Streaming model

- The input consists of \( m \) objects/items/tokens \( e_1, e_2, \ldots, e_m \) that are seen one by one by the algorithm.
- The algorithm has “limited” memory say for \( B \) tokens where \( B < m \) (often \( B \ll m \)) and hence cannot store all the input.
- Want to compute interesting functions over input.
Streaming model

- The input consists of \( m \) objects/items/tokens \( e_1, e_2, \ldots, e_m \) that are seen one by one by the algorithm.
- The algorithm has “limited” memory say for \( B \) tokens where \( B < m \) (often \( B \ll m \)) and hence cannot store all the input.
- Want to compute interesting functions over input.

Examples:
- Each token in a number from \([n]\)
- High-speed network switch: tokens are packets with source, destination IP addresses and message contents.
- Each token is an edge in graph (graph streams)
- Each token in a point in some feature space
- Each token is a row/column of a matrix
Streaming model

- The input consists of $m$ objects/items/tokens $e_1, e_2, \ldots, e_m$ that are seen one by one by the algorithm.
- The algorithm has “limited” memory say for $B$ tokens where $B < m$ (often $B \ll m$) and hence cannot store all the input.
- Want to compute interesting functions over input.

Examples:
- Each token in a number from $[n]$
- High-speed network switch: tokens are packets with source, destination IP addresses and message contents.
- Each token is an edge in graph (graph streams)
- Each token in a point in some feature space
- Each token is a row/column of a matrix

**Question:** What are the tradeoffs between memory size, accuracy, randomness and other resources?
Counting problem

Simplest streaming question: how many events in the stream?

Obvious: counter that increments on seeing each new item. Requires \( \lceil \log n \rceil = \Theta(\log n) \) bits to be able to count up to \( n \) events.

Question: can we do better?

"Counting large numbers of events in small registers" by Robert Morris (Bell Labs), Communications of the ACM (CACM), 1978.
Counting problem

Simplest streaming question: how many events in the stream?

Obvious: counter that increments on seeing each new item. Requires \[\lceil \log n \rceil = \Theta(\log n)\] bits to be able to count up to \(n\) events.
Counting problem

Simplest streaming question: how many events in the stream?

Obvious: counter that increments on seeing each new item. Requires $\lceil \log n \rceil = \Theta(\log n)$ bits to be able to count up to $n$ events.

Question: can we do better?
Counting problem

Simplest streaming question: how many events in the stream?

Obvious: counter that increments on seeing each new item. Requires \( \lceil \log n \rceil = \Theta(\log n) \) bits to be able to count up to \( n \) events.

**Question:** can we do better? Not deterministically.

“Counting large numbers of events in small registers” by Rober Morris (Bell Labs), Communications of the ACM (CACM), 1978
Probabilistic Counting Algorithm

**ProbabilisticCounting:**

\[ X \leftarrow 0 \]

While (a new event arrives)
  
  Toss a biased coin that is heads with probability \( \frac{1}{2^X} \)
  
  If (coin turns up heads)
    
    \[ X \leftarrow X + 1 \]

endWhile

Output \( 2^X - 1 \) as the estimate for the length of the stream.
Probabilistic Counting Algorithm

**ProbabilisticCounting:**

\[ X \leftarrow 0 \]

While (a new event arrives)
   
   Toss a biased coin that is heads with probability \( \frac{1}{2^X} \)

   If (coin turns up heads)
      
      \[ X \leftarrow X + 1 \]
   
endWhile

Output \( 2^X - 1 \) as the estimate for the length of the stream.

**Intuition:** \( X \) keeps track of \( \log n \) in a probabilistic sense. Hence requires \( O(\log \log n) \) bits
Probabilistic Counting Algorithm

**ProbabilisticCounting:**

\[ X \leftarrow 0 \]

While (a new event arrives)

- Toss a biased coin that is heads with probability \( \frac{1}{2^X} \)
- If (coin turns up heads)
  \[ X \leftarrow X + 1 \]

endWhile

Output \( 2^X - 1 \) as the estimate for the length of the stream.

