香港城市大學
City University of Hong Kong
專業 創新 胸懷全球
Professional·Creative
For The World

Data Anonymization and Differential Privacy

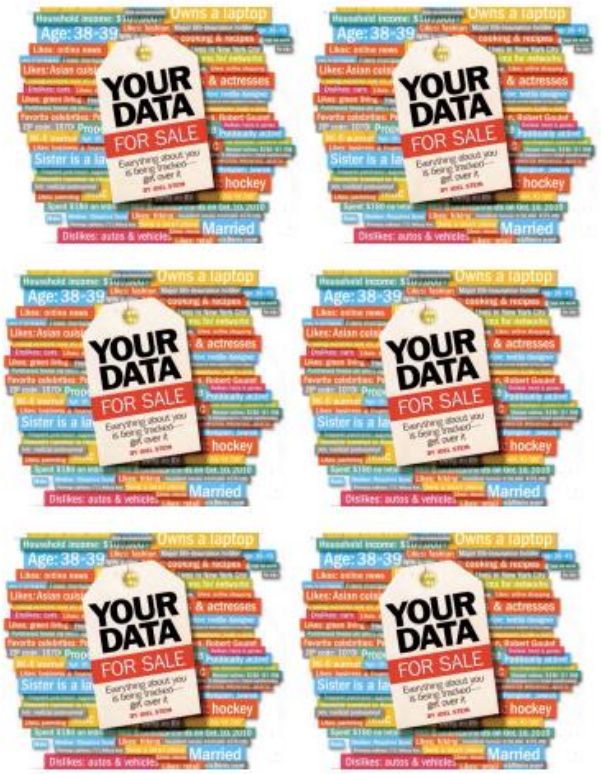
Prof. Cong Wang
Department of Computer Science
City University of Hong Kong

Slides credits in part from Graham Cormode, Vitaly Shmatikov, Ashwin Machanavajjhala, Michael Hay, Xi He, Tianhao Wang, Zhan QIN, etc.

This “Concise” Lecture

- Will NOT make you a differential privacy expert immediately.
- Will provide background and inspire you to be a differential privacy expert in the future.

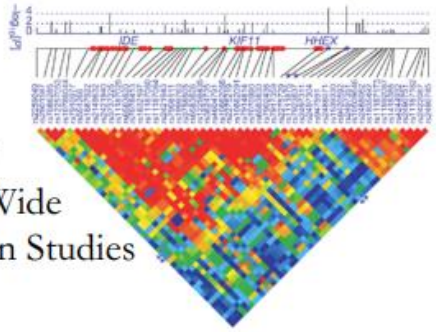
Aggregated Personal Data is **Invaluable**



Advertising

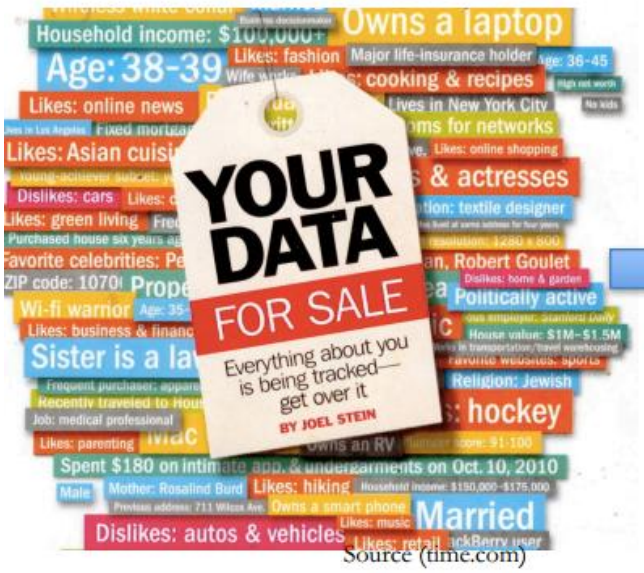
Source (esri.com)

Genome Wide Association Studies



Human Mobility analysis

Personal Data is … Well … Personal!



Source (time.com)

Age
Income
Address
Likes/Dislikes
Sexual Orientation
Medical History

Redlining

Discrimination

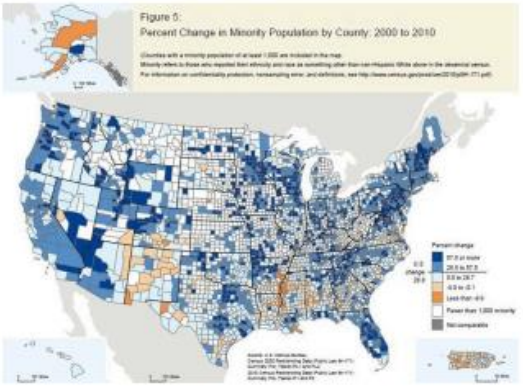
Physical/Financial Harm

Aggregated Personal Data

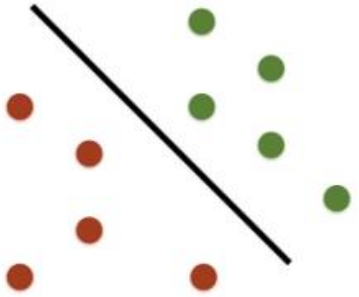
- … is made publicly available in many forms.

De-identified records
(e.g., medical)

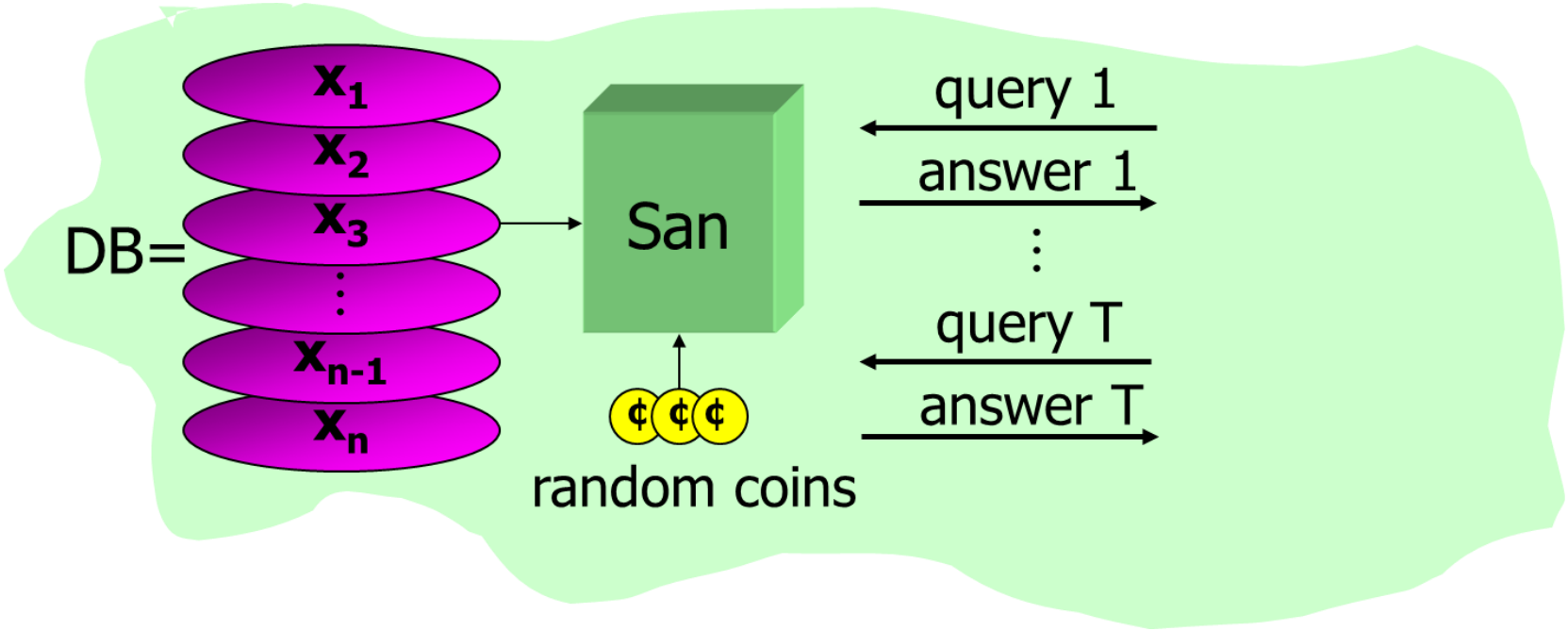
Statistics
(e.g., demographic)



