

Distinct Elements and Moment Estimation

Lecture 1

Probability

Events, probabilities,
conditional probabilities,
union bound

Random variables,
expected value,
linearity of expectation
Markov's inequality

Algorithms

Approx. 3-SAT,

Big idea: we only want the average

$$\begin{aligned} \text{Linearity of Expectation} \\ E[X+Y] &= E[X] + E[Y] \\ \text{"average sum = sum of averages"} \end{aligned}$$

let $Y_i = \begin{cases} 1 & \text{if } i\text{th clause satisfied} \\ 0 & \text{otherwise} \end{cases}$ ($i=1, \dots, m$)

$$E[Y_i] = 7/8 \quad (1 \text{ clause case})$$

$$E[\text{\# clauses satisfied}] = E\left[\sum_{i=1}^m Y_i\right] = \sum_{i=1}^m E[Y_i] = 7/8 m$$

let $Y_i = \begin{cases} 1 & \text{if } i\text{th clause satisfied} \\ 0 & \text{otherwise} \end{cases}$

$$Z = \sum_{i=1}^m Y_i = \text{\# clauses satisfied}$$

$$E[Z] = \frac{1}{2} E[Z | x_1 = \text{true}] + \frac{1}{2} E[Z | x_1 = \text{false}]$$

$\Rightarrow$ either $E[Z | x_1 = \text{true}]$ or $E[Z | x_1 = \text{false}]$ is $\geq E[Z]$!

QuickSort,

Random divide and conquer

① pick random 'pivot'



② Divide A into two parts



Expected running time of QuickSort

For each $i, j \in [n]$ w/ $i < j$,

$X_{ij} = \begin{cases} 1 & \text{if } i\text{th smallest element is compared to } j\text{th smallest} \\ 0 & \text{otherwise} \end{cases}$

$\sum_{i < j} X_{ij}$ = total # comparisons made by the algorithm
 $\approx$ total running time

$E[\sum_{i < j} X_{ij}]$ = average running time

$$E[\sum_{i < j} X_{ij}] = \sum_{i < j} E[X_{ij}]$$

now analyze $E[X_{ij}]$ in isolation

$$\begin{aligned} E[X_{ij}] &= \Pr[X_{ij} = 1] \\ &= \Pr[i\text{th element is compared to } j\text{th}] \end{aligned}$$



i th elem is compared w/ j th
 $\Leftrightarrow$ i th or j th elem selected before any element in between happens w/ probability $\frac{2}{j-i+1}$

$$E[X_{ij}] = \frac{2}{j-i+1}$$

$$E[\sum_{i < j} X_{ij}] = \sum_{i < j} \frac{2}{j-i+1}$$

$$= \sum_{i=1}^n \sum_{j=i+1}^n \frac{2}{j-i+1} = \sum_{k=1}^n \sum_{i=1}^k \frac{2}{k+1-i+1}$$

$$\leq 2n \left(\frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n} \right)$$

$$\leq O(n \log n) \quad \text{with harmonic series}$$

Quickselect (HW)

Lecture 2

Probability

Hash Functions,

Universal Hash Functions

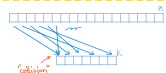
$h: [n] \rightarrow k$ s.t. for all pairs $i, j \in [n]$,
 $\Pr[h(i) = h(j)] \leq \frac{1}{k}$ (much weaker than ideal hash)

e.g. $h(x) = (ax + b) \bmod p \bmod k$

where • p prime, $> k$

• a in $\{1, \dots, p-1\}$ unif. random

• b in $\{0, \dots, p-1\}$ unif. random



Markov's Inequality

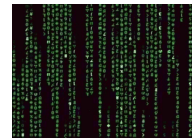
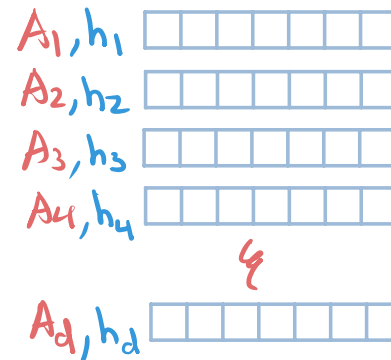
Markov's Inequality (special case)

Let $X \geq 0$ be nonnegative RV

$$\Pr[X \geq 2E[X]] \leq \frac{1}{2}$$

Algorithms

Frequency Estimation, Heavy Hitters (in streams)



estimate f_i w/
 $\min_{j=1, \dots, d} A_j[h_j(i)]$

Lecture 3 (last Tuesday)

Probability

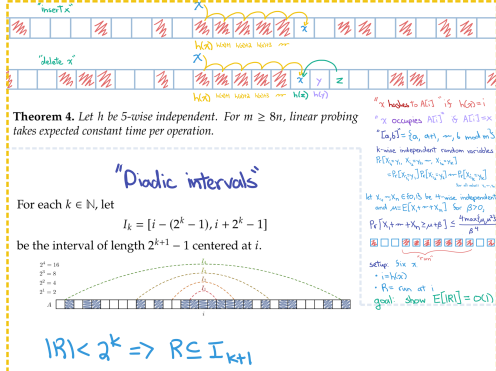
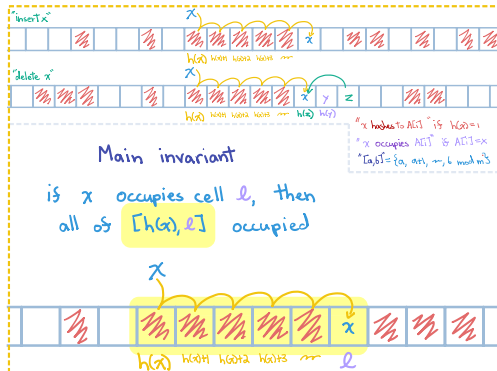
k-wise independence,
higher order moments
(generalizing universal hash
functions and Markov's inequality)

