm{var}\left[g(a) \sum_{j \in [n] \setminus \{a\}} f_j \cdot g(j) Y_j\right] \\
&= \mathbb{E}\left[g(a)^2 \sum_{j \in [n] \setminus \{a\}} f_j^2 Y_j^2 + \sum_{\substack{j \in [n] \setminus \{a\} \\ i \neq j}} f_i f_j g(i) g(j) Y_i Y_j\right] - \\
&\qquad \left[\sum_{j \in [n] \setminus \{a\}} f_j \mathbb{E}[g(a) g(j) Y_j]\right]^2
\end{aligned}$$

We will need two facts to simplify these terms.

CountSketch - Analysis IIb

$$\mathrm{var}(X) = \mathbb{E}\left[g(a)^2 \sum_{j\in[n]\setminus\{a\}} f_j^2 Y_j^2 + \sum_{\substack{j\in[n]\setminus\{a\} \\ i\neq j}} f_i f_j g(i) g(j) Y_i Y_j\right] - \left[\sum_{j\in[n]\setminus\{a\}} f_j \mathbb{E}[g(a)g(j)Y_j]\right]^2$$

CountSketch - Analysis IIb

$$\mathrm{var}(X) = \mathbb{E}\left[g(a)^2 \sum_{j\in[n]\setminus\{a\}} f_j^2 Y_j^2 + \sum_{\substack{j\in[n]\setminus\{a\}\\ i\neq j}} f_i f_j g(i) g(j) Y_i Y_j\right] - \left[\sum_{j\in[n]\setminus\{a\}} f_j \mathbb{E}[g(a)g(j)Y_j]\right]^2$$

Now, the two facts:

1. $\mathbb{E}(Y_j^2) = \mathbb{E}(Y_j) = \Pr(h(j) = h(a)) = \frac{1}{k}$.
2. $\mathbb{E}(g(i)g(j)Y_iY_j) = \mathbb{E}(g(i)) \cdot \mathbb{E}(g(j)) \cdot \mathbb{E}(Y_iY_j) = 0 \cdot 0 \cdot \mathbb{E}(Y_iY_j) = 0$

CountSketch - Analysis IIb

$$\mathrm{var}(X) = \mathbb{E}\left[g(a)^2 \sum_{j\in[n]\setminus\{a\}} f_j^2 Y_j^2 + \sum_{\substack{j\in[n]\setminus\{a\}\\ i\neq j}} f_i f_j g(i) g(j) Y_i Y_j\right] - \left[\sum_{j\in[n]\setminus\{a\}} f_j \mathbb{E}[g(a)g(j)Y_j]\right]^2$$

Now, the two facts:

1. $\mathbb{E}(Y_j^2) = \mathbb{E}(Y_j) = \Pr(h(j) = h(a)) = \frac{1}{k}$.
2. $\mathbb{E}(g(i)g(j)Y_iY_j) = \mathbb{E}(g(i)) \cdot \mathbb{E}(g(j)) \cdot \mathbb{E}(Y_iY_j) = 0 \cdot 0 \cdot \mathbb{E}(Y_iY_j) = 0$

Therefore,

$$\mathrm{var}(X) = \sum_{j\in[n]\setminus\{a\}} \frac{f_j^2}{k} + 0 - 0$$

CountSketch - Analysis IIb

$$\mathrm{var}(X) = \mathbb{E}\left[g(a)^2 \sum_{j \in [n] \setminus \{a\}} f_j^2 Y_j^2 + \sum_{\substack{j \in [n] \setminus \{a\} \\ i \neq j}} f_i f_j g(i) g(j) Y_i Y_j \right] - \left[\sum_{j \in [n] \setminus \{a\}} f_j \mathbb{E}[g(a) g(j) Y_j] \right]^2$$

Now, the two facts:

1. $\mathbb{E}(Y_j^2) = \mathbb{E}(Y_j) = \Pr(h(j) = h(a)) = \frac{1}{k}$.
2. $\mathbb{E}(g(i)g(j)Y_iY_j) = \mathbb{E}(g(i)) \cdot \mathbb{E}(g(j)) \cdot \mathbb{E}(Y_iY_j) = 0 \cdot 0 \cdot \mathbb{E}(Y_iY_j) = 0$

Therefore,

$$\mathrm{var}(X) = \sum_{j \in [n] \setminus \{a\}} \frac{f_j^2}{k} + 0 - 0$$

$$= \frac{\|\boldsymbol{f}\|_2^2 - f_a^2}{k} \quad \text{where } \boldsymbol{f} \text{ is the array of frequencies}$$

CountSketch - Analysis III

Using the variance $\text{var}(X) = \frac{\|\boldsymbol{f}\|_2^2 - f_a^2}{k}$ we can apply Chebyshev.