Predictive models
(e.g., advertising)



Basic Application Setting



Data “Anonymization”

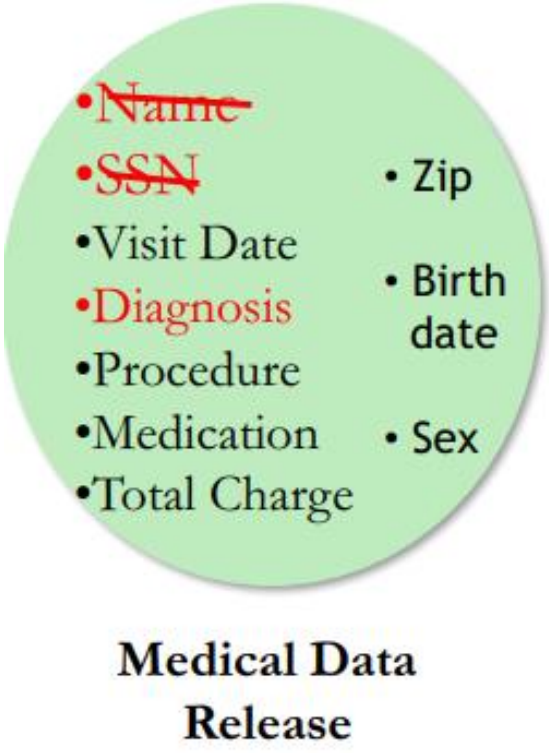
- How?

- Remove “personally identifying information” (PII)
  - Name, Social Security number, phone number, email, address… what else?

- Problem: PII has no technical meaning
  - In privacy breaches, any information can be personally identifying
    - Examples: AOL dataset, Netflix Prize dataset

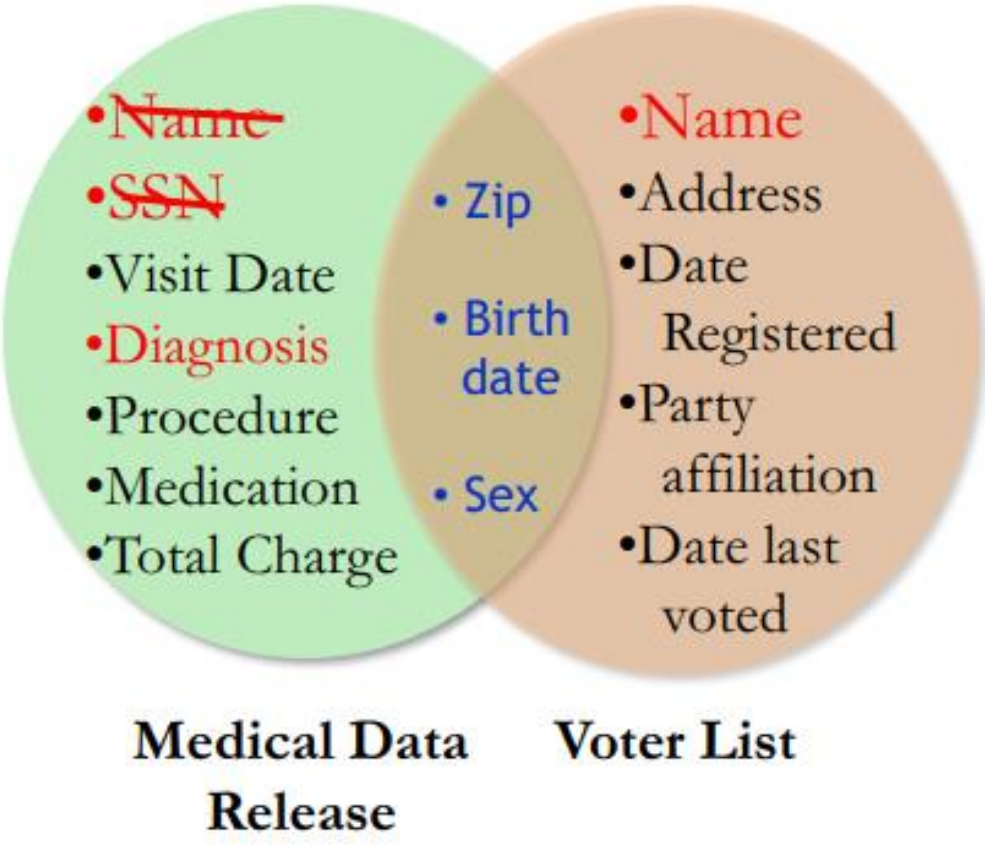
The Massachusetts Governor Privacy Breach

[Sweeney, 2010]

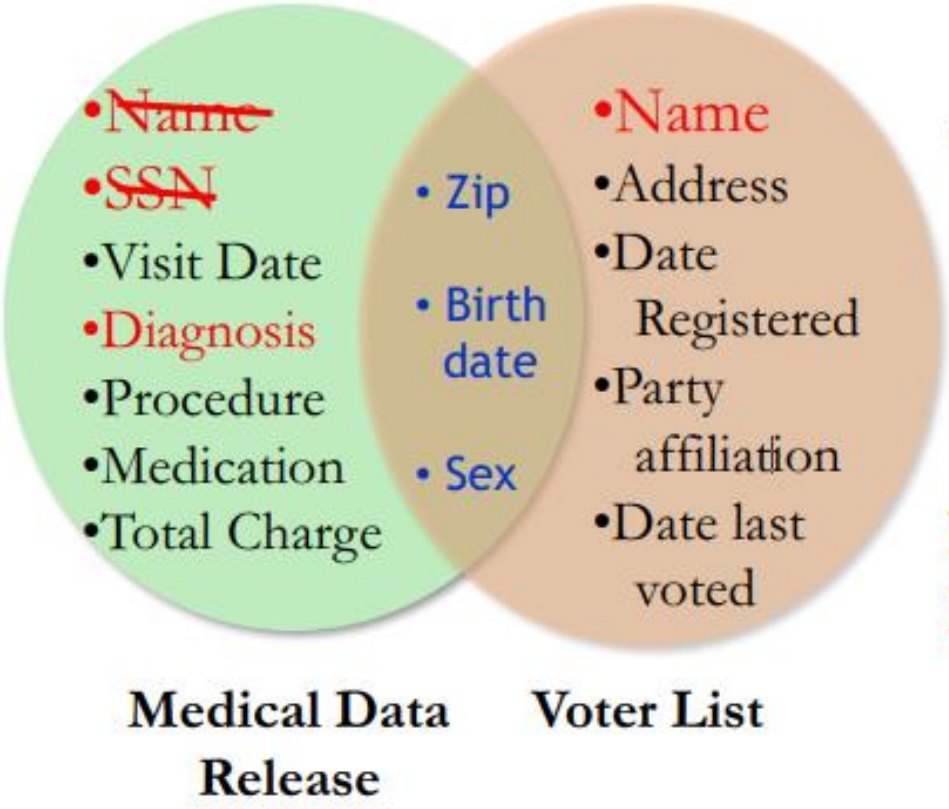


The Massachusetts Governor Privacy Breach

[Sweeney, 2010]



Linkage Attack



[Sweeney, 2010]

- Governor of MA **uniquely identified** using ZipCode, Birth Date, and Sex.

Name linked to Diagnosis

Observation #1: Dataset Joins

- Attacker learns sensitive data by joining two datasets on common attributes
  - Anonymized dataset with sensitive attributes
    - Example: age, race, symptoms
  - "Harmless" dataset with individual identifiers
    - Example: name, address, age, race

- Demographic attributes (age, ZIP code, race, etc.) are common in datasets with information about individuals

Observation #2: Quasi-Identifiers

- Sweeney's observation: (birthdate, ZIP code, gender) uniquely identifies 87% of US population
  - Side note: actually, only 63% [Golle, WPES '06]

- Publishing a record with a quasi-identifier is as bad as publishing it with an explicit identity

