LECTURE 6

Last time

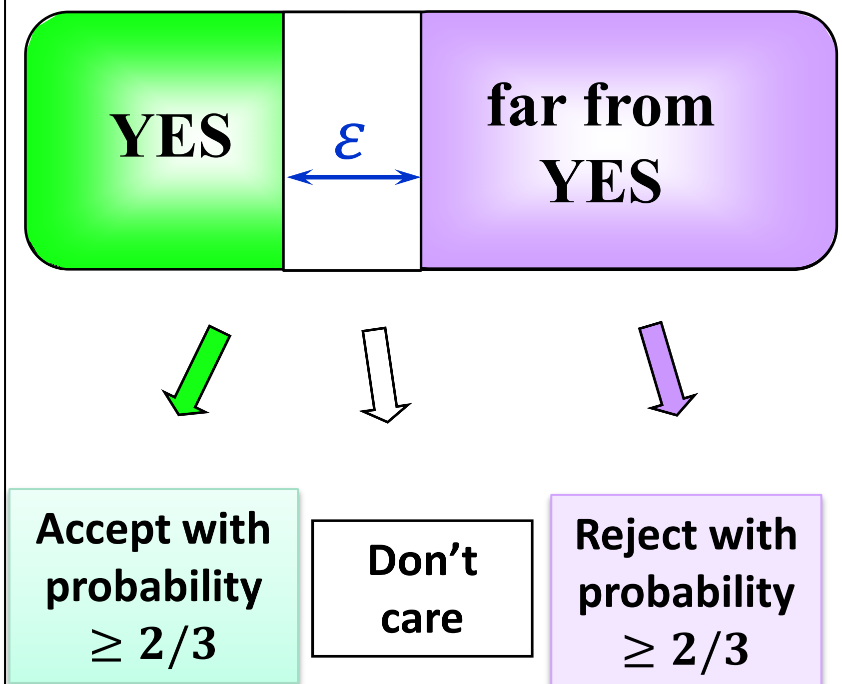
- Communication complexity
- Testing with adversarial erasures

Today

- Other models of computation
- Streaming

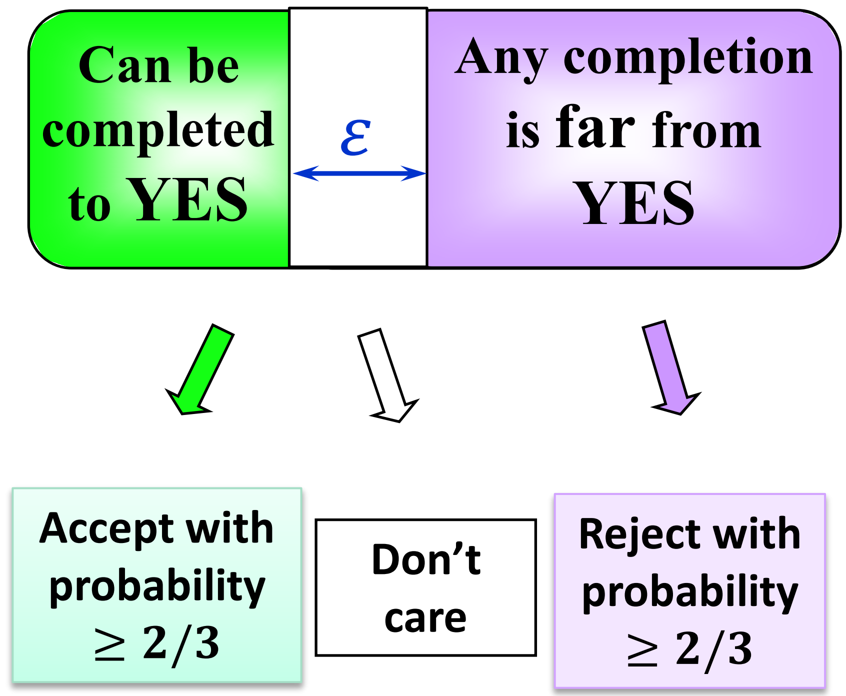
Property Testing with Erasures

Property Tester [Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]



Erasure-Resilient Property Tester [Dixit Raskhodnikova Thakurta Varma 16]