CountSketch - Analysis III

Using the variance $\mathrm{var}(X) = \frac{\|\boldsymbol{f}\|_2^2 - f_a^2}{k}$ we can apply Chebyshev.

$$
\begin{aligned}
\Pr(|\hat{f}_a - f_a| \geq \epsilon\sqrt{\|\boldsymbol{f}\|_2^2 - f_a^2}) &= \Pr(|X - \mathbb{E}(X)| \geq \epsilon\sqrt{\|\boldsymbol{f}\|_2^2 - f_a^2}) \\
&\leq \frac{\mathrm{var}(X)}{\epsilon^2(\|\boldsymbol{f}\|_2^2 - f_a^2)} \\
&= \frac{1}{k\epsilon^2} \\
&= \frac{1}{3} \qquad \text{(set } k = 3/\epsilon^2\text{)}
\end{aligned}
$$

CountSketch - Analysis III

Using the variance $\mathrm{var}(X) = \frac{\|\boldsymbol{f}\|_2^2 - f_a^2}{k}$ we can apply Chebyshev.

$$\begin{aligned}
\Pr\left(|\hat{f}_a - f_a| \geq \epsilon\sqrt{\|\boldsymbol{f}\|_2^2 - f_a^2}\right) &= \Pr\left(|X - \mathbb{E}(X)| \geq \epsilon\sqrt{\|\boldsymbol{f}\|_2^2 - f_a^2}\right) \\
&\leq \frac{\mathrm{var}(X)}{\epsilon^2(\|\boldsymbol{f}\|_2^2 - f_a^2)} \\
&= \frac{1}{k\epsilon^2} \\
&= \frac{1}{3} \qquad \text{(set } k = 3/\epsilon^2\text{)}
\end{aligned}$$

Using the notation $\boldsymbol{f}_{-j}$ for $\boldsymbol{f}$ with the jth element dropped, $\|\boldsymbol{f}_{-j}\|_2^2 = \|\boldsymbol{f}\|_2^2 - f_j^2$.

CountSketch - Analysis III

Using the variance $\mathrm{var}(X) = \frac{\|\boldsymbol{f}\|_2^2 - f_a^2}{k}$ we can apply Chebyshev.

$$\begin{aligned}
\Pr(|\hat{f}_a - f_a| \geq \epsilon\sqrt{\|\boldsymbol{f}\|_2^2 - f_a^2}) &= \Pr(|X - \mathbb{E}(X)| \geq \epsilon\sqrt{\|\boldsymbol{f}\|_2^2 - f_a^2}) \\
&\leq \frac{\mathrm{var}(X)}{\epsilon^2(\|\boldsymbol{f}\|_2^2 - f_a^2)} \\
&= \frac{1}{k\epsilon^2} \\
&= \frac{1}{3} \qquad \text{(set } k = 3/\epsilon^2\text{)}
\end{aligned}$$

Using the notation $\boldsymbol{f}_{-j}$ for $\boldsymbol{f}$ with the jth element dropped, $\|\boldsymbol{f}_{-j}\|_2^2 = \|\boldsymbol{f}\|_2^2 - f_j^2$. And so,

$$\Pr(|\hat{f}_a - f_a| \geq \epsilon\,\|\boldsymbol{f}_{-a}\|_2) \leq \frac{1}{3}$$

CountSketch - Analysis IV

So how good is our sketch that takes the median?

CountSketch - Analysis IV

So how good is our sketch that takes the median?

We take the median of $|\hat{f}_a - f_a|$ for t different independent runs. If this is at least $\epsilon \|\boldsymbol{f}_{-a}\|_2$ then at least $t/2$ iterations are that big.

CountSketch - Analysis IV

So how good is our sketch that takes the median?

We take the median of $|\hat{f}_a - f_a|$ for t different independent runs. If this is at least $\epsilon \|\boldsymbol{f}_{-a}\|_2$ then at least $t/2$ iterations are that big.

We show that this is exponentially unlikely to happen as a function of the number of iterations, t.

CountSketch - Analysis IV

So how good is our sketch that takes the median?

We take the median of $|\hat{f}_a - f_a|$ for t different independent runs. If this is at least $\epsilon \|\boldsymbol{f}_{-a}\|_2$ then at least $t/2$ iterations are that big.

We show that this is exponentially unlikely to happen as a function of the number of iterations, t.

For the ith iteration, let $Z_i = 1$ if $|\hat{f}_a - f_a| \geq \epsilon \|\boldsymbol{f}_{-a}\|_2$ and 0 otherwise.

CountSketch - Analysis IV

So how good is our sketch that takes the median?

We take the median of $|\hat{f}_a - f_a|$ for t different independent runs. If this is at least $\epsilon \left\| \boldsymbol{f}_{-a} \right\|_2$ then at least $t/2$ iterations are that big.

We show that this is exponentially unlikely to happen as a function of the number of iterations, t.

For the ith iteration, let $Z_i = 1$ if $|\hat{f}_a - f_a| \geq \epsilon \left\| \boldsymbol{f}_{-a} \right\|_2$ and 0 otherwise.