Algorithms

Hash tables w/
chaining +
linear probing

k-wise independent hash functions
 $h: [U] \rightarrow [m]$
s.t. for any k keys $x_1, x_2, \dots, x_k \in [U]$,
 k values $v_1, v_2, \dots, v_k \in [m]$,
 $P[h(x_1)=v_1, h(x_2)=v_2, \dots, h(x_k)=v_k] = \frac{1}{m^k}$
eg. $h(x) = a_0 + a_1x + a_2x^2 + \dots + a_{k-1}x^{k-1} \pmod p$
where $p \geq U$ prime,
 $a_0, a_1, \dots, a_{k-1} \in [p]$ indep, uniformly random

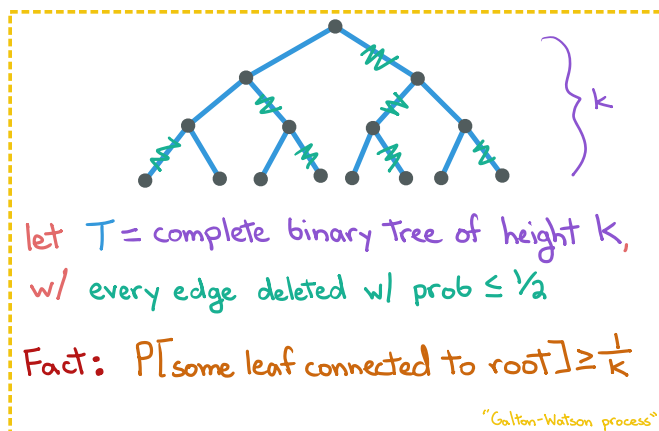
Important lemma
k-wise independent random variables
 $P[X_1=y_1, X_2=y_2, \dots, X_k=y_k]$
 $= P[X_1=y_1]P[X_2=y_2] \dots P[X_k=y_k]$
for all values $y_1, \dots, y_k$
let $X_1, \dots, X_n \in \{0,1\}$ be 4-wise independent
and $\mu = E[X_1 + \dots + X_n]$. then for $\beta > 0$,
 $P[X_1 + \dots + X_n \geq \mu + \beta] \leq \frac{4 \max\{\mu, \mu^3\}}{\beta^4}$



Lecture 4 (last Thursday)

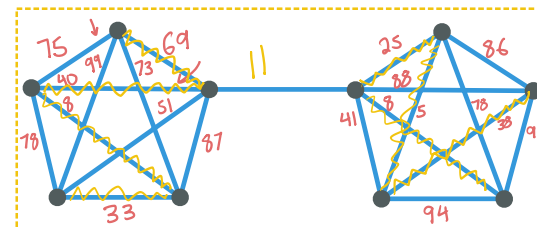
Probability

Galton-Watson processes



Algorithms

Random Contractions for min cut



Randomized contractions

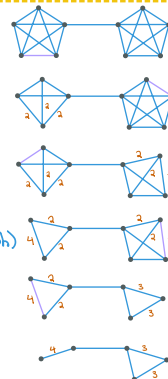
Input: $G=(V,E)$, weights $c \in \mathbb{R}_{\geq 0}^E$

① while $|E| > 1$

② sample $e \in E$ in proportion to c

③ "contract" e

④ return set of edges (in original graph) contracted to remaining edge



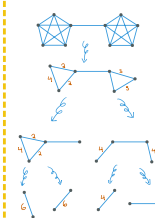
Amplification by branching

Instead of restarting at the beginning, restart partway through

High-level algorithm

① randomly contract until prob. of ruining the min-cut is $\sim 1/2$

② Branch:
make two copies of current graph
recurse on both copies



Lecture 5

Probability

Multiplicative Chernoff Bound

Multiplicative Chernoff bound

let $X_1, \dots, X_n \in [0, 1]$ be independent

$$\mu = E[X_1 + \dots + X_n]$$

$$\epsilon \in (0, 1)$$

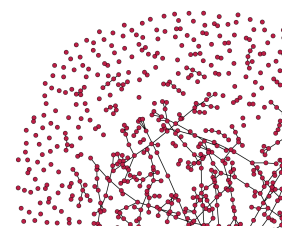
- $P[X_1 + \dots + X_n \geq (1 + \epsilon)\mu] \leq e^{-\epsilon^2 \mu / 3}$
- $P[X_1 + \dots + X_n \leq (1 - \epsilon)\mu] \leq e^{-\epsilon^2 \mu / 2}$

Algorithms (/ results)

Random graphs & the giant component

Erdős-Rényi graphs $G(n, p)$

- n vertices
- each edge sampled independently w/ probability p



Theorem 5.3. Consider a random graph $G \sim \mathcal{G}(n, p)$ for $p = c/n$, where c is a constant.

- If $c > 1$, then with high probability, there is exactly one connected component of G with $\Omega(n)$ vertices, and all other components have size $\leq O(\log n)$.
- For $c < 1$, then with high probability, all connected components of G has size $< O(\log n)$.