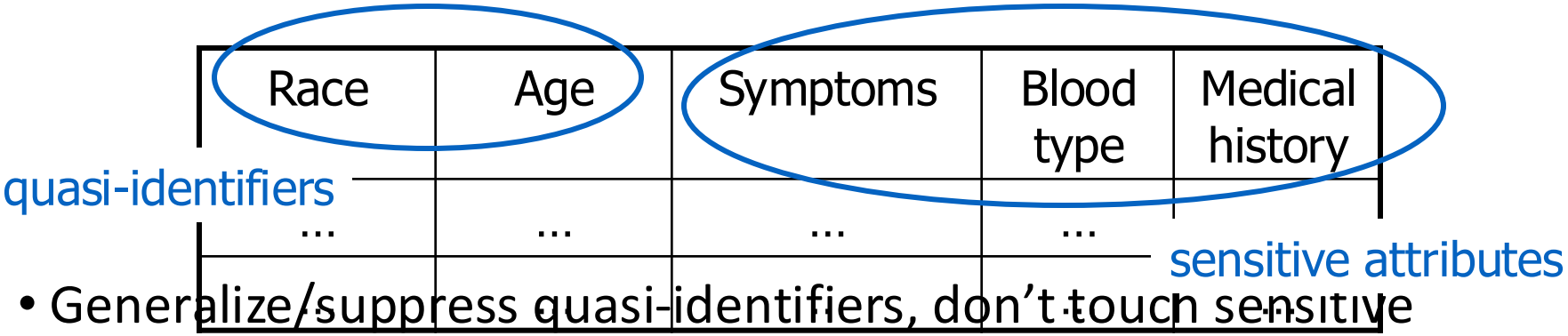
- Eliminating quasi-identifiers is not desirable
  - For example, users of the dataset may want to study distribution of diseases by age and ZIP code

k-Anonymity

- Proposed by Samarati and Sweeney (1998)
- Hundreds of papers since then
  - Extremely popular in the database and data mining communities (SIGMOD, ICDE, KDD, VLDB)
- Most based on generalization and suppression

Anonymization in a Nutshell

- Dataset is a relational table
- Attributes (columns) are divided into quasi-identifiers and sensitive attributes



- Generalize/suppress quasi-identifiers, don't touch sensitive attributes (keep them "truthful")

k-Anonymity: Definition

- Each (transformed) quasi-identifier group must appear in at least k records in the anonymized dataset
  - k is chosen by the data owner (how?)
  - Example: any age-race combination from original DB must appear at least 10 times in anonymized DB

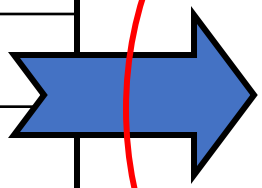
- Guarantees that any join on quasi-identifiers with the anonymized dataset will contain at least k records for each quasi-identifier

Achieving k-Anonymity

- Generalization
  - Individual values of attributes replaced by broader category
    - Area code instead of phone number: 3442 8765 -- >> 3442 xxxx
    - Value “23” of the age attribute is replaced by 20<Age<=30
- Suppression
  - Replace certain values of the attributes by an asterisk '*’.
    - Example: replace all the values in the 'Name' attribute with a ‘*’.
- Lots of algorithms in the literature
  - Aim to produce “useful” anonymizations, though … usually without any clear notion of utility

Example: 3-Anonymity

Caucas	78712	Flu
Asian	78705	Shingles
Caucas	78754	Flu
Asian	78705	Acne
AfrAm	78705	Acne
Caucas	78705	Flu



Caucas	787XX	Flu
Asian/AfrAm	78705	Shingles
Caucas	787XX	Flu
Asian/AfrAm	78705	Acne
Asian/AfrAm	78705	Acne
Caucas	787XX	Flu

This is 3-anonymous, right?

Problem of k-anonymity

When joining with external database ,
adversary learns Rusty Shackleford has Flu

…	…	…
Rusty Shackleford	Caucas	78705
…	…	…

+

Caucas	787XX	Flu
Asian/AfrAm	78705	Shingles
Caucas	787XX	Flu
Asian/AfrAm	78705	Acne
Asian/AfrAm	78705	Acne
Caucas	787XX	Flu

Problem: sensitive attributes are not "diverse"
within each quasi-identifier group

Other Attempts

- *L*-diversity
  - Entropy of sensitive attributes within each quasi-identifier group must be at least *L*
- *t*-closeness
  - Distribution of sensitive attributes within each quasi-identifier group should be “close” to their distribution in the entire original database

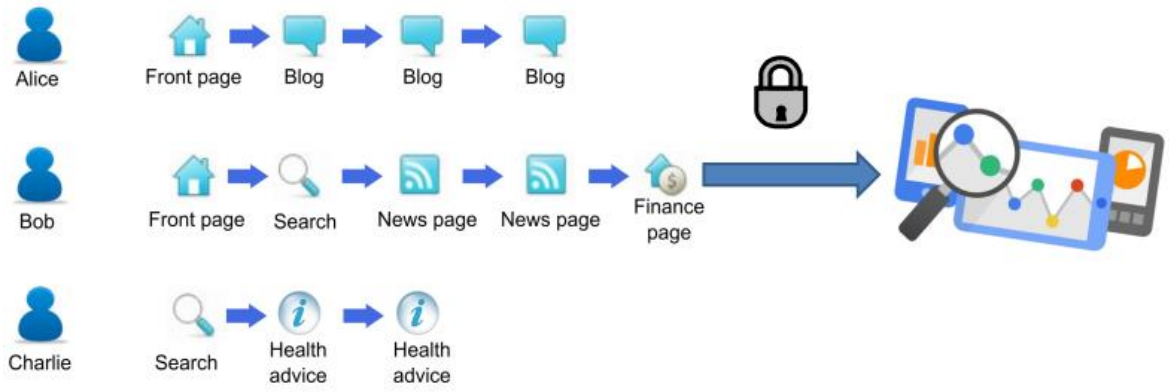
Issues with Syntactic Definitions

- What adversary do they apply to?
  - Do not consider adversaries with side information
  - Do not consider probability
  - Do not consider adversarial algorithms for making decisions (inference)

- Any attribute is a potential quasi-identifier
  - External / auxiliary / background information about people is very easy to obtain

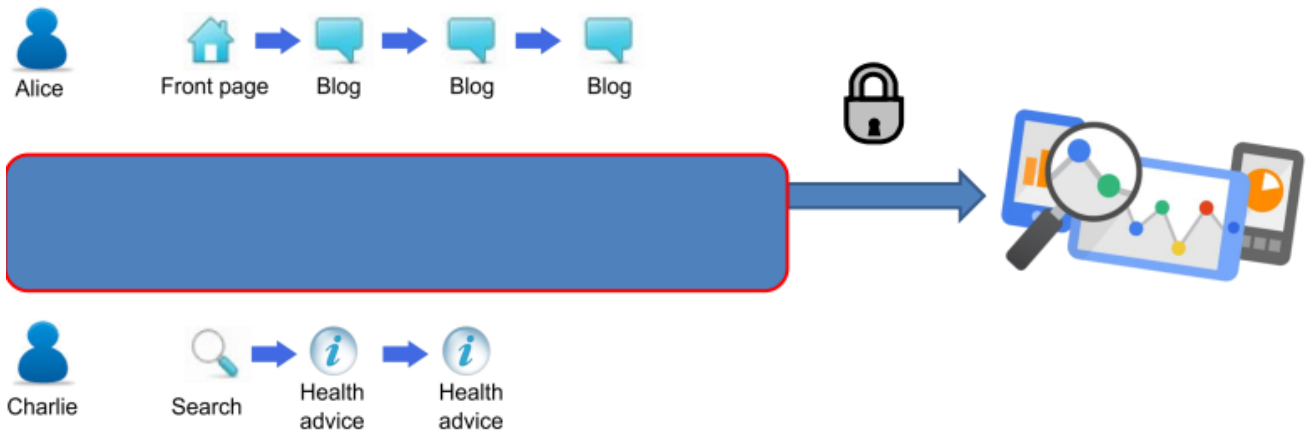
Differential Privacy

- Statistical outcome is indistinguishable regardless of whether a particular user record is in the data or not
  - "Whatever is learned would be learned regardless of whether or not you participate"