Using Chernoff's bound with $\mu = t/3$

$$\Pr\left(\sum_{i=1}^{t} Z_i \geq (1+\delta)\mu\right) \leq \exp(-\delta^2\mu/3) \qquad = \exp(-\delta^2 t/9)$$

$$\Pr\left(\sum_{i=1}^{t} Z_i \geq (1+1/2)\mu\right) \leq \exp(-(1/2)^2 t/9) \qquad = \exp(-t/36)$$

CountSketch - Analysis IV

So how good is our sketch that takes the median?

We take the median of $|\hat{f}_a - f_a|$ for t different independent runs. If this is at least $\epsilon \left\| \boldsymbol{f}_{-a} \right\|_2$ then at least $t/2$ iterations are that big.

We show that this is exponentially unlikely to happen as a function of the number of iterations, t.

For the ith iteration, let $Z_i = 1$ if $|\hat{f}_a - f_a| \geq \epsilon \left\| \boldsymbol{f}_{-a} \right\|_2$ and 0 otherwise.

Using Chernoff's bound with $\mu = t/3$

$$\Pr\left(\sum_{i=1}^{t} Z_i \geq (1+\delta)\mu\right) \leq \exp(-\delta^2\mu/3) \qquad = \exp(-\delta^2 t/9)$$

$$\Pr\left(\sum_{i=1}^{t} Z_i \geq (1+1/2)\mu\right) \leq \exp(-(1/2)^2 t/9) \qquad = \exp(-t/36)$$

For an arbitrary token a, the probability of being further than $\epsilon \left\| \boldsymbol{f}_{-a} \right\|_2$ from the correct frequency is at most $\exp(-t/36)$.

CountSketch - Space/Time

We need $O(\log m)$ bits per counter in our sketch. There are tk counters.

CountSketch - Space/Time

We need $O(\log m)$ bits per counter in our sketch. There are tk counters.

We store t pairwise independent hash functions making $O(t \log n)$ bits.

CountSketch - Space/Time

We need $O(\log m)$ bits per counter in our sketch. There are tk counters.

We store t pairwise independent hash functions making $O(t \log n)$ bits.

Overall space is therefore $O(t \log n + tk \log m)$ bits.

CountSketch - Space/Time

We need $O(\log m)$ bits per counter in our sketch. There are tk counters.

We store t pairwise independent hash functions making $O(t \log n)$ bits.

Overall space is therefore $O(t \log n + tk \log m)$ bits.

With $k = \lceil 3/\epsilon^2 \rceil$ and $t = \lceil \ln 1/\delta \rceil$, this equals

$$O\left(\frac{1}{\epsilon^2} \log \frac{1}{\delta} \cdot (\log m + \log n)\right) \text{ bits}$$

CountSketch - Space/Time

We need $O(\log m)$ bits per counter in our sketch. There are tk counters.

We store t pairwise independent hash functions making $O(t \log n)$ bits.

Overall space is therefore $O(t \log n + tk \log m)$ bits.

With $k = \lceil 3/\epsilon^2 \rceil$ and $t = \lceil \ln 1/\delta \rceil$, this equals

$$O\left(\frac{1}{\epsilon^2} \log \frac{1}{\delta} \cdot (\log m + \log n)\right) \text{ bits}$$

Running time: one-pass and $O(t)$ time per token.

CountSketch summary

CountSketch is a one-pass randomised algorithm to estimate the frequency of the tokens in a stream.

CountSketch summary

CountSketch is a one-pass randomised algorithm to estimate the frequency of the tokens in a stream.

Once ϵ and δ are decided we can set t and k accordingly.

CountSketch summary

CountSketch is a one-pass randomised algorithm to estimate the frequency of the tokens in a stream.

Once ϵ and δ are decided we can set t and k accordingly.

The running time is $O(t)$ time per token.

CountSketch summary

CountSketch is a one-pass randomised algorithm to estimate the frequency of the tokens in a stream.

Once ϵ and δ are decided we can set t and k accordingly.

The running time is $O(t)$ time per token.

The space usage is $O(t \log n + tk \log m)$ bits.

CountSketch summary

CountSketch is a one-pass randomised algorithm to estimate the frequency of the tokens in a stream.

Once ϵ and δ are decided we can set t and k accordingly.

The running time is $O(t)$ time per token.

The space usage is $O(t \log n + tk \log m)$ bits.

Assuming we set $k = 3/\epsilon^2$, for an arbitrary token a, the probability that CountSketch's estimate is further than $\epsilon \|\boldsymbol{f}_{-a}\|_2$ from the correct frequency is at most $\exp(-t/36)$.

Count-Min sketch

The sketch is a 2D-array C with t rows and k columns. All hash functions are chosen from a pairwise independent family.