**Intuition:** \( X \) keeps track of \( \log n \) in a probabilistic sense. Hence requires \( O(\log \log n) \) bits

**Theorem**

Let \( Y = 2^X \). Then \( E[Y] - 1 = n \), the number of events seen.
Analysis of Expectation

Induction on $n$. For $i \geq 0$, let $X_i$ be the counter value after $i$ events. Let $Y_i = 2^{X_i}$. Both are random variables.
Analysis of Expectation

Induction on $n$. For $i \geq 0$, let $X_i$ be the counter value after $i$ events. Let $Y_i = 2^{X_i}$. Both are random variables.

**Base case: $n = 0, 1$** easy to check: $X_i, Y_i \leftarrow 1$ deterministically equal to $0, 1$. 
Analysis of Expectation

\[ \mathbb{E}[Y_n] = \mathbb{E}[2^{X_n}] = \sum_{j=0}^{\infty} 2^j \Pr[X_n = j] \]

\[ = \sum_{j=0}^{\infty} 2^j \left( \Pr[X_{n-1} = j] \cdot \left(1 - \frac{1}{2^j}\right) + \Pr[X_{n-1} = j - 1] \cdot \frac{1}{2^{j-1}} \right) \]

\[ = \sum_{j=0}^{\infty} 2^j \Pr[X_{n-1} = j] \]

\[ + \sum_{j=0}^{\infty} (2 \Pr[X_{n-1} = j - 1] - \Pr[X_{n-1} = j]) \]

\[ = \mathbb{E}[Y_{n-1}] + 1 \quad \text{(by applying induction)} \]

\[ = n + 1 \]
Jensen’s Inequality

Definition

A real-valued function $f : \mathbb{R} \to \mathbb{R}$ is convex if

$$f((a + b)/2) \leq (f(a) + f(b))/2$$

for all $a, b$. Equivalently,

$$f(\lambda a + (1 - \lambda)b) \leq \lambda f(a) + (1 - \lambda)f(b)$$

for all $\lambda \in [0, 1]$. 
Jensen’s Inequality

Definition
A real-valued function $f : \mathbb{R} \to \mathbb{R}$ is convex if

$$f((a + b)/2) \leq (f(a) + f(b))/2$$

for all $a, b$. Equivalently,

$$f(\lambda a + (1 - \lambda)b) \leq \lambda f(a) + (1 - \lambda)f(b)$$

for all $\lambda \in [0, 1]$.

Theorem (Jensen’s inequality)
Let $Z$ be random variable with $E[Z] < \infty$. If $f$ is convex then

$$f(E[Z]) \leq E[f(Z)]$$.
Implication for counter size

We have $Y_n = 2^{X_n}$. The function $f(z) = 2^z$ is convex. Hence

$$2^{E[X_n]} \leq E[Y_n] \leq n + 1$$

which implies

$$E[X_n] \leq \log(n + 1)$$

Hence expected number of bits in counter is $O(\log \log n)$. 
Question: Is the random variable $Y_n$ well behaved even though expectation is right? What is its variance? Is it concentrated around expectation?

\[
\mathbb{E}[Y_n^2] = 3^2 n + 3^2 n + 1
\]
and hence
\[
\text{Var}[Y_n] = n(n-1)/2.
\]
Variance calculation

**Question:** Is the random variable $Y_n$ well behaved even though expectation is right? What is its variance? Is it concentrated around expectation?

**Lemma**

$$E[Y_n^2] = \frac{3}{2}n^2 + \frac{3}{2}n + 1 \text{ and hence } Var[Y_n] = \frac{n(n - 1)}{2}.$$
Variance analysis

Analyze $E[Y_n^2]$ via induction.