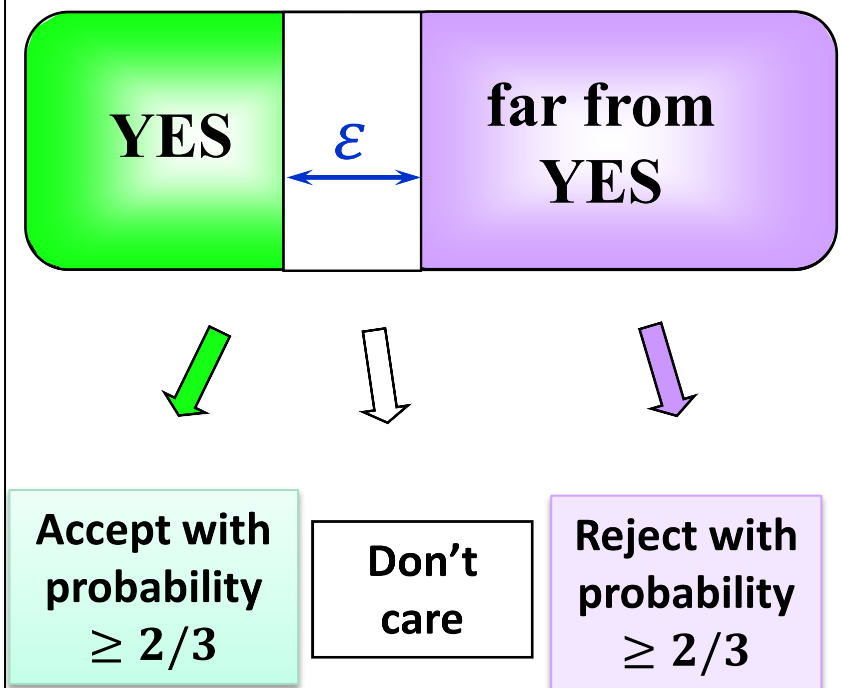
- $\leq \alpha$ fraction of the input is erased adversarially



Two objects are at distance ε = they differ in an ε fraction of places

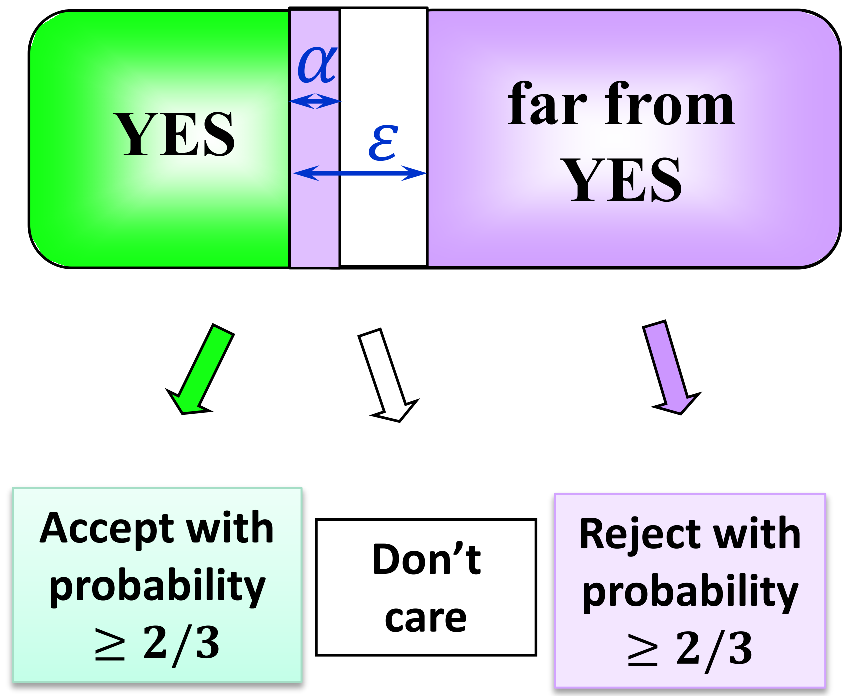
Property Testing with Errors

Property Tester [Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]



Tolerant Property Tester [Parnas Ron Rubinfeld 06]