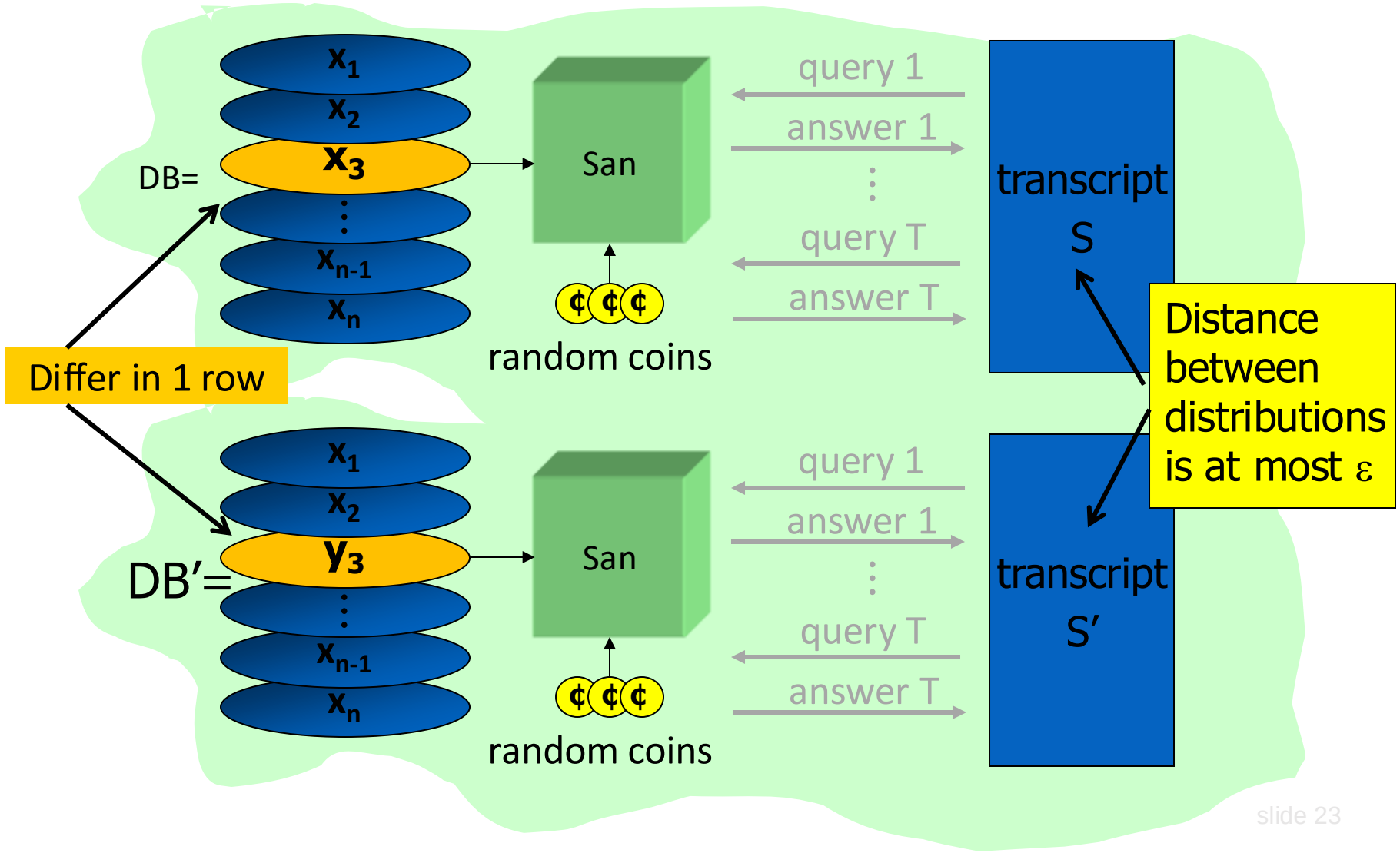


Differential Privacy

- Statistical outcome is indistinguishable regardless of whether a particular user record is in the data or not
  - "Whatever is learned would be learned regardless of whether or not you participate"

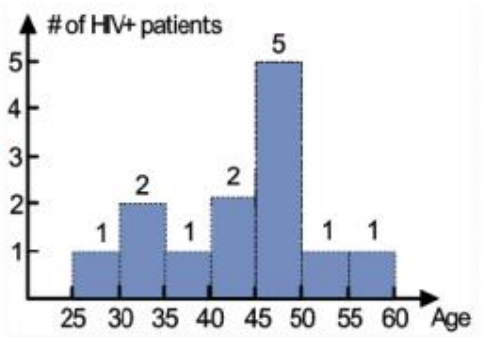


Indistinguishability

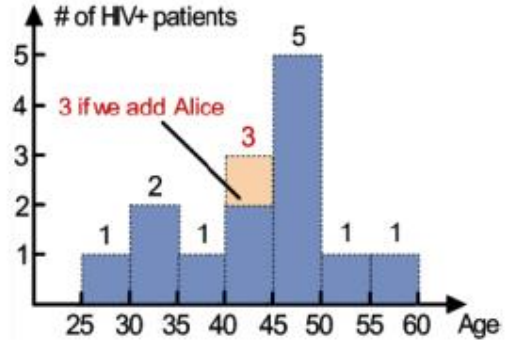


An Example: Statistical Data Release

Name	Age	HIV+
Frank	42	Y
Bob	31	Y
Mary	28	Y
Dave	43	N
…	…	…



Name	Age	HIV+
Alice	43	Y
Frank	42	Y
Bob	31	Y
Mary	28	Y
Dave	43	N
…	…	…

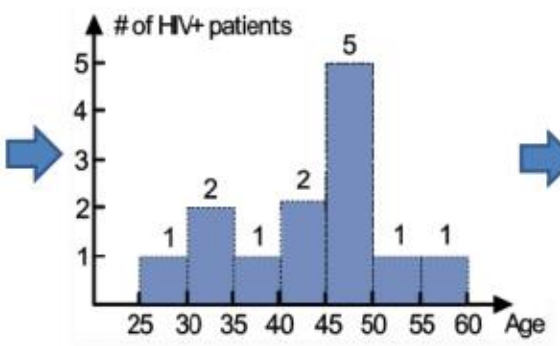


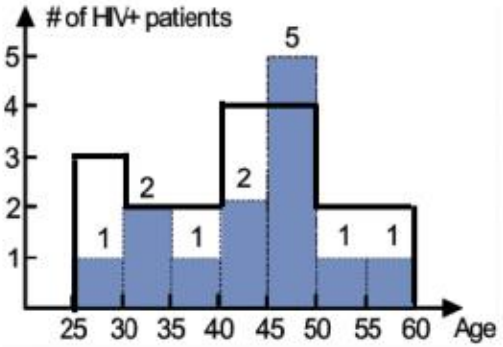
Original records

Original histogram

An Example: Statistical Data Release

Name	Age	HIV+
Frank	42	Y
Bob	31	Y
Mary	28	Y
Dave	43	N
…	…	…



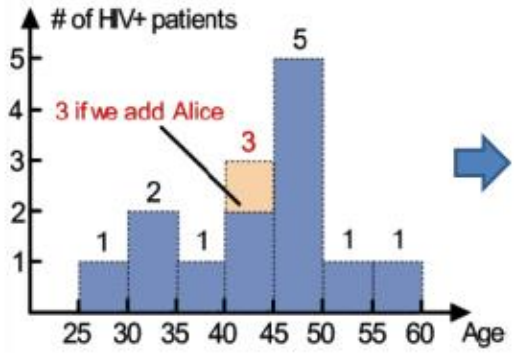


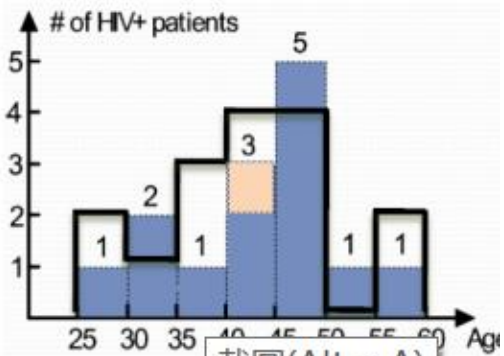
Original records

Original histogram

Perturbed histogram with differential privacy

Name	Age	HIV+
Alice	43	Y
Frank	42	Y
Bob	31	Y
Mary	28	Y
Dave	43	N
…	…	…





Formalizing Indistinguishability

[Dwork, ICALP'06]

For every pair of neighboring databases that differ in only one record

For every output

D_1 D_2

O

If algorithm A satisfies differential privacy then

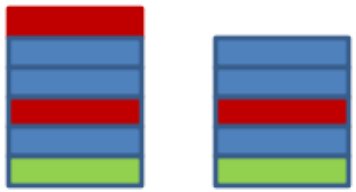
$$\frac{\Pr[A(D_1) = O]}{\Pr[A(D_2) = O]} < \exp(\varepsilon) \quad (\varepsilon > 0)$$

Intuition: adversary should not be able to use output O to distinguish between any D_1 and D_2

Why Pair of Databases that Differ in One Row

For every pair of neighboring databases that differ in only one record

For every output



D_1 D_2

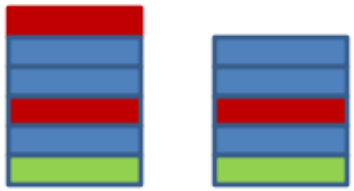
O

Simulate the presence or absence of a single record

Why All Pairs of Databases?