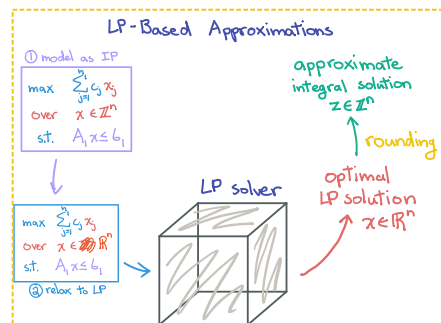
Lecture 6

Probability (& techniques)

Approximations via linear programming & randomized rounding

Linear programs (LPs)
Similar to IPs, except no integrality constraints

min/max $\sum_{j=1}^n c_j x_j$ linear objective
over $x \in \mathbb{R}^n$ linear integral variables
st. $A_1 x \leq b_1$ linear constraints
 $A_2 x = b_2$
 $A_3 x \geq b_3$



Continuous optimization vs Discrete optimization

randomization

Algorithms

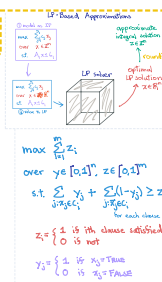
Max Sat Set Cover Covering Integer Programs

For each variable x_j , independently, randomly set $x_j = 1$ with probability p_j , and $x_j = 0$ otherwise.

$$E[\# \text{ sat. clauses}] \geq (1 - 1/e) \text{OPT}$$

k	oblivious	LP	average
1	1/2	$z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
2	3/4	$(3/4)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
3	7/8	$(1 - 1/e)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
4	15/16	$(1 - 1/e)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
5	31/32	$(1 - 1/e)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$
...	...	...	...
k	$1 - 1/2^k$	$(1 - 1/e)z_i \geq (3/4)z_i$	$\geq (3/4)z_i$

Can we get $\frac{3}{4}$ -APX?



Rounding LP solution
 $x \in [0, 1]^n \rightsquigarrow F \subseteq \{S_1, \dots, S_n\}$

For each set S_j , independently, take S_j with probability x_j .

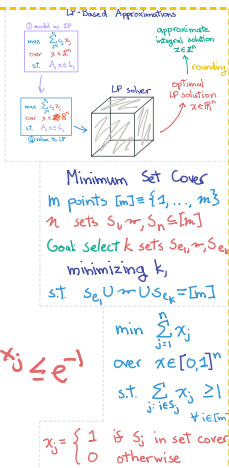
$$E[|F|] = \sum x_j \leq \text{OPT}$$

Fix a point i .

$$E[\# \text{ sets in } F \text{ covering } i] = \sum_{j \in S_i} x_j \geq z_i$$

$$P[F \text{ not cover } i] = \prod_{j \in S_i} (1 - x_j) \leq e^{-\sum_{j \in S_i} x_j} \leq e^{-z_i}$$

$$P[F \text{ cover } i] \geq 1 - e^{-1}$$



Lecture 7

Probability
(and techniques)

Algorithms
(models and applications)

Caching

Buy-or-rent

Today

Probability (& techniques)

Algorithms

Continuous
Random Variables

Variance

Variance Reduction

Distinct Elements

F_2 estimation



Get essential customer insights.

Get a complete understanding of your customers across devices and platforms. Google Analytics gives you the tools, free of charge, to understand the customer journey and improve marketing ROI.

Get started today

See what's in it for you.

This article is about Universal Analytics properties, which will [stop processing data](#) on July 1, 2023 (October 1, 2023 for Analytics 360 properties). If you haven't already, [start using a Google Analytics 4 property](#).

How users are identified for user metrics

The Users and Active Users metrics show how many users engaged with your site or app.

In order for Google Analytics to determine which traffic belongs to which user, a unique identifier associated with each user is sent with each hit. This identifier can be a single, first-party cookie named `_ga` that stores a Google Analytics client ID, or you can use the [User-ID feature](#) in conjunction with the client ID to more accurately identify users across all the devices they use to access your site or app. For more information on identifiers, read about [cookies and user identification](#) in our developer documentation.

In early 2017, Google Analytics began updating the calculation for the Users and Active Users metrics to more efficiently count users with high accuracy and low error rate (typically less than 2%). The metrics are available across your standard reports. You may notice a small difference in user count from the previous calculation method (explained below), with the exception of [Unsampled Reports](#) and [BigQuery Export](#), which will continue to use the previous calculation method. The New Users calculation also still uses the previous method.

This new calculation is applied to user data back through September 2016. If the date range for your report includes data from prior to September 2016, then sampling occurs. If you want unsampled reports that include data prior to 2016, then use the directions below to turn off the new calculation method.

As of August 2017, the new calculation is applied to data in [Custom Tables](#).

<https://support.google.com/analytics/answer/2992042>

HyperLogLog in Practice: Algorithmic Engineering of a State of The Art Cardinality Estimation Algorithm

Stefan Heule
ETH Zurich and Google, Inc.
stheule@ethz.ch

Marc Nunkesser
Google, Inc.
marcnunkesser@google.com

Alexander Hall
Google, Inc.
alexhall@google.com

ABSTRACT

Cardinality estimation has a wide range of applications and is of particular importance in database systems. Various algorithms have been proposed in the past, and the HYPERLOGLOG algorithm is one of them. In this paper, we present a series of improvements to this algorithm that reduce its memory requirements and significantly increase its accuracy for an important range of cardinalities. We have implemented our proposed algorithm for a system at Google and evaluated it empirically, comparing it to the original HYPERLOGLOG algorithm. Like HYPERLOGLOG, our improved algorithm parallelizes perfectly and computes the cardinality estimate in a single pass.

1. INTRODUCTION

Cardinality estimation is the task of determining the number of distinct elements in a data stream. While the cardinality can be easily computed using space linear in the cardinality, for many applications, this is completely impractical and requires too much memory. Therefore, many algorithms that approximate the cardinality while using less resources have been developed. These algorithms play an important role in network monitoring systems, data mining applications, as well as database systems.