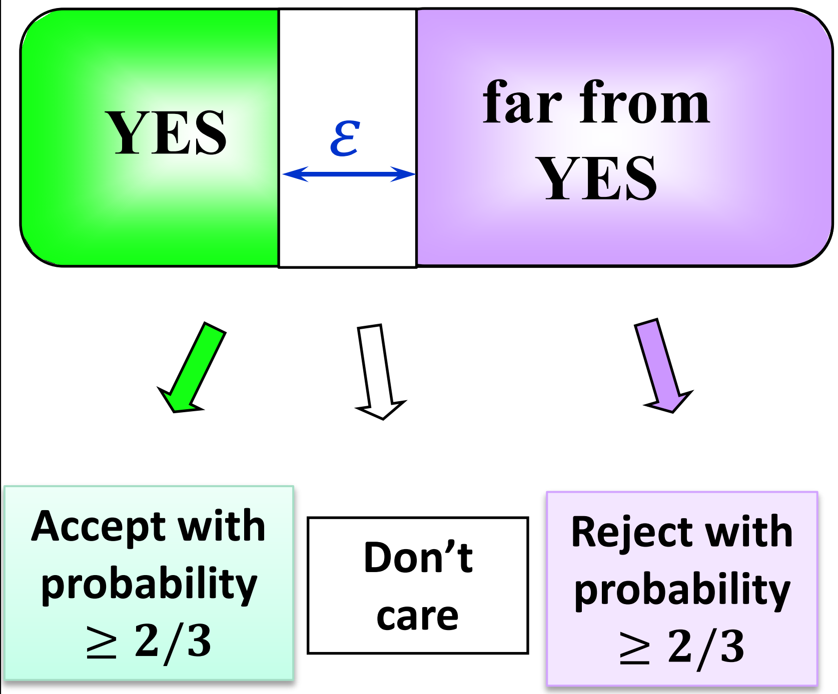
- $\leq \alpha$ fraction of the input is wrong



Two objects are at distance ε = they differ in an ε fraction of places

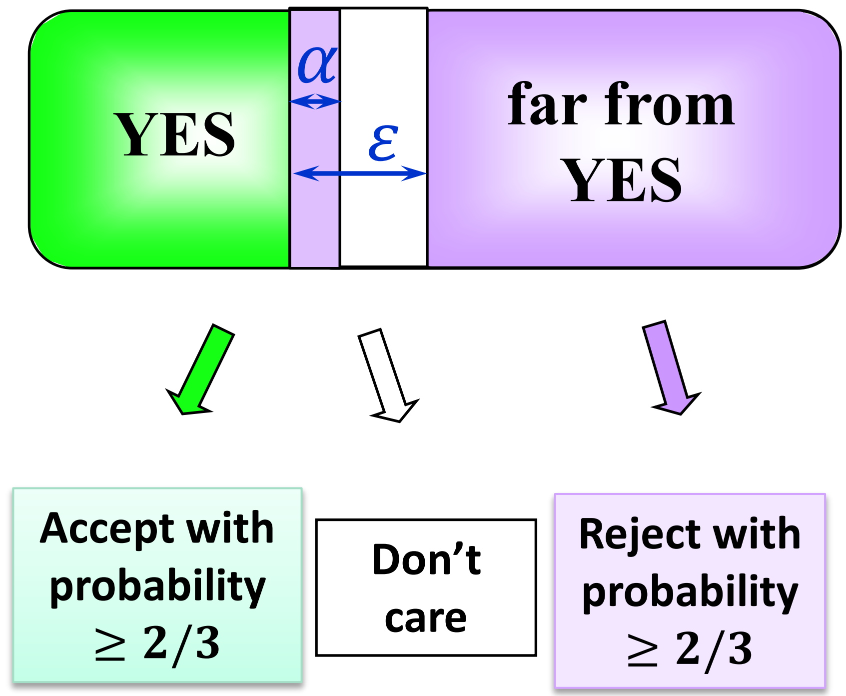
Property Testing with Errors

Property Tester [Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]



Tolerant Property Tester [Parnas Ron Rubinfeld 06]