For every pair of neighboring databases that differ in only one record

For every output



D_1 D_2

O

Guarantee holds no matter what the other records are.

Why All Outputs?

For every pair of neighboring databases that differ in only one record

D_1 D_2

For every output

O

Should not be able to distinguish whether input was D_1 or D_2 no matter what the output

Privacy Parameter ε

For every pair of neighboring databases that differ in only one record

For every output

D_1 D_2

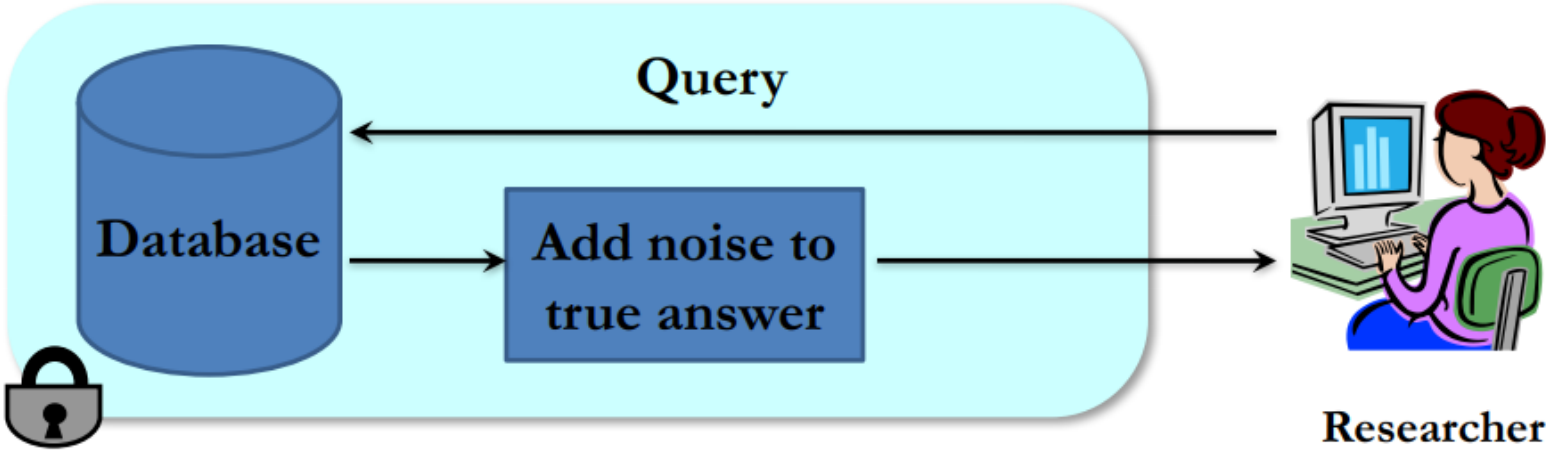
O

$$\Pr[A(D_1) = O] \leq e^{\varepsilon} \Pr[A(D_2) = O]$$

Controls the degree to which D_1 and D_2 can be distinguished.
Smaller ε gives more privacy (and worse utility)

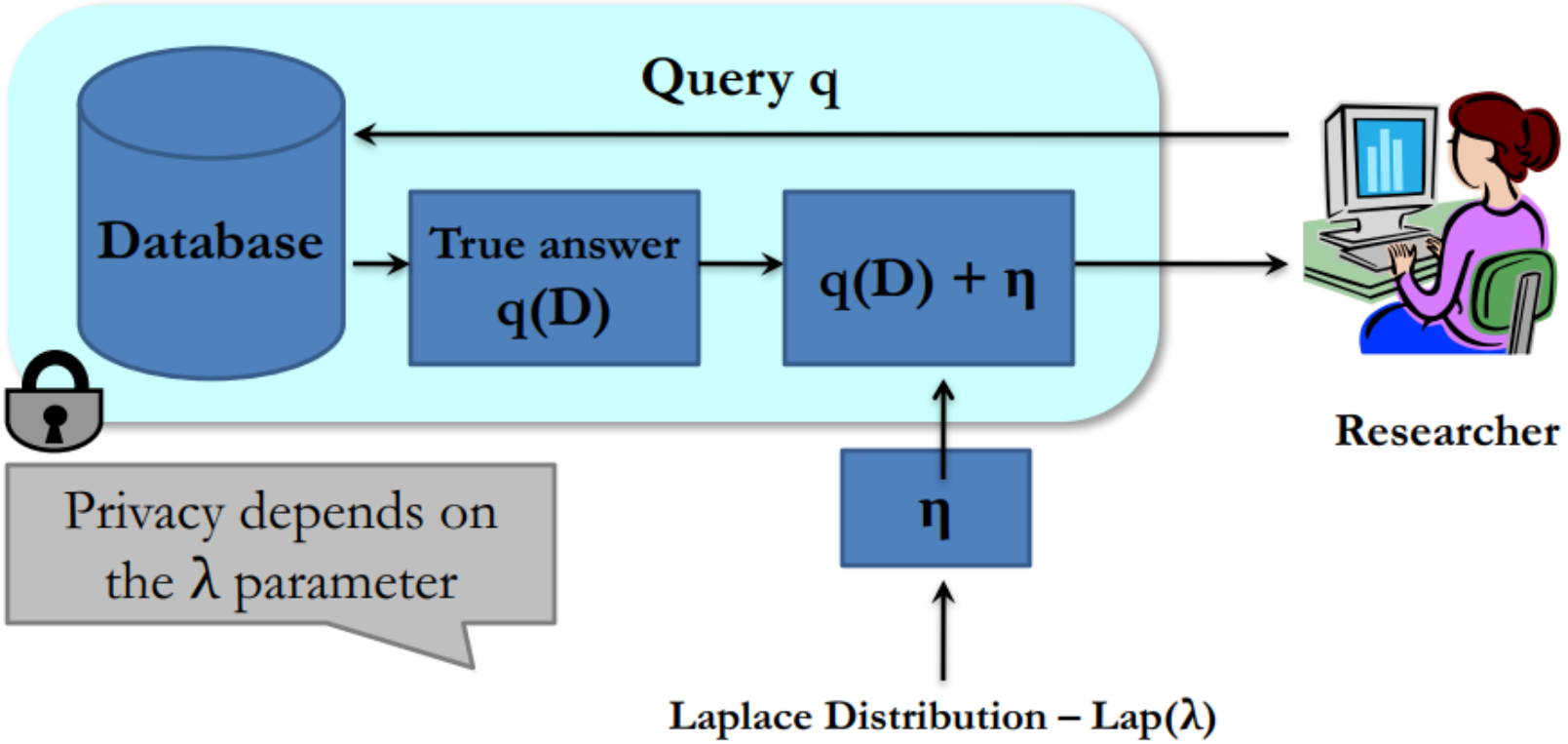
Basic Algorithm

Output Randomization



- Add noise to answers such that:
  - Each answer does not leak too much information about the database.
  - Noisy answers are close to the original answers.

Laplace Mechanism



Note that λ^2 is the variance, with mean = 0

A random variable has a $\mathrm{Laplace}(\mu, b)$ distribution if its probability density function is