At Google, various data analysis systems such as Sawzall [15], Dremel [13] and PowerDrill [9] estimate the cardinality of very large data sets every day, for example to determine the number of distinct search queries on `google.com` over a time period. Such queries represent a hard challenge in terms of computational resources, and memory in particular: For the PowerDrill system, a non-negligible fraction of queries historically could not be computed because they exceeded the available memory.

In this paper we present a series of improvements to the HYPERLOGLOG algorithm by Flajolet et al. [7] that estimates cardinalities efficiently. Our improvements decrease memory usage as well as increase the accuracy of the es-

timate significantly for a range of important cardinalities. We evaluate all improvements empirically and compare with the HYPERLOGLOG algorithm from [7]. Our changes to the algorithm are generally applicable and not specific to our system. Like HYPERLOGLOG, our proposed improved algorithm parallelizes perfectly and computes the cardinality estimate in a single pass.

Outline. The remainder of this paper is organized as follows; we first justify our algorithm choice and summarize related work in Section 2. In Section 3 we give background information on our practical use cases at Google and list the requirements for a cardinality estimation algorithm in this context. In Section 4 we present the HYPERLOGLOG algorithm from [7] that is the basis of our improved algorithm. In Section 5 we describe the improvements we made to the algorithm, as well as evaluate each of them empirically. Section 6 explains advantages of HYPERLOGLOG for dictionary encodings in column stores. Finally, we conclude in Section 7.

2. RELATED WORK AND ALGORITHM CHOICE

Starting with work by Flajolet and Martin [6] a lot of research has been devoted to the cardinality estimation problem. See [3, 14] for an overview and a categorization of available algorithms. There are two popular models to describe and analyse cardinality estimation algorithms: Firstly, the data streaming (ϵ, δ) -model [10, 11] analyses the necessary space to get a $(1 \pm \epsilon)$ -approximation with a fixed success probability of δ , for example $\delta = 2/3$, for cardinalities in $\{1, \dots, n\}$. Secondly it is possible to analyse the relative accuracy defined as the standard error of the estimator [3].

The algorithm previously implemented at Google [13, 15, 9] was MINCOUNT, which is presented as *algorithm one* in [2]. It has an accuracy of $1.0/\sqrt{m}$, where m is the maximum number of hash values maintained (and thus linear in the required memory). In the (ϵ, δ) -model, this algorithm needs $O(\epsilon^{-2} \log n)$ space, and it is near exact for cardinalities up to m (modulo hash collisions). See [8] for the statistical analysis and suggestions for making the algorithm more computationally efficient.

The algorithm presented by Kane et al. [11] meets the lower bound on space of $\Omega(\epsilon^{-2} \log n)$ in the (ϵ, δ) -model [10] and is optimal in that sense. However, the algorithm is complex

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
EDBT/ICDT '13 March 18 - 22 2013, Genoa, Italy
Copyright 2013 ACM 978-1-4503-1597-5/13/03 ...\$15.00

<https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/40671.pdf>

Streaming model



Goal: estimate # of distinct elements in the stream

m = length of stream, n = # elem, assume $m = \text{poly}(n)$

Moments

f_i = frequency of element i $i=1, \dots, n$

k th Moment: $F_k = \sum_{i=1}^n f_i^k$

e.g. $F_0 = \sum_{i=1}^n f_i^0 = |\{i: f_i > 0\}|$ = distinct elements

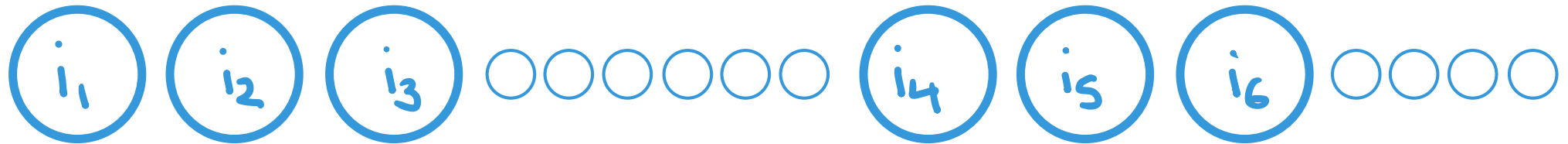
$F_1 = \sum_{i=1}^n f_i$ = total length

$F_2 = \sum_{i=1}^n f_i^2$ = "repeat rate"

today

F_k (for $k \geq 2$) measures skew of data

Streaming model



Algorithm w/
limited space

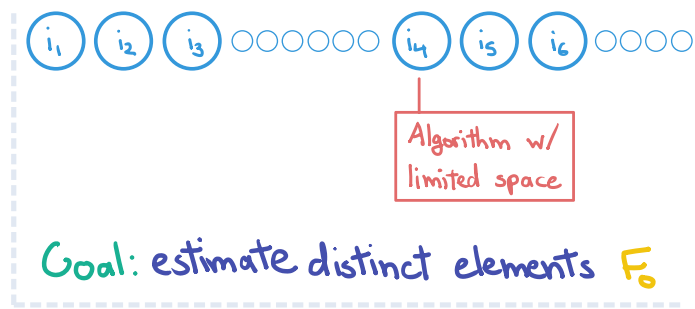
Goal: estimate # of distinct elements in the stream

ideas?