- $\leq \alpha$ fraction of the input is wrong

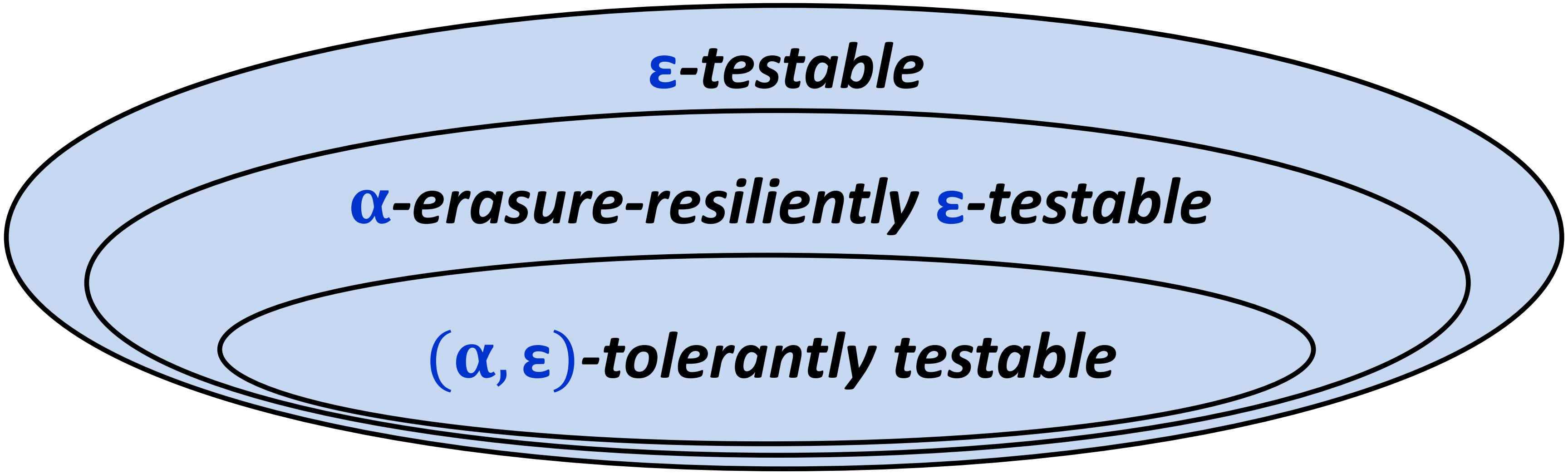


Two objects are at distance ε = they differ in an ε fraction of places

Relationships Between Models

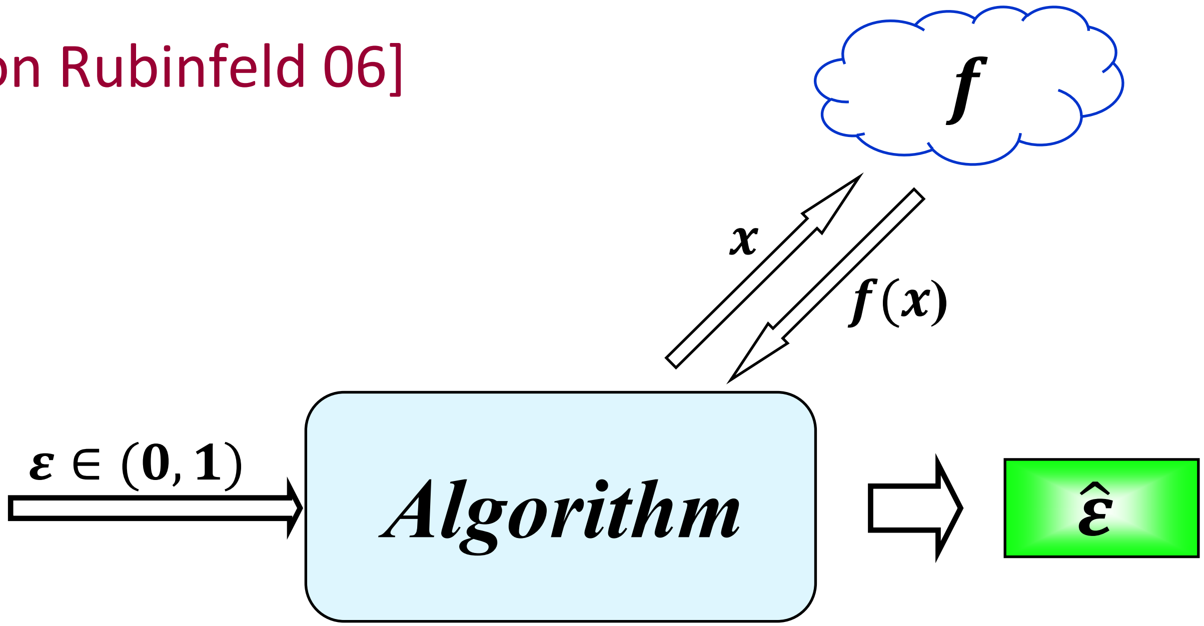
Containments are strict:

- [Fischer Fortnow 05]: standard vs. tolerant
- [Dixit **R** Thakurta Varma 16]: standard vs. erasure-resilient
- [**R** Ron-Zewi Varma 19]: erasure-resilient vs. tolerant



Distance Approximation for Boolean Functions

[Parnas Ron Rubinfeld 06]

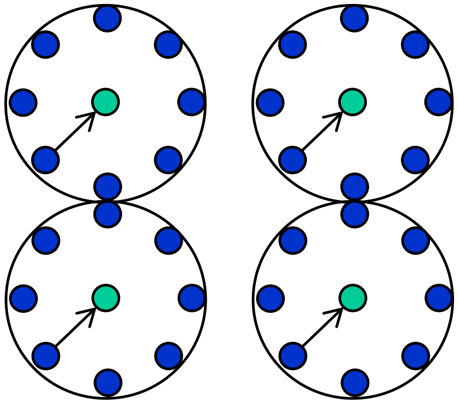


Goal: Output $dist(f, \mathcal{P}) \pm \varepsilon$

in sublinear time

Sublinear-Time "Restoration" Models

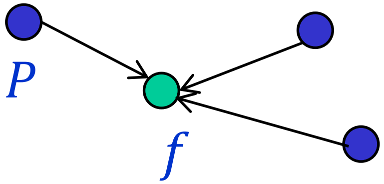
Local Decoding



Input: A slightly corrupted codeword

Requirement: Recover individual bits of the closest codeword with a constant number of queries per recovered bit.

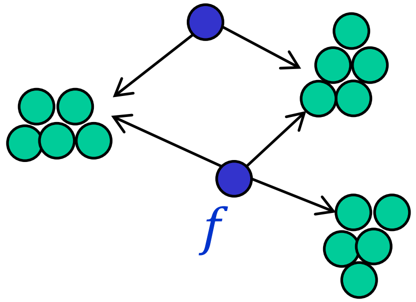
Program Checking



Input: A program P computing f correctly on most inputs.

Requirement: Self-correct program P: for a given input x, compute $f(x)$ by making a few calls to P.