Count-Min sketch

The sketch is a 2D-array C with t rows and k columns. All hash functions are chosen from a pairwise independent family.

```
stream ⟨a_1, ..., a_m⟩, a_i ∈ [n]
initialise C[1...t][1...k] = 0
choose hash functions h_1, ... h_t : [n] → [k]

Count-Min(a_i)
for each j ∈ [t]
      C[j, h_j(a_i)] += c_i



return f̂_a = min_{1≤i≤t} C[i, h_i(a)]
```

c_i is the number of instances of a_i. In the turnstile model this can be either positive of negative.

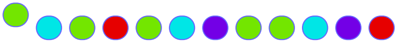
Count-Min - worked example

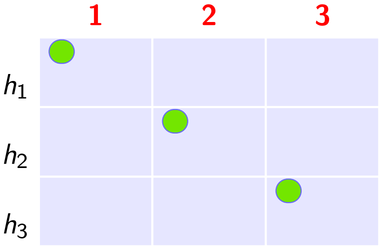
	1	2	3
h_1			
h_2			
h_3			

	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

```
Count-Min(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i
```

Count-Min - worked example

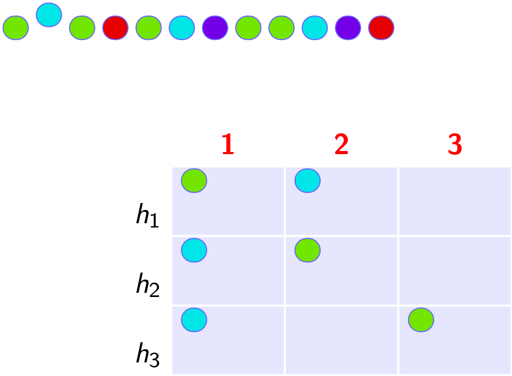




	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

$\textsc{Count-Min}(a_i)$
for each $j \in [t]$
    $C[j, h_j(a_i)] += c_i$

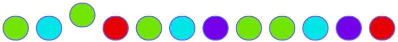
Count-Min - worked example

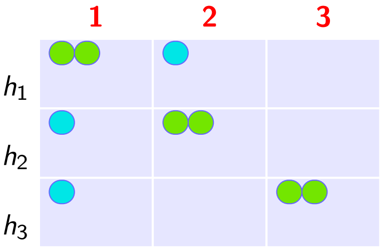


	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

```
Count-Min(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i
```

Count-Min - worked example

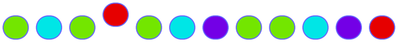


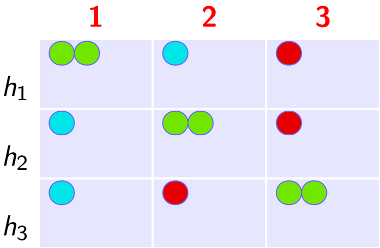


	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

```
Count-Min(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i
```

Count-Min - worked example

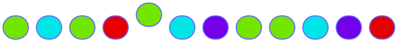


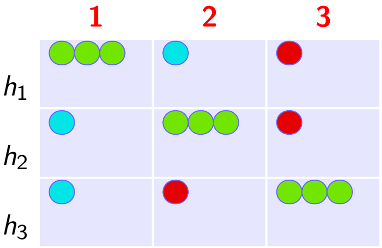


	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

Count-Min(a_i)
for each $j \in [t]$
    $C[j, h_j(a_i)] \mathrel{+}= c_i$

Count-Min - worked example

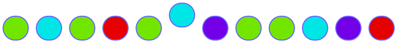


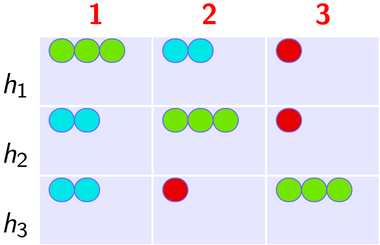


	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

Count-Min(a_i)
for each $j \in [t]$
    $C[j, h_j(a_i)] \mathrel{+}= c_i$

Count-Min - worked example

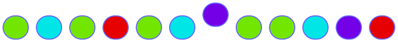


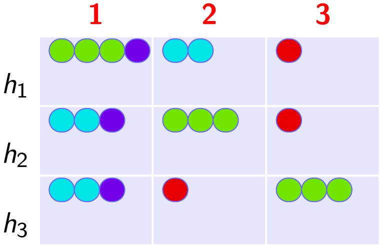


	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

```
Count-Min(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i
```

Count-Min - worked example

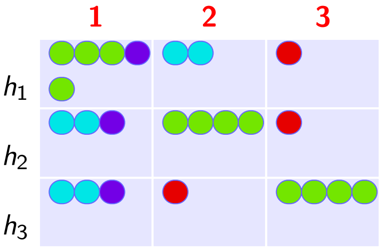




	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

```
COUNT-MIN(a_i)
for each j ∈ [t]
        C[j, h_j(a_i)] += c_i
```