Base cases: $n = 0, 1$ are easy to verify since $Y_n$ is deterministic.

\[
E[Y_n^2] = E[2^{2X_n}] \sum_{j \geq 0} 2^{2j} \cdot Pr[X_n = j]
\]
\[
= \sum_{j \geq 0} 2^{2j} \cdot \left( Pr[X_{n-1} = j] (1 - \frac{1}{2j}) + Pr[X_{n-1} = j - 1] \frac{1}{2^{j-1}} \right)
\]
\[
= \sum_{j \geq 0} 2^{2j} \cdot Pr[X_{n-1} = j]
\]
\[
+ \sum_{j \geq 0} \left( -2^j Pr[X_{n-1} = j - 1] + 42^{j-1} Pr[X_{n-1} = j - 1] \right)
\]
\[
= E[Y_{n-1}^2] + 3E[Y_{n-1}]
\]
\[
= \frac{3}{2}(n - 1)^2 + \frac{3}{2}(n - 1) + 1 + 3n = \frac{3}{2}n^2 + \frac{3}{2}n + 1.
\]
Error analysis via Chebyshev inequality

We have $E[Y_n] = n$ and $Var(Y_n) = n(n - 1)/2$. Applying Chebyshev:

$$Pr[|Y_n - E[Y_n]| \geq tn] \leq 1/(2t^2).$$

Hence constant factor approximation with constant probability (for instance set $t = 1/2$).
Error analysis via Chebyshev inequality

We have $\mathbb{E}[Y_n] = n$ and $\text{Var}(Y_n) = n(n - 1)/2$. Applying Chebyshev:

$$\Pr[|Y_n - \mathbb{E}[Y_n]| \geq tn] \leq 1/(2t^2).$$

Hence constant factor approximation with constant probability (for instance set $t = 1/2$).

**Question:** Want estimate to be tighter. For any given $\epsilon > 0$ want estimate to have error at most $\epsilon n$ with say constant probability or with probability at least $(1 - \delta)$ for a given $\delta > 0$. 

Variance reduction via averaging

- Run $h$ parallel copies of algorithm with *independent* randomness
- Let $Y^{(1)}, Y^{(2)}, \ldots, Y^{(h)}$ be estimators from the $h$ parallel copies
- Output $Z = \frac{1}{h} \sum_{i=1}^{h} Y^{(i)}$

Claim: $\mathbb{E}[Z] = n$ and $\text{Var}(Z) = \frac{1}{h} \left( \frac{n(n-1)}{2} \right)$.

Choose $h = \frac{2}{\epsilon^2}$. Then applying Chebyshev $\Pr[|Z - \mathbb{E}[Z]| \geq \epsilon n] \leq \frac{1}{4}$.

To run $h$ copies need $O\left( \frac{1}{\epsilon^2} \log \log n \right)$ bits for the counters.
Variance reduction via averaging

- Run \( h \) parallel copies of algorithm with *independent* randomness
- Let \( Y^{(1)}, Y^{(2)}, \ldots, Y^{(h)} \) be estimators from the \( h \) parallel copies
- Output \( Z = \frac{1}{h} \sum_{i=1}^{h} Y^{(i)} \)

**Claim:** \( E[Z_n] = n \) and \( \text{Var}(Z_n) = \frac{1}{h} \left( n(n-1)/2 \right) \).
Variance reduction via averaging

- Run \( h \) parallel copies of algorithm with *independent* randomness
- Let \( Y^{(1)}, Y^{(2)}, \ldots, Y^{(h)} \) be estimators from the \( h \) parallel copies
- Output \( Z = \frac{1}{h} \sum_{i=1}^{h} Y^{(i)} \)

Claim: \( \mathbb{E}[Z_n] = n \) and \( \text{Var}(Z_n) = \frac{1}{h}(n(n-1)/2) \).

Choose \( h = 2/\epsilon^2 \). Then applying Chebyshev

\[
\Pr[|Z_n - \mathbb{E}[Z_n]| \geq \epsilon n] \leq 1/4.
\]
Variance reduction via averaging

- Run $h$ parallel copies of algorithm with *independent* randomness.
- Let $Y^{(1)}, Y^{(2)}, \ldots, Y^{(h)}$ be estimators from the $h$ parallel copies.
- Output $Z = \frac{1}{h} \sum_{i=1}^{h} Y^{(i)}$

Claim: $E[Z_n] = n$ and $Var(Z_n) = \frac{1}{h} \left( n(n - 1)/2 \right)$.