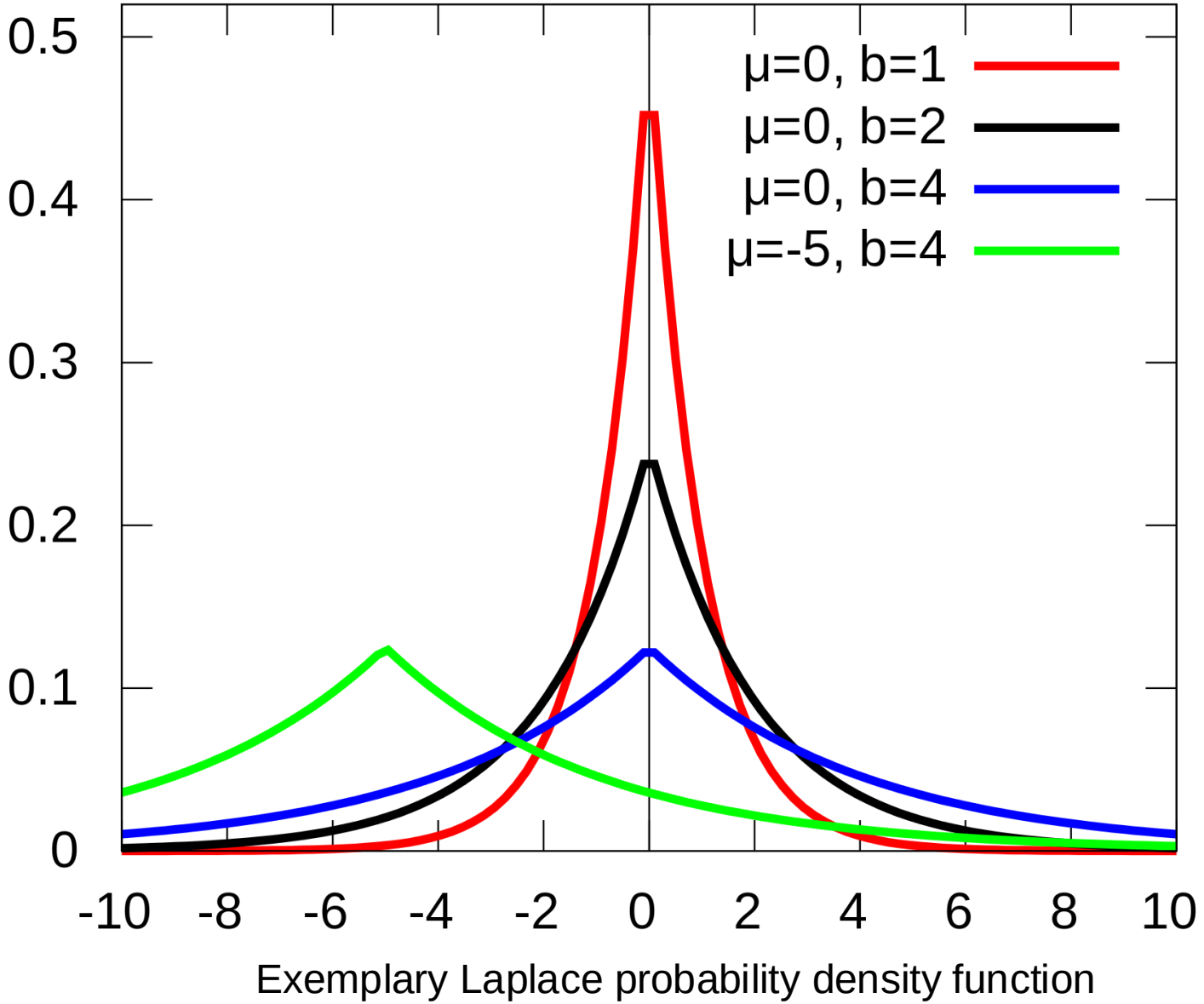
$$f(x \mid \mu, b) = \frac{1}{2b} \exp\left(-\frac{|x-\mu|}{b}\right)$$

$$= \frac{1}{2b} \begin{cases} \exp\left(-\frac{\mu - x}{b}\right) & \text{if } x < \mu \\ \exp\left(-\frac{x-\mu}{b}\right) & \text{if } x \geq \mu \end{cases}$$

Variance = 2 b^2 = λ^2, with mean μ = 0.

Here, μ is a location parameter and $b > 0$, which is sometimes referred to as the diversity, is a scale parameter. If $\mu = 0$ and $b = 1$, the positive half-line is exactly an exponential distribution scaled by 1/2.

https://en.wikipedia.org/wiki/Laplace_distribution



Exemplary Laplace probability density function

https://en.wikipedia.org/wiki/Laplace_distribution

How much noise for privacy?

[Dwork et al., TCC 2006]

Sensitivity: Consider a query q: $I \rightarrow$ R. S(q) is the smallest number s.t. for any neighboring tables D, D',

$$| \, q(D) - q(D') \, | \leq S(q)$$

i.e., Maximum change in the output by adding/deleting any one of the records in the database.

Thm: If **sensitivity** of the query is **S**, then the following guarantees **ε**-differential privacy.

$$\lambda = S/\varepsilon$$

Sensitivity example : COUNT query

- Number of people having disease
- Sensitivity=1

Disease (Y/N)
Y
Y
N
Y
N
N

Composition Theorems

Why composition?

- Reasoning about privacy of a complex algorithm is hard.

- Helps software design
  - If building blocks are proven to be private, it would be easy to reason about privacy of a complex algorithm built entirely using these building blocks.

A Bound on the Number of Queries

- In order to ensure utility, a statistical database must leak some information about each individual
- We can only hope to bound the amount of disclosure

- Hence, there is a limit on number of queries that can be answered

Sequential Composition

- If $M_1, M_2, ..., M_k$ are algorithms that access a private database D such that each M_i satisfies ε_i -differential privacy,

  then running all k algorithms sequentially satisfies ε-differential privacy with $\varepsilon=\varepsilon_1+...+\varepsilon_k$

Parallel Composition

- If $M_1, M_2, ..., M_k$ are algorithms that access disjoint databases $D_1, D_2, \ldots, D_k$ such that each M_i satisfies ε_i -differential privacy,

  then running all k algorithms in "parallel" satisfies ε-differential privacy with $\varepsilon = \max\{\varepsilon_1,\ldots,\varepsilon_k\}$

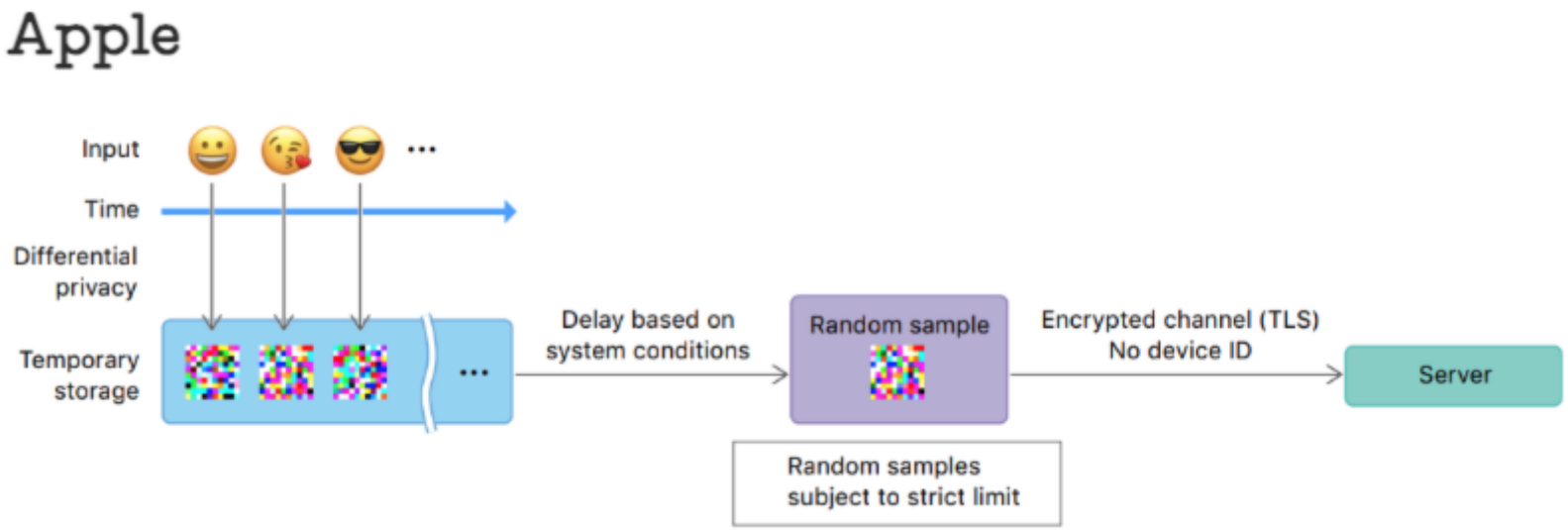
Postprocessing

- If M_1 is an ε-differentially private algorithm that accesses a private database D,

  then outputting $M_2(M_1(D))$ also satisfies ε-differential privacy.

A List of Real-world Uses of Differential Privacy - Apple

- Apple uses local DP to collect some data from end-user devices running iOS or macOS.



A List of Real-world Uses of Differential Privacy - Facebook

- The Full URLs Data Set provides data on user interactions with web pages shared on Facebook.

Facebook Privacy-Protected Full URLs Data Set

Solomon Messing, Christina DeGregorio, Bennett Hillenbrand, Gary King, Saurav Mahanti, Zagreb Mukerjee, Chaya Nayak, Nathaniel Persily, Bogdan State, Arjun Wilkins

13 February 2020

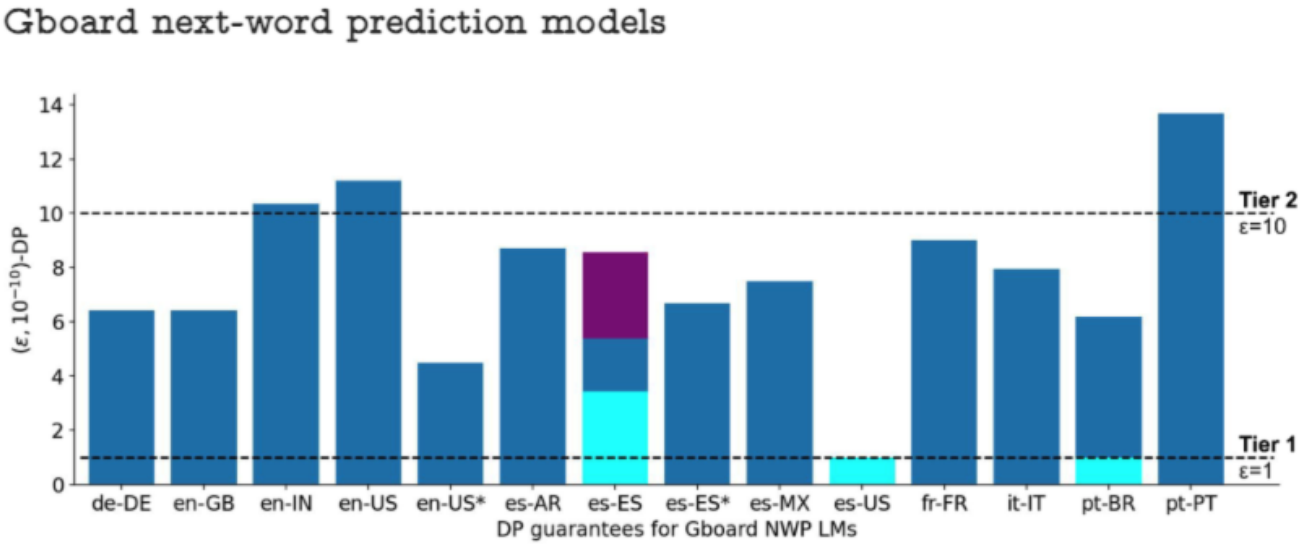
This dataset is the result of a collaboration between Facebook and Social Science One. It describes the dataset's scope, structure, fields, and privacy-preserving characteristics. This is the second of two planned steps in the release of this "Full URLs dataset", which we described at socialscience.one/blog/update-social-science-one. (We may also have other releases in the future.)

Citation

Messing, Solomon; DeGregorio, Christina; Hillenbrand, Bennett; King, Gary; Mahanti, Saurav; Mukerjee, Zagreb; Nayak, Chaya; Persily, Nate; State, Bogdan; Wilkins, Arjun, 2020, "Facebook Privacy-Protected Full URLs Data Set", https://doi.org/10.7910/DVN/TDOAPG, Harvard Dataverse, V2

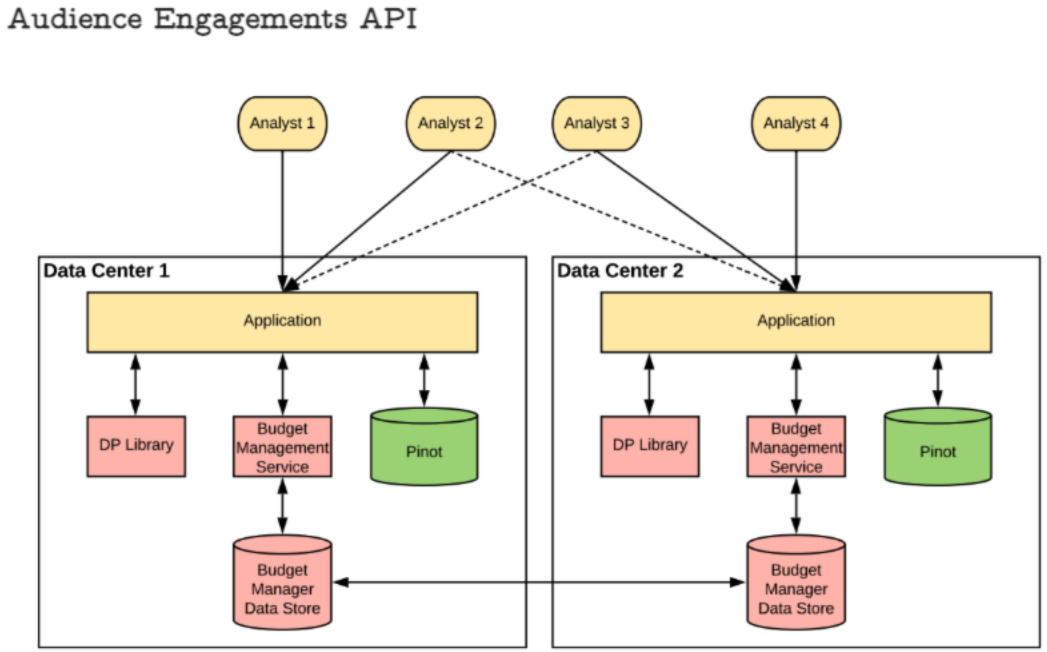
A List of Real-world Uses of Differential Privacy - Google

- Google uses federated learning along with DP to build next-word prediction models for Gboard, a virtual keyboard application for Android.



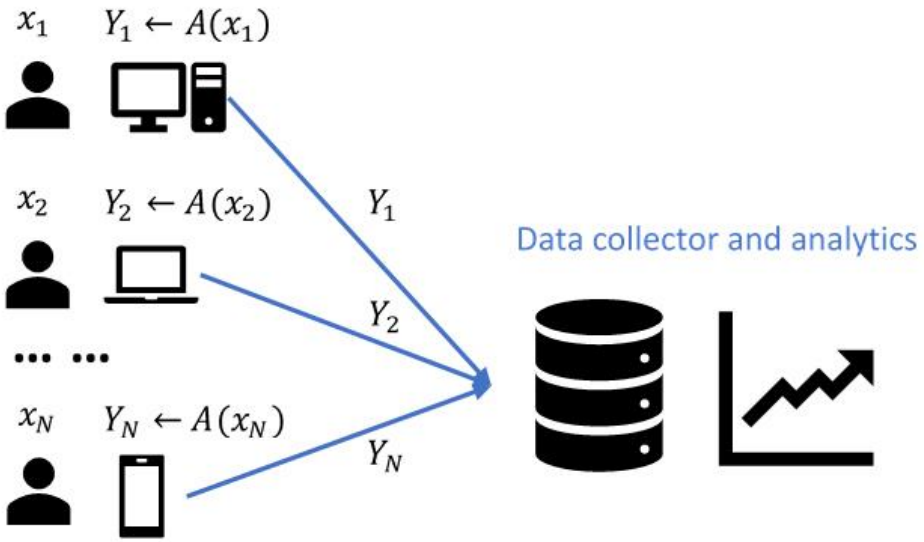
A List of Real-world Uses of Differential Privacy - LinkedIn

- The Audience Engagements API allows marketers to get information about LinkedIn users engaging with their content.



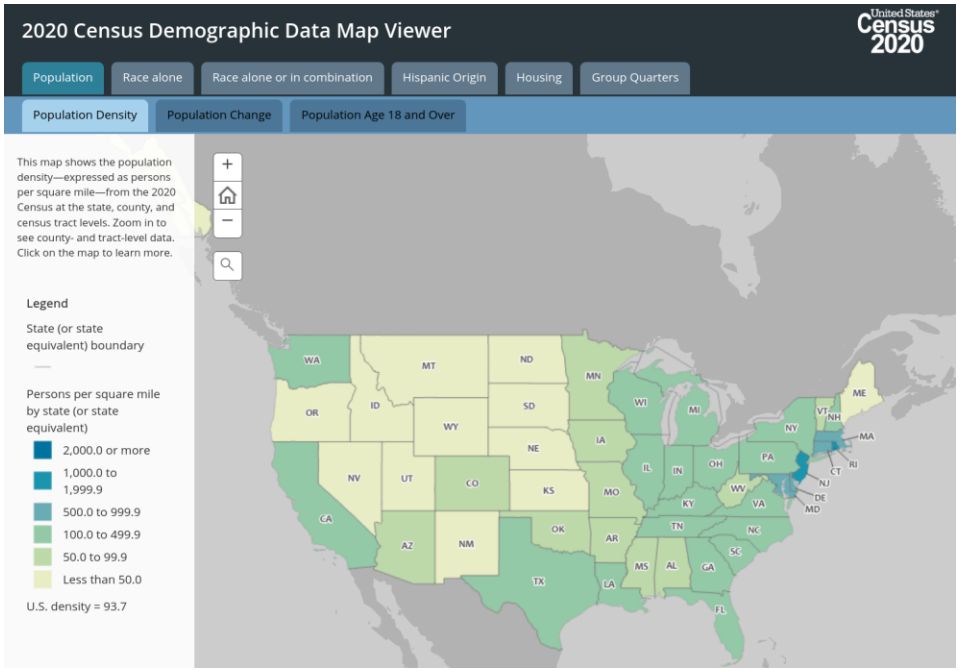
A List of Real-world Uses of Differential Privacy - Microsoft

- Microsoft collects telemetry data in Windows. The process used to get information about how much time users spend using particular apps uses local DP



A List of Real-world Uses of Differential Privacy - United States Census Bureau

- The U.S. Census Bureau published demonstration tables for their County Business Patterns data product via personalized DP tools.



Summary

- Differential privacy ensures that an attacker can't infer the presence or absence of a single record in the input based on any output

- Basic algorithm with random perturbation
  - Laplacian mechanism

- Composition rules help build complex algorithms using building blocks

Main Takeaways

- Accumulating failures: anonymization techniques.
- Differential privacy:
  - Not an algorithm; A standard providing a rigorous framework for developing
  - privacy technologies with provable quantifiable guarantees.
  - Rich theoretical work, now transitioning to practice.
    - First real-world applications and use, including by US Census, Google, Apple.
  - Very strong protection for cases where data flows across trust boundaries.
  - Legal landscape needs to be taken into account; DP to be combined (wisely!) with other technical and policy tools.

Learning More about Differential Privacy

- [Nissim et al, 2017] Differential Privacy: A Primer for a Non-technical Audience, Privacy Tools project.
- [Dwork 2011] A Firm Foundation for Private Data Analysis, CACM January 2011.
- [Heffetz & Ligett, 2014] Privacy and Data-Based Research, Journal of Economic Perspectives.
- [Dwork & Roth, 2014] The Algorithmic Foundations of Differential Privacy, Now publishers.
- [Vadhan'16] The complexity of differential privacy, Privacy Tools Project.
- Online course material, lectures and tutorials.

Projects, Software Tools [Partial List]

- [Microsoft Research] PINQ
- [UT Austin] Airavat: Security & Privacy for MapReduce
- [UC Berkeley] GUPT
- [CMU-Cornell-PennState] Integrating Statistical and Computational Approaches to Privacy
- [US Census] OnTheMap
- [Google] Rappor
- [UCSD] Integrating Data for Analysis, Anonymization, and Sharing (iDash)
- [Upenn] Putting Differential Privacy to Work
- [Stanford-Berkeley-Microsoft] Towards Practicing Privacy
- [Duke-NISS] Triangle Census Research Network
- [Harvard] Privacy Tools
- [Harvard, Georgetown, Buffalo] Computing Over Distributed Sensitive Data
- [Georgetown, Harvard, BU] Formal Privacy Models and Title 13

Privacy at Scale: Challenge and Sollution of Local Differential Privacy

Slides adapted from Tutorial on Local Differential Privacy at KDD'18

Aims

- To introduce the privacy model of local differential privacy (LDP)
  - Provide technical understanding, scaling of basic LDP algorithms
  - Show how these have been used in practice
- To connect with the privacy research literature
  - Give a deep understanding of the frequency estimation problem
  - Describe new work on LDP data mining and data modeling
- To suggest directions for future research
  - Identify topics that have just recently been considered
  - Suggest open problems and grand challenges for the area

Outline

- Introduction and Preliminaries
- State of the Art Deployments
- Theoretical Foundations
- Open Problems

Data Release Horror Stories

NYC Taxi & Limousine Commission

Jessica Alba (actor)

Strava's fitness tracker heat map reveals the location of military bases

Geolocation isn't a new problem for the military

By Andrew Liptak | @AndrewLiptak | Jan 28, 2018, 3:51pm EST

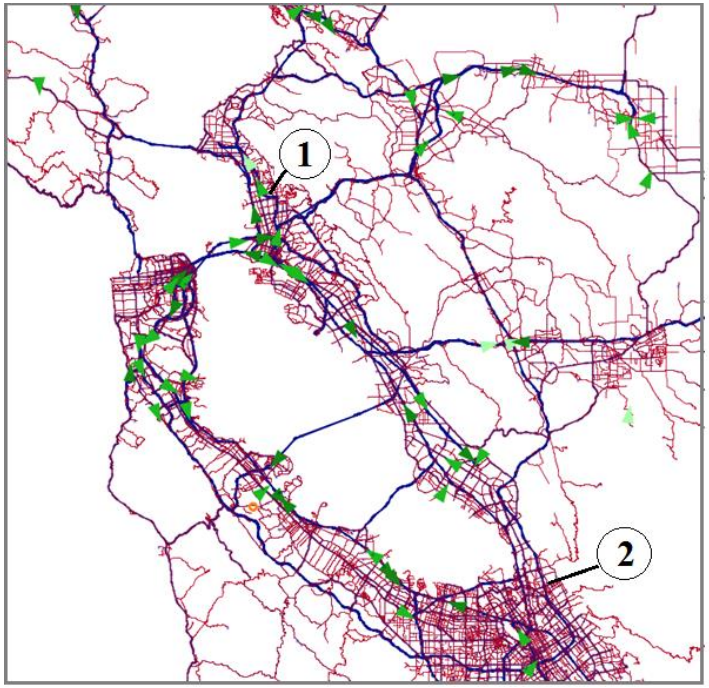
AOL

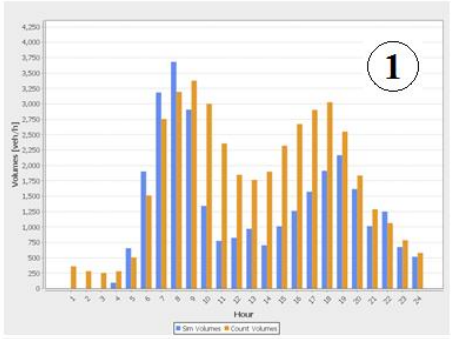
NETFLIX

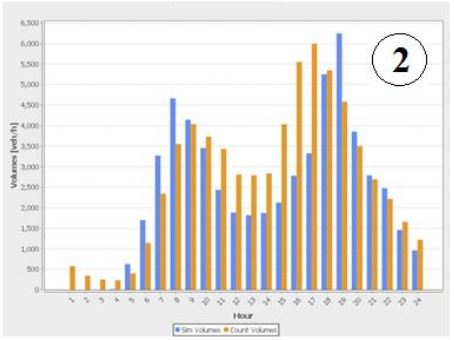
Commonwealth of Massachusetts
Group Insurance Commission
Your Benefits Connection

We need to solve this data release problem ...

But Why Release Data?







- Urban mobility modeling at scale
  - Smart traffic design is critical, congestion is getting worse

Big Data is Expected to Solve Big Problems

- The big data concept has taken off
  - Many organizations have adopted it
  - Entered the public vocabulary

- But this data is mostly about individuals
  - Individuals want privacy for their data

- How can researchers work on sensitive data?
  - The easy answer: anonymize it and share
  - The problem: we don't know how to do this

Record Level Anonymization: A Brief History

- Pre-1997: hash identifiers
- 1997: linking attack [S07]
  - Multiple QIs ≈ identifier
- 1998: k-anonymity [SS98]
  - Alphabet soup followed
- 2006+: syntactic attacks
- 2006: differential privacy [DM+06]
  - Semantic notion of privacy
- 2009: deFinetti's thm [K09]
  - Machine learning attack

Differential Privacy

A randomized algorithm K satisfies ε-differential privacy if:

Given two data sets that differ by one individual, D and D', any property S, and $\varepsilon > 0$:

$$\Pr[K(D) \in S] \leq e^{\varepsilon} \Pr[K(D') \in S]$$

- Can achieve ε-DP for counts by adding a random noise value

- Uncertainty due to noise → plausible deniability
  - Hides whether someone is present in the data

Achieving ε-DP in the Centralized Model