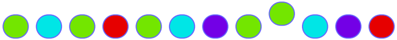
Count-Min - worked example

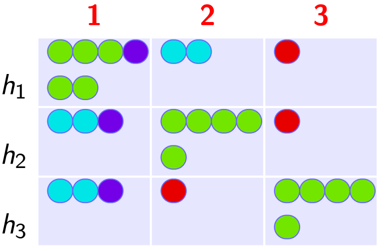


	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

`Count-Min`(a_i)
`for each` $j \in [t]$
    $C[j, h_j(a_i)] \mathrel{+}= c_i$

Count-Min - worked example

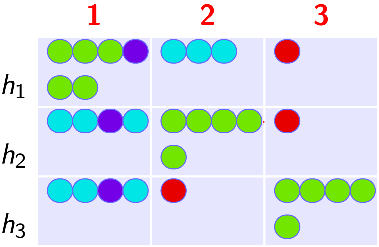




	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

Count-Min(a_i)
for each $j \in [t]$
  $C[j, h_j(a_i)] \mathrel{+}= c_i$

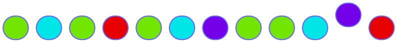
Count-Min - worked example

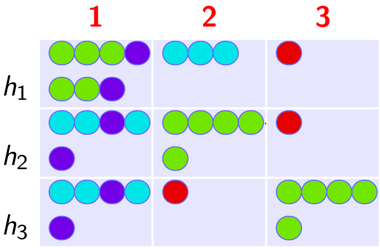


	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

Count-Min(a_i)
for each $j \in [t]$
    $C[j, h_j(a_i)] \mathrel{+}= c_i$

Count-Min - worked example

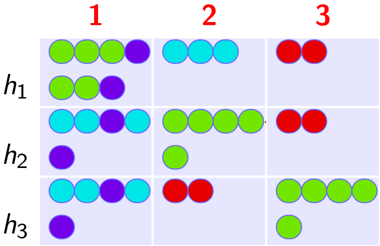




	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

Count-Min(a_i)
for each $j \in [t]$
$\quad C[j, h_j(a_i)] \mathrel{+}= c_i$

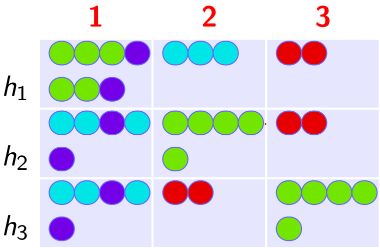
Count-Min - worked example



	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

`return` $\hat{f}_a = \min_{1 \leq i \leq t} C[i, h_i(a)]$

Count-Min - worked example

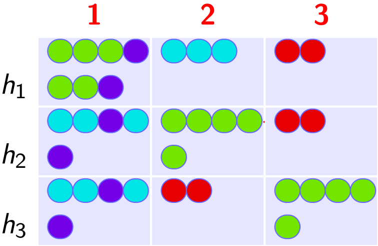


	h_1	h_2	h_3
$\bigcirc$	1	2	3
$\bigcirc$	2	1	1
$\bigcirc$	1	1	1
$\bigcirc$	3	3	2

`return` $\hat{f}_a = \min_{1 \le i \le t} C[i, h_i(a)]$

$\hat{f}_{\bigcirc} = \min(C[1, h_1(\bigcirc)], C[2, h_2(\bigcirc)], C[3, h_3(\bigcirc)]) = \min(7, 5, 5) = 5\checkmark$

Count-Min - worked example



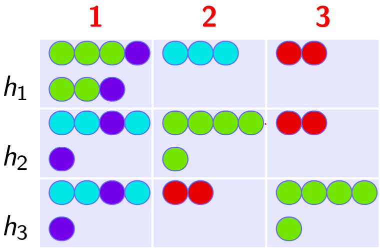
	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

`return` $\hat{f}_a = \min_{1 \leq i \leq t} C[i, h_i(a)]$

$\hat{f}$  $= \min(C[1, h_1($  $)], C[2, h_2($  $)], C[3, h_3($  $]))= \min(7, 5, 5) = 5 \checkmark$

$\hat{f}$  $= \min(C[1, h_1($  $)], C[2, h_2($  $)], C[3, h_3($  $]))= \min(3, 5, 5) = 3 \checkmark$

Count-Min - worked example

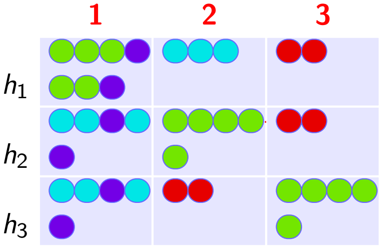


	h_1	h_2	h_3
	1	2	3
	2	1	1
	1	1	1
	3	3	2

```
return
```
$\hat{f}_a = \min_{1 \le i \le t} C[i, h_i(a)]$

$$\hat{f}_{\bullet} = \min(C[1, h_1(\bullet)], C[2, h_2(\bullet)], C[3, h_3(\bullet)])) = \min(7, 5, 5) = 5\checkmark$$
$$\hat{f}_{\bullet} = \min(C[1, h_1(\bullet)], C[2, h_2(\bullet)], C[3, h_3(\bullet)])) = \min(3, 5, 5) = 3\checkmark$$
$$\hat{f}_{\bullet} = \min(C[1, h_1(\bullet)], C[2, h_2(\bullet)], C[3, h_3(\bullet)])) = \min(7, 5, 5) = 5$$

COUNT-MIN - worked example



	h_1	h_2	h_3
$\bullet$	1	2	3
$\bullet$	2	1	1
$\bullet$	1	1	1
$\bullet$	3	3	2

`return` $\hat{f}_a = \min_{1 \le i \le t} C[i, h_i(a)]$

$\hat{f}_{\bullet} = \min(C[1, h_1(\bullet)], C[2, h_2(\bullet)], C[3, h_3(\bullet)])) = \min(7, 5, 5) = 5\checkmark$

$\hat{f}_{\bullet} = \min(C[1, h_1(\bullet)], C[2, h_2(\bullet)], C[3, h_3(\bullet)])) = \min(3, 5, 5) = 3\checkmark$

$\hat{f}_{\bullet} = \min(C[1, h_1(\bullet)], C[2, h_2(\bullet)], C[3, h_3(\bullet)])) = \min(7, 5, 5) = 5$

$\hat{f}_{\bullet} = \min(C[1, h_1(\bullet)], C[2, h_2(\bullet)], C[3, h_3(\bullet)])) = \min(2, 2, 2) = 2\checkmark$

Count-Min - Analysis I

For simplicity, consider positive counts of tokens (the cash register model) so that $\hat{f}_a \geq f_a$ for all tokens a.

Count-Min - Analysis I

For simplicity, consider positive counts of tokens (the cash register model) so that $\hat{f}_a \geq f_a$ for all tokens a.

Let $Y_{i,j} = 1$ if $h_i(j) = h_i(a)$ and 0 otherwise. Note that token j contributes to $C[i, h_i(a)]$ iff $Y_{i,j} = 1$.

COUNT-MIN - Analysis I

For simplicity, consider positive counts of tokens (the cash register model) so that $\hat{f}_a \geq f_a$ for all tokens a.

Let $Y_{i,j} = 1$ if $h_i(j) = h_i(a)$ and 0 otherwise. Note that token j contributes to $C[i, h_i(a)]$ iff $Y_{i,j} = 1$.

Let r.v. X_i be the excess count in cell $C[i, h_i(a)]$. That is

$$X_i = \sum_{j \in [n] \setminus \{a\}} f_j Y_{i,j}$$

Count-Min - Analysis I

For simplicity, consider positive counts of tokens (the cash register model) so that $\hat{f}_a \geq f_a$ for all tokens a.

Let $Y_{i,j} = 1$ if $h_i(j) = h_i(a)$ and 0 otherwise. Note that token j contributes to $C[i, h_i(a)]$ iff $Y_{i,j} = 1$.

Let r.v. X_i be the excess count in cell $C[i, h_i(a)]$. That is

$$X_i = \sum_{j \in [n] \setminus \{a\}} f_j Y_{i,j}$$

$$\mathbb{E}(X_i) = \sum_{j \in [n] \setminus \{a\}} f_j \mathbb{E}(Y_{i,j}) = \sum_{j \in [n] \setminus \{a\}} \frac{f_j}{k} = \frac{\|\boldsymbol{f}\|_1 - f_a}{k} = \frac{\|\boldsymbol{f}_{-a}\|_1}{k}$$

Count-Min - Analysis I

For simplicity, consider positive counts of tokens (the cash register model) so that $\hat{f}_a \geq f_a$ for all tokens a.

Let $Y_{i,j} = 1$ if $h_i(j) = h_i(a)$ and 0 otherwise. Note that token j contributes to $C[i, h_i(a)]$ iff $Y_{i,j} = 1$.

Let r.v. X_i be the excess count in cell $C[i, h_i(a)]$. That is

$$X_i = \sum_{j \in [n] \setminus \{a\}} f_j Y_{i,j}$$

$$\mathbb{E}(X_i) = \sum_{j \in [n] \setminus \{a\}} f_j \mathbb{E}(Y_{i,j}) = \sum_{j \in [n] \setminus \{a\}} \frac{f_j}{k} = \frac{\|\boldsymbol{f}\|_1 - f_a}{k} = \frac{\|\boldsymbol{f}_{-a}\|_1}{k}$$

By Markov's inequality

$$\Pr(X_i \geq \epsilon \|\boldsymbol{f}_{-a}\|_1) \leq \frac{\|\boldsymbol{f}_{-a}\|_1}{k\epsilon \|\boldsymbol{f}_{-a}\|_1} = \frac{1}{2} \qquad \text{set } k = 2/\epsilon$$

Count-Min - Analysis II

We have a bound for a single counter. Over t counters the reported excess is the minimum over all X_i. We can now derive the probability that all the excesses are at least $\epsilon \left\| \boldsymbol{f}_{-a} \right\|_1$ directly.

Count-Min - Analysis II

We have a bound for a single counter. Over t counters the reported excess is the minimum over all X_i. We can now derive the probability that all the excesses are at least $\epsilon \left\| \boldsymbol{f}_{-a} \right\|_1$ directly.

$$\Pr(\hat{f}_a - f_a \geq \epsilon \left\| \boldsymbol{f}_{-a} \right\|_1)) \leq \frac{1}{2^t} = \delta \qquad \text{set } t = \left\lceil \log_2 \left(\frac{1}{\delta} \right) \right\rceil$$

Count-Min - Analysis II

We have a bound for a single counter. Over t counters the reported excess is the minimum over all X_i. We can now derive the probability that all the excesses are at least $\epsilon \|\boldsymbol{f}_{-a}\|_1$ directly.

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_1)) \leq \frac{1}{2^t} = \delta \qquad \text{set } t = \left\lceil \log_2\left(\frac{1}{\delta}\right)\right\rceil$$

$k = 2/\epsilon, t = \lceil \log_2(\frac{1}{\delta})\rceil$ gives total space in bits

$$O\left(\frac{1}{\epsilon}\log\frac{1}{\delta}\cdot(\log m + \log n)\right)$$

COUNT-MIN - Analysis II

We have a bound for a single counter. Over t counters the reported excess is the minimum over all X_i. We can now derive the probability that all the excesses are at least $\epsilon \|\boldsymbol{f}_{-a}\|_1$ directly.

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_1)) \leq \frac{1}{2^t} = \delta \qquad \text{set } t = \left\lceil \log_2\left(\frac{1}{\delta}\right)\right\rceil$$

$k = 2/\epsilon, t = \lceil \log_2(\frac{1}{\delta})\rceil$ gives total space in bits

$$O\left(\frac{1}{\epsilon}\log\frac{1}{\delta}\cdot(\log m + \log n)\right)$$

The space usage is better than COUNTSKETCH by a factor of $1/\epsilon$.

Count-Min - Analysis II

We have a bound for a single counter. Over t counters the reported excess is the minimum over all X_i. We can now derive the probability that all the excesses are at least $\epsilon \|\boldsymbol{f}_{-a}\|_1$ directly.

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_1)) \leq \frac{1}{2^t} = \delta \qquad \text{set } t = \left\lceil \log_2\left(\frac{1}{\delta}\right) \right\rceil$$

$k = 2/\epsilon, t = \lceil \log_2(\frac{1}{\delta}) \rceil$ gives total space in bits

$$O\left(\frac{1}{\epsilon} \log \frac{1}{\delta} \cdot (\log m + \log n)\right)$$

The space usage is better than CountSketch by a factor of $1/\epsilon$. Count-Min's error probability is bounded by $\epsilon \|\boldsymbol{f}_{-a}\|_1$ instead of $\epsilon \|\boldsymbol{f}_{-a}\|_2$ for CountSketch.

$\textsc{Count-Min}$ - Analysis II

We have a bound for a single counter. Over t counters the reported excess is the minimum over all X_i. We can now derive the probability that all the excesses are at least $\epsilon \|\boldsymbol{f}_{-a}\|_1$ directly.

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_1)) \leq \frac{1}{2^t} = \delta \qquad \text{set } t = \left\lceil \log_2\left(\frac{1}{\delta}\right)\right\rceil$$

$k = 2/\epsilon, t = \lceil \log_2(\frac{1}{\delta})\rceil$ gives total space in bits

$$O\left(\frac{1}{\epsilon}\log\frac{1}{\delta}\cdot(\log m + \log n)\right)$$

The space usage is better than $\textsc{CountSketch}$ by a factor of $1/\epsilon$.

$\textsc{Count-Min}$'s error probability is bounded by $\epsilon \|\boldsymbol{f}_{-a}\|_1$ instead of $\epsilon \|\boldsymbol{f}_{-a}\|_2$ for $\textsc{CountSketch}$.

For all vectors $z \in \mathbb{R}^n$, we have that $\|z\|_1 \geq \|z\|_2$.

Frequency estimation - space/time summary

We have seen two one-pass sketching algorithms for frequency estimation.

Frequency estimation - space/time summary

We have seen two one-pass sketching algorithms for frequency estimation.

$\textsc{CountSketch}$ runs in $O(t) = O(\log \frac{1}{\delta})$ time per token if $t = \lceil 1/\delta \rceil$.

Frequency estimation - space/time summary

We have seen two one-pass sketching algorithms for frequency estimation.

CountSketch runs in $O(t) = O(\log \frac{1}{\delta})$ time per token if $t = \lceil 1/\delta \rceil$.

CountSketch space usage is

$$O(t \log n + tk \log m) = O\left(\frac{1}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)(\log m + \log n) \text{ bits}$$

bits if $k = \lceil 3/\epsilon^2 \rceil$.

Frequency estimation - space/time summary

We have seen two one-pass sketching algorithms for frequency estimation.

CountSketch runs in $O(t) = O(\log \frac{1}{\delta})$ time per token if $t = \lceil 1/\delta \rceil$.

CountSketch space usage is

$$O(t \log n + tk \log m) = O\left(\frac{1}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)(\log m + \log n) \text{ bits}$$

bits if $k = \lceil 3/\epsilon^2 \rceil$.

Count-Min runs in $O(t) = O(\log \frac{1}{\delta})$ time per token if $t = \lceil 1/\delta \rceil$.

Frequency estimation - space/time summary

We have seen two one-pass sketching algorithms for frequency estimation.

CountSketch runs in $O(t) = O(\log \frac{1}{\delta})$ time per token if $t = \lceil 1/\delta \rceil$.

CountSketch space usage is

$$O(t \log n + tk \log m) = O\left(\frac{1}{\epsilon^2} \log\left(\frac{1}{\delta}\right)\right)(\log m + \log n) \text{ bits}$$

bits if $k = \lceil 3/\epsilon^2 \rceil$.

Count-Min runs in $O(t) = O(\log \frac{1}{\delta})$ time per token if $t = \lceil 1/\delta \rceil$.

Count-Min space usage is

$$O(t \log n + tk \log m) = O\left(\frac{1}{\epsilon} \log\left(\frac{1}{\delta}\right)\right)(\log m + \log n)$$

bits if $k = \lceil 2/\epsilon \rceil$. This is a factor of $1/\epsilon$ improvement.

Frequency estimation - estimation error summary

CountSketch: with $k = \lceil 3/\epsilon^2 \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \left\| \boldsymbol{f}_{-a} \right\|_2)) \leq \delta$$

Frequency estimation - estimation error summary

CountSketch: with $k = \lceil 3/\epsilon^2 \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \left\| \boldsymbol{f}_{-a} \right\|_2) \leq \delta$$

Count-Min: with $k = \lceil 2/\epsilon \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \left\| \boldsymbol{f}_{-a} \right\|_1) \leq \delta$$

Frequency estimation - estimation error summary

CountSketch: with $k = \lceil 3/\epsilon^2 \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_2) \leq \delta$$

Count-Min: with $k = \lceil 2/\epsilon \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_1) \leq \delta$$

For all vectors $z \in \mathbb{R}^n$, we have that $\|z\|_1 \geq \|z\|_2$ so the estimation error is worse for Count-Min.

Frequency estimation - estimation error summary

CountSketch: with $k = \lceil 3/\epsilon^2 \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_2)) \leq \delta$$

Count-Min: with $k = \lceil 2/\epsilon \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_1)) \leq \delta$$

For all vectors $z \in \mathbb{R}^n$, we have that $\|z\|_1 \geq \|z\|_2$ so the estimation error is worse for Count-Min.

By setting $k = 1/\epsilon$, Misra-Gries gives us an estimate

$$f_j - \epsilon \|\boldsymbol{f}\|_1 \leq \hat{f}_j \leq f_j \text{ for every } j \in [n]$$

Frequency estimation - estimation error summary

$\textsc{CountSketch}$: with $k = \lceil 3/\epsilon^2 \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \left\| \boldsymbol{f}_{-a} \right\|_2)) \leq \delta$$

$\textsc{Count-Min}$: with $k = \lceil 2/\epsilon \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \left\| \boldsymbol{f}_{-a} \right\|_1)) \leq \delta$$

For all vectors $z \in \mathbb{R}^n$, we have that $\|z\|_1 \geq \|z\|_2$ so the estimation error is worse for $\textsc{Count-Min}$.

By setting $k = 1/\epsilon$, $\textsc{Misra-Gries}$ gives us an estimate

$$f_j - \epsilon \left\| \boldsymbol{f} \right\|_1 \leq \hat{f}_j \leq f_j \text{ for every } j \in [n]$$

$\textsc{Misra-Gries}$ gives a lower bound on frequency where $\textsc{Count-Min}$ gives an upper bound.

Frequency estimation - estimation error summary

COUNTSKETCH: with $k = \lceil 3/\epsilon^2 \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_2)) \leq \delta$$

COUNT-MIN: with $k = \lceil 2/\epsilon \rceil$ and $t = \lceil \log_2(1/\delta) \rceil$,

$$\Pr(\hat{f}_a - f_a \geq \epsilon \|\boldsymbol{f}_{-a}\|_1)) \leq \delta$$

For all vectors $z \in \mathbb{R}^n$, we have that $\|z\|_1 \geq \|z\|_2$ so the estimation error is worse for COUNT-MIN.

By setting $k = 1/\epsilon$, MISRA-GRIES gives us an estimate

$$f_j - \epsilon \|\boldsymbol{f}\|_1 \leq \hat{f}_j \leq f_j \text{ for every } j \in [n]$$

MISRA-GRIES gives a lower bound on frequency where COUNT-MIN gives an upper bound.

MISRA-GRIES uses $O((1/\epsilon)(\log m + \log n)$ bits but does not work in the turnstile model (with deletions).