Local Reconstruction



Input: Function f nearly satisfying some property P

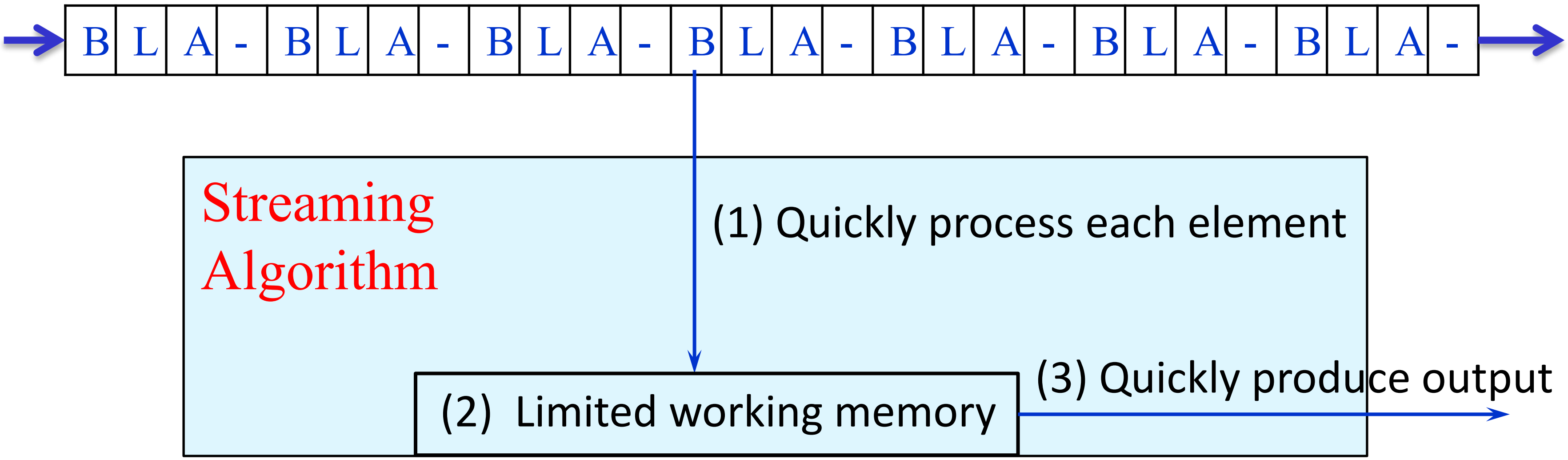
Requirement: Reconstruct function f to ensure that the reconstructed function g satisfies P, changing f only when necessary. For each input x, compute $g(x)$ with a few queries to f.

Generalization: Local Computation

[Rubinfeld Tamir Vardi Xie 2011]

- Compute the i-th character y_i of a legal output y.
- If there are several legal outputs for a given input, be consistent with one.
- Example: maximal independent set in a graph.

Data Stream Model [Alon Matias Szegedy 96]



Motivation: internet traffic analysis

Model the stream as m elements from $[n]$, e.g.,

$$\langle a_1, a_2, \dots, a_m \rangle = 3, 5, 3, 7, 5, 4, \dots$$

Goal: Compute a function of the stream, e.g., median, number of distinct elements, longest increasing sequence.

Based on Andrew McGregor's slides: http://www.cs.umass.edu/~mcgregor/slides/10-jhu1.pdf

Streaming Puzzle

A stream contains $n-1$ **distinct** elements from $[n]$ in arbitrary order.

Problem: Find the missing element, using $O(\log n)$ space.

Sampling from a Stream of Unknown Length

Warm-up: Find a uniform sample s from a stream $\langle a_1, a_2, \ldots, a_m \rangle$ of *known* length m.

Sampling from a Stream of Unknown Length

Problem: Find a uniform sample s from a stream $\langle a_1, a_2, \dots, a_m \rangle$ of *unknown* length m

Algorithm (Reservoir Sampling)

1. Initially, $s \leftarrow a_1$
2. On seeing the t^{th} element, $s \leftarrow a_t$ with probability $1/t$

Analysis:

What is the probability that $s = a_i$ at some time $t \geq i$?

$$\Pr[s = a_i] = \frac{1}{i} \cdot \left(1 - \frac{1}{i+1}\right) \cdot \ldots \cdot \left(1 - \frac{1}{t}\right)$$

$$= \frac{1}{i} \cdot \frac{i}{i+1} \cdot \ldots \cdot \frac{t-1}{t} = \frac{1}{t}$$

Space: $O(k\ (\log n + \log m))$ bits to get k samples.

Frequency Moments Estimation

Input: a stream $\langle a_1, a_2, \dots, a_m \rangle \in [n]^m$

- The frequency vector of the stream is $f = (f_1, \dots, f_n)$, where f_i is the number of times i appears in the stream
- The p-th frequency moment is $F_p = \left||f|\right|_p^p = \sum_{i=1}^{n} f_i^p$

  F_0 is the number of nonzero entries of f (# of distinct elements)

  $F_1 = m$ (# of elements in the stream)

  $F_2 = \left||f|\right|_2^2$ is a measure of non-uniformity

  used e.g. for anomaly detection in network analysis

  $F_\infty = \max_i f_i$ is the most frequent element

Goal: Estimate F_p up to a multiplicative factor $(1 \pm \varepsilon)$ with probability $\geq 2/3$

Summary

Streaming Model

- Reservoir sampling
- Distinct Elements (approximating F_0)
- k-wise independent hashing