"distinct elements sampler"

Let B independently sample each distinct element w/ prob p (Later: how to implement B)

$$E[|B|] = p F_0 \quad \left(\frac{|B|}{p} \text{ unbiased estimator of } F_0 \right)$$



Concentration phenomena

let $X_1, \dots, X_n \in [0, 1]$ be independent

$$\mu = E[X_1 + \dots + X_n]$$

Q: probability that

$X_1 + \dots + X_n$ is close to μ ?
(multiplicatively)

A: error probability exponentially small in μ

Multiplicative Chernoff bound

let $X_1, \dots, X_n \in [0, 1]$ be independent

$$\mu = E[X_1 + \dots + X_n]$$

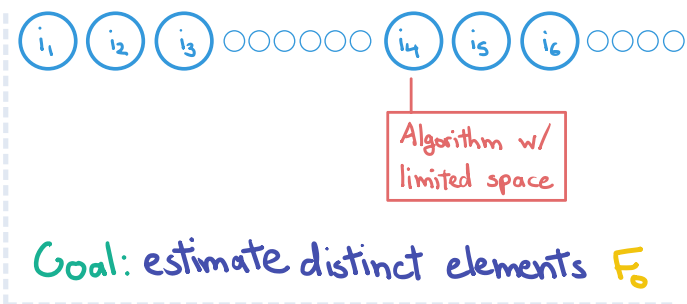
$$\epsilon \in (0, 1)$$

$$\bullet \quad P[X_1 + \dots + X_n \geq (1 + \epsilon)\mu] \leq e^{-\epsilon^2 \mu / 3}$$

$$\bullet \quad P[X_1 + \dots + X_n \leq (1 - \epsilon)\mu] \leq e^{-\epsilon^2 \mu / 2}$$

(proof at end if time permits)

"distinct elements sampler"



Let B independently sample each distinct element w/ prob p (Later: how to implement B)

$$E[|B|] = p F_0 \quad \left(\frac{|B|}{p} \text{ unbiased estimator of } F_0 \right)$$

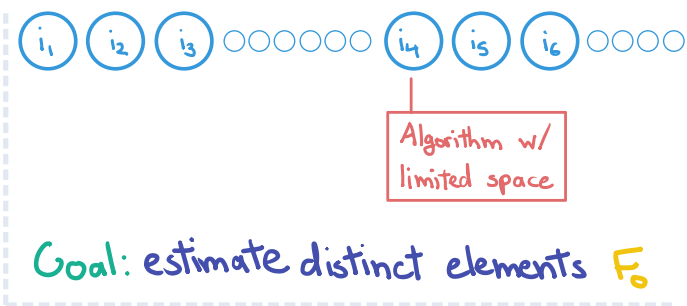
$$|B| = \sum x_i \quad \text{w/ } x_i = \begin{cases} 1 & i \text{ in } B \\ 0 & \text{otherwise} \end{cases}$$

$$Pr[|B| \geq (1+\epsilon)pF_0] \leq e^{-\Omega(\epsilon^2 pF_0)}$$

(same for $|B| \leq (1-\epsilon)pF_0$)

$$\begin{aligned} X_1, m, X_n &\in [0,1] \text{ be independent} \\ \mu &= E[X_1 + m + X_n] \quad \epsilon \in (0,1) \\ P[X_1 + m + X_n &\geq (1+\epsilon)\mu] \leq e^{-\epsilon^2 \mu / 3} \\ P[X_1 + m + X_n &\leq (1-\epsilon)\mu] \leq e^{-\epsilon^2 \mu / 2} \end{aligned}$$

"distinct elements sampler"



Let B independently sample each distinct element w/ prob p (Later: how to implement B)

$$P_r[|B| \geq (1+\epsilon)pF_0] \leq e^{-\Omega(\epsilon^2 pF_0)}$$

(or $|B| \leq (1-\epsilon)pF_0$)

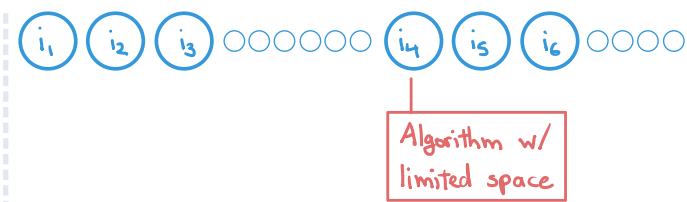
$$e^{-\Omega(\epsilon^2 pF_0)} \leq \frac{1}{\text{poly}(n)} \quad \text{for } pF_0 \geq \Omega\left(\frac{\log n}{\epsilon^2}\right)$$

Ideally: $\frac{T}{2} \leq pF_0 \leq T$

for $T = \frac{C \log n}{\epsilon^2}$, C large

$X_1, \dots, X_n \in [0,1]$ be independent
 $\mu = E[X_1 + \dots + X_n] \in (0,1)$
 $P[X_1 + \dots + X_n \geq (1+\epsilon)\mu] \leq e^{-\epsilon^2 \mu/3}$
 $P[X_1 + \dots + X_n \leq (1-\epsilon)\mu] \leq e^{-\epsilon^2 \mu/2}$

"distinct elements sampler"



Goal: estimate distinct elements F_0

Let B independently sample each distinct element w/ prob p (Later: how to implement B)

$\frac{|B|}{p}$ unbiased estimator

Ideally: $pF_0 = \Theta\left(\frac{\log n}{\epsilon^2}\right)$

$$E[|B|] = pF_0$$

$$|B| = \sum X_i \quad X_i = \begin{cases} 1 & i \text{ in } B \\ 0 & \text{otherwise} \end{cases}$$

$$\Pr[|B| \geq (1+\epsilon)pF_0] \leq e^{-\Omega(\epsilon^2 pF_0)} \\ (\text{or } |B| \leq (1-\epsilon)pF_0)$$

Q1: How to implement B in a stream?
(for fixed p)

Q2: How to "guess" right value of p ?
(w/out knowing $F_0 \dots$)

"distinct elements sampler"

Let B independently sample each distinct element w/ prob p



Algorithm w/
limited space

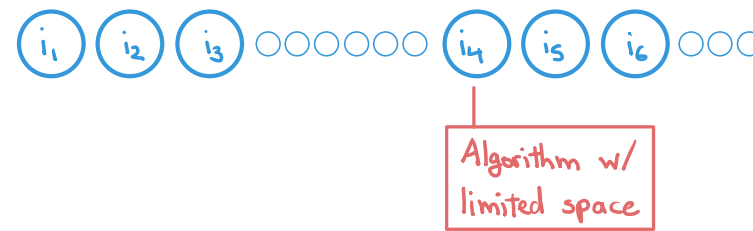
Goal: estimate distinct elements F_0

Q1: How to implement B in a stream?
(for fixed p)

ideas?

"distinct elements sampler"

Let B independently sample each distinct element w/ prob p



Goal: estimate distinct elements F_0

Q1: How to implement B in a stream?
(for fixed p)

1. $B \leftarrow \emptyset$

2. For each e_i (in the stream)

a. If $e_i \in B$ then remove e_i from B

b. Add e_i to B w/ prob p

3. Return $|B|/p$

B samples last instance of each element w/ prob p !

"distinct elements sampler"

Let B independently sample each distinct element w/ prob p

1. $B \leftarrow \emptyset$

2. For each e_i (in the stream)

a. If $e_i \in B$ then $B \leftarrow B - e_i$

b. $B \leftarrow B + e_i$ w/ prob p

Q2: How to "guess" right value of p ?

(w/out knowing $F_0 \dots$)

$$\text{Ideally: } pF_0 = \Theta\left(\frac{\log n}{\epsilon^2}\right)$$

"distinct elements sampler"

Let B independently sample each distinct element w/ prob p

1. $B \leftarrow \emptyset$

2. For each e_i (in the stream)

a. If $e_i \in B$ then $B \leftarrow B - e_i$

b. $B \leftarrow B + e_i$ w/ prob p

Q2: How to "guess" right value of p ? Ideally: $pF_0 = \Theta(\frac{\log n}{\epsilon^2})$

Guess in parallel:

B_0 samples w/ prob 1

B_1 samples w/ prob $\frac{1}{2}$

B_2 samples w/ prob $\frac{1}{4}$

$\vdots$

B_i samples w/ prob $\frac{1}{2^i}$ (up to $i = \log n$)

what if B_i gets very big?

which B_i to use to estimate?

"distinct elements sampler"

Let B independently sample each distinct element w/ prob p

1. $B \leftarrow \emptyset$
2. For each e_i (in the stream)
 - a. If $e_i \in B$ then $B \leftarrow B - e_i$
 - b. $B \leftarrow B + e_i$ w/ prob p

Q2: How to "guess" right value of p ? Ideally: $pF_0 = \Theta(\frac{\log n}{\epsilon^2})$

Guess in parallel:

B_0 samples w/ prob $p_1 = 1$

B_1 samples w/ prob $p_2 = \frac{1}{2}$

$\vdots$

B_i samples w/ prob $p_i = \frac{1}{2^i}$

$\vdots$

$B_{10}, \dots$

let $T = \frac{C \log n}{\epsilon^2}$ for large C s.t.

if $p_i F_0 \geq T/4$, then

$$P[|B_i - p_i F_0| \geq \epsilon p_i F_0] \leq \frac{1}{\text{poly}(n)}$$

if $|B_i| \geq T$, then:

- p_i too big
- p_{i+1} big enough

$\Rightarrow$ Discard B_i

"distinct elements sampler"

Let B independently sample each distinct element w/ prob p

1. $B \leftarrow \emptyset$
2. For each e_i (in the stream)
 - a. If $e_i \in B$ then $B \leftarrow B - e_i$
 - b. $B \leftarrow B + e_i$ w/ prob p

Q2: How to "guess" right value of p ? Ideally: $pF_0 = \Theta(\frac{\log n}{\epsilon^2})$

1. $B \leftarrow \emptyset$, $p \leftarrow 1$, $T = \frac{C \log n}{\epsilon^2}$ for large C

2. For each e_i (in the stream)

a. If $e_i \in B$ then remove e_i from B

b. Add e_i to B w/ prob p

c. If $|B| \geq T$:

a. $p \leftarrow p/2$

b. drop each $e \in B$ indep. w/ prob $1/2$

w/ high probability
($1 \pm \epsilon$)-APX in
 $O(\frac{\log n}{\epsilon^2})$ space

Today

Probability (& techniques)

Algorithms

Continuous
Random Variables

Variance

Variance Reduction

Distinct Elements

F_2 estimation

Moments

f_i = frequency of element i $i=1, \dots, n$

k th Moment: $F_k = \sum_{i=1}^n f_i^k$

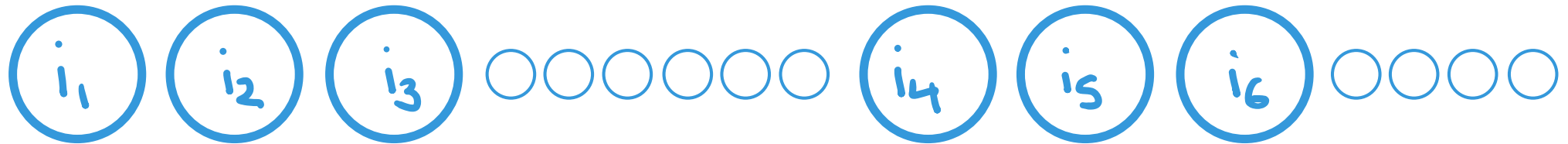
e.g. $F_0 = \sum_{i=1}^n f_i^0 = |\{i: f_i > 0\}|$ = distinct elements

$F_1 = \sum_{i=1}^n f_i$ = total length

$F_2 = \sum_{i=1}^n f_i^2$ = "repeat rate"

F_k (for $k \geq 2$) measures skew of data

Streaming model



Algorithm w/
limited space

Goal: estimate

$$F_2 = \sum_{i=1}^n f_i^2$$

m = length of stream, n = # elem, assume $m = \text{poly}(n)$

Ideas?

Randomly signed sums

let $h: [n] \rightarrow \{-1, 1\}$ be 4-wise indep.

$$F_2 = \sum_{i=1}^n f_i^2$$

Estimator: Y^2 w/ $Y = \sum_{i=1}^n h(i) f_i$

$$E[Y^2] = F_2$$

Q: How often is Y^2 "close" to F_2 ?

Lecture 7

Probability (& techniques)

Algorithms

Continuous
Random Variables

Variance

Variance Reduction

Distinct Elements

Variance

$$\text{Var}(X) = E[(X - E[X])^2] = E[X^2] - (E[X])^2$$

Chebyshev's inequality

$$P[|X - E[X]| \geq \alpha] \leq \frac{\text{Var}(X)}{\alpha^2}$$

proof

Randomly signed sums

let $h: [n] \rightarrow \{-1, 1\}$ be 4-wise indep.

$$F_2 = \sum_{i=1}^n f_i^2$$

Estimator: Y^2 w/ $Y = \sum_{i=1}^n h(i) f_i$

$$\bullet E[Y^2] = F_2 \quad \bullet E[Y^4] \leq 3F_2^2$$

$$\bullet \text{var}(Y^2) = E[Y^4] - E[Y^2]^2 \leq 2F_2^2$$

$$\text{Chebyshev's} \Rightarrow P[|Y^2 - F_2| \geq \epsilon F_2] \leq \frac{2F_2^2}{\epsilon^2 F_2^2} = \frac{2}{\epsilon^2}$$

$\Rightarrow O(1)$ -APX w/ constant probability

Better?

$$\text{Var}(X) = E[(X - E[X])^2] = E[X^2] - (E[X])^2$$

$$\text{Chebyshev's } P[|X - E[X]| \geq \alpha] \leq \frac{\text{Var}(X)}{\alpha^2}$$

$$\text{Var}(X) = E[(X - E[X])^2] = E[X^2] - (E[X])^2$$

Q: how to reduce $\text{Var}(X)$ (in general)?

averaging

let $X_1, \dots, X_\ell$ be indep. copies of X

$$\bar{X} = \frac{1}{\ell}(X_1 + \dots + X_\ell)$$

$$\text{Var}(\bar{X}) = \frac{1}{\ell^2} E[(X_1 + \dots + X_\ell - E[X_1] - \dots - E[X_\ell])^2]$$

$$= \frac{1}{\ell^2} \sum_{i,j} E[(X_i - E[X_i])(X_j - E[X_j])]$$

$$= \frac{1}{\ell} \text{Var}(X)$$

0 if $i \neq j$

let $h: [n] \rightarrow \{-1, 1\}$ be 4-wise indep

$$F_2 = \sum_{i=1}^n f_i^2$$

Estimator: Y^2 w/ $Y = \sum_{i=1}^n h(i) f_i$

$$\bullet E[Y^2] = F_2 \quad \bullet \text{var}(Y^2) \leq 2F_2^2$$

$$Z = \frac{1}{\ell} (Y_1^2 + \dots + Y_\ell^2) \quad \text{w/ } Y_i \sim Y$$

$$\text{Chebyshev's} \Rightarrow P[|Z^2 - F_2| \geq \epsilon F_2] \leq \frac{2}{\ell \epsilon^2}$$

$$\ell = O\left(\frac{1}{\epsilon^2}\right) \Rightarrow (1+\epsilon)\text{-APX w/ constant probability}$$

what about prob $1-\delta$?

what is $\delta = \frac{1}{\text{poly}(n)}$?

$$\text{Var}(X) = E[(X - E[X])^2] = E[X^2] - (E[X])^2$$

$$\text{Chebyshev's } P[|X - E[X]| \geq \alpha] \leq \frac{\text{Var}(X)}{\alpha^2}$$

Concentration phenomena

let $X_1, \dots, X_n \in [0, 1]$ be independent

$$\mu = E[X_1 + \dots + X_n]$$

Q: probability that

$X_1 + \dots + X_n$ is close to μ ?
(multiplicatively)

A: error probability exponentially small in μ

Multiplicative Chernoff bound

let $X_1, \dots, X_n \in [0, 1]$ be independent

$$\mu = E[X_1 + \dots + X_n]$$

$$\epsilon \in (0, 1)$$

$$\bullet \quad P[X_1 + \dots + X_n \geq (1 + \epsilon)\mu] \leq e^{-\epsilon^2 \mu / 3}$$

$$\bullet \quad P[X_1 + \dots + X_n \leq (1 - \epsilon)\mu] \leq e^{-\epsilon^2 \mu / 2}$$

(proof at end if time permits)

$$Z = \frac{1}{2} (Y_1^2 + \dots + Y_\ell^2)$$

$$P[|Z - F_2| \geq \epsilon F_2] \leq \frac{2}{\ell \epsilon^2}$$

$$\ell = O\left(\frac{1}{\epsilon^2}\right) \Rightarrow (1+\epsilon)\text{-APX w/ prob } \frac{3}{4}$$

$$F_2 = \sum_{i=1}^n f_i^2$$

$$Y_j = \sum_{i=1}^n h_j(i) f_i$$

$$E[Y_j^2] = F_2$$

$$\text{var}(Y_j^2) \leq 2F_2^2$$

$$\text{median}(Z_1, \dots, Z_q)$$

$$\text{for } q = O(\ln(1/\delta))$$

$$Z_i \sim Z, \text{ indep}$$

why?

median wrong iff
half of the Z_i 's wrong!

$$\text{Chebyshev's } P[|X - E[X]| \geq d] \leq \frac{\text{Var}(X)}{d^2}$$

$$X_1, \dots, X_n \in [0,1] \text{ indep. } \mu = E[X_1 + \dots + X_n]$$

$$P[X_1 + \dots + X_n \geq (1+\epsilon)\mu] \leq e^{-\epsilon^2 \mu / 3}$$

$$P[X_1 + \dots + X_n \leq (1-\epsilon)\mu] \leq e^{-\epsilon^2 \mu / 2}$$

All put together

$(1 \pm \epsilon)$ -APX F_2 w/ prob $\geq 1 - \delta$ in $O(\frac{\ln(1/\delta)}{\epsilon^2})$ space

"median trick", "median of means"

- ① take mean to reduce variance,
get constant prob. of error
- ② take median to amplify success prob.
exponentially

Optimal Sub-Gaussian Mean Estimation in $\mathbb{R}$

Jasper C.H. Lee

University of Wisconsin-Madison & Brown University
jasper.lee@wisc.edu

Paul Valiant

IAS & Purdue University
pvaliant@gmail.com

Abstract—We settle the fundamental problem of estimating the mean of a real-valued distribution in the high probability regime, under the minimal (and essentially necessary) assumption that the distribution has finite but unknown variance: we propose an estimator with convergence tight up to a $1 + o(1)$ factor. Crucially, in contrast to prior works, our estimator does not require prior knowledge of the variance, and works across the entire gamut of distributions with finite variance, including those without any higher moments. Parameterized by the sample size n , the failure probability δ , and the variance σ^2 , our estimator has additive accuracy within $\sigma \cdot (1 + o(1)) \sqrt{\frac{2 \log \frac{1}{\delta}}{n}}$, which is optimal up to the $1 + o(1)$ term. This asymptotically matches the convergence of the sample mean for the Gaussian distribution with the same variance. Our estimator construction and analysis gives a framework generalizable to other problems, tightly analyzing a sum of dependent random variables by viewing the sum implicitly as a 2-parameter ψ -estimator, and constructing bounds using mathematical programming and duality techniques.

I. INTRODUCTION

We revisit one of the most fundamental problems in statistics: estimating the mean of a real-valued distribution, using as few independent samples from it as possible. Our proposed estimator has convergence that is optimal not only in a big-O sense (i.e. “up to multiplicative constants”), but tight to a $1 + o(1)$ factor, under the minimal (and essentially necessary, see below) assumption of the finiteness of the variance. Previous works, discussed further in Section II, are either only big-O tight [12, 22, 1], or require additional strong assumptions such as the variance being known to the estimator [4] or assumptions that allow for accurate estimates of the variance, such as knowledge of an upper bound on the kurtosis (normalized fourth moment) [4, 7].

A. The Model and Main Result

Given a set of i.i.d. samples from a real-valued distribution, the goal is to return, with extremely high probability, an accurate estimate of the distribution’s mean. Specifically, given a sample set X of size n consisting of independent draws from a real-valued distribution D , an (ϵ, δ) -estimator of the mean is a function $\hat{\mu} : \mathbb{R}^n \rightarrow \mathbb{R}$ such that, except with failure

probability $\leq \delta$, the estimate $\hat{\mu}(X)$ is within ϵ of the true mean $\mu(D)$. Namely,

$$\mathbb{P}(|\hat{\mu}(X) - \mu(D)| \leq \epsilon) \geq 1 - \delta \quad (1)$$

The goal is to find the optimal tradeoff between the sample size n , and the error parameters ϵ and δ , for the distribution D . Fixing any two of the three parameters and minimizing the third yields essentially equivalent reformulations of the problem: we can fix ϵ, δ and minimize the *sample complexity* n ; we can fix δ, n and minimize the *error* ϵ ; or we can fix ϵ, n and minimize the *failure probability* δ (maximizing the *robustness* $1 - \delta$).

Perhaps the most standard and well-behaved setting for mean estimation is when the distribution D is a Gaussian. The sample mean (the empirical mean) is a provably optimal estimator in our sense when D is Gaussian: for any $\epsilon, \delta > 0$, the sample mean $\mu(X)$ is an (ϵ, δ) -estimator when given a sample set of size $n = 2 \cdot \frac{\sigma^2(D) \cdot \log \frac{1}{\delta}}{\epsilon^2}$ (all logarithms will be base e); and there is *no* (ϵ, δ) -estimator for Gaussians if the constant 2 in the previous expression for the sample size is changed to any smaller number.

The main result of this paper is an estimator (Section III) that performs as well, on *any* distribution with finite variance, as the sample mean does on a Gaussian, without knowledge of the distribution or its variance:

Theorem 1. *Estimator 1, given $\delta, n > 0$, and a sample set X of n independent samples from distribution D , will, with probability at least $1 - \delta$ over the sampling process, yield an estimate $\hat{\mu}$ with error at most*

$$|\hat{\mu}(X) - \mu(D)| \leq \sigma(D) \cdot (1 + o(1)) \sqrt{\frac{2 \log \frac{1}{\delta}}{n}}$$

Here, the $o(1)$ term tends to 0 as $(\frac{\log \frac{1}{\delta}}{n}, \delta) \rightarrow (0, 0)$, at a rate that is independent of D .

Furthermore, as evidenced by the Gaussian case, there is no estimator which, under the same settings, produces an error that improves on our guarantees by more than a $1 + o(1)$ multiplicative factor.

We have parameterized the above theorem in terms of fixing the sample size n and the robustness param-