Choose $h = 2/\epsilon^2$. Then applying Chebyshev

$$\Pr[|Z_n - E[Z_n]| \geq \epsilon n] \leq 1/4.$$  

To run $h$ copies need $O\left( \frac{1}{\epsilon^2} \log \log n \right)$ bits for the counters.
Error reduction via median trick

We have:

\[ \Pr[|Z_n - E[Z_n]| \geq \epsilon n] \leq 1/4. \]

Want:

\[ \Pr[|Z_n - E[Z_n]| \geq \epsilon n] \leq \delta \]

for some given parameter \( \delta \).
Error reduction via median trick

We have:

$$\Pr[|Z_n - E[Z_n]| \geq \epsilon n] \leq 1/4.$$ 

Want:

$$\Pr[|Z_n - E[Z_n]| \geq \epsilon n] \leq \delta$$

for some given parameter $\delta$.

Idea: Repeat independently $c \log(1/\delta)$ times. We know that with probability $(1 - \delta)$ one of the counters will be $\epsilon n$ close to $n$. Which one?
Error reduction via median trick

We have:

$$\Pr[|Z_n - E[Z_n]| \geq \epsilon n] \leq 1/4.$$  

Want:

$$\Pr[|Z_n - E[Z_n]| \geq \epsilon n] \leq \delta$$

for some given parameter $\delta$.

**Idea:** Repeat independently $c \log(1/\delta)$ times. We know that with probability $(1 - \delta)$ one of the counters will be $\epsilon n$ close to $n$. Which one?

**Algorithm:** Output median of $Z^{(1)}, Z^{(2)}, \ldots, Z^{(\ell)}$. 
Let $Z'$ be median of the $\ell = c \log(1/\delta)$ independent estimators.

**Lemma**

$$\Pr[|Z' - n| \geq \epsilon n] \leq \delta.$$
Let $Z'$ be median of the $\ell = c \log(1/\delta)$ independent estimators.

**Lemma**

$$\Pr[|Z' - n| \geq \epsilon n] \leq \delta.$$ 

- Let $A_i$ be event that estimate $Z^{(i)}$ is *bad*: that is, $|Z^{(i)} - n| > \epsilon n$. $\Pr[A_i] < 1/4$. Hence expected number of bad estimates is $\ell/4$. 

Let $Z'$ be median of the $\ell = c \log(1/\delta)$ independent estimators.

**Lemma**

$$\Pr[|Z' - n| \geq \epsilon n] \leq \delta.$$ 

- Let $A_i$ be event that estimate $Z^{(i)}$ is bad: that is, $|Z^{(i)} - n| > \epsilon n$. $\Pr[A_i] < 1/4$. Hence expected number of bad estimates is $\ell/4$.

- For median estimate to be bad, more than half of $A_i$’s have to be bad.
Let $Z'$ be median of the $\ell = c \log(1/\delta)$ independent estimators.

**Lemma**

$$\Pr[|Z' - n| \geq \epsilon n] \leq \delta.$$  

- Let $A_i$ be event that estimate $Z^{(i)}$ is *bad*: that is, $|Z^{(i)} - n| > \epsilon n$. $\Pr[A_i] < 1/4$. Hence expected number of bad estimates is $\ell/4$.
- For median estimate to be bad, more than half of $A_i$'s have to be bad.
- Using Chernoff bounds: probability of bad median is at most $2^{-c'\ell}$ for some constant $c'$. 

Summarizing

Using variance reduction and median trick: with $O\left(\frac{1}{\epsilon^2} \log(1/\delta) \log \log n\right)$ bits one can maintain a $(1 - \epsilon)$-factor estimate of the number of events with probability $(1 - \delta)$.

Can do (much) better by changing algorithm and better analysis. See homework and references in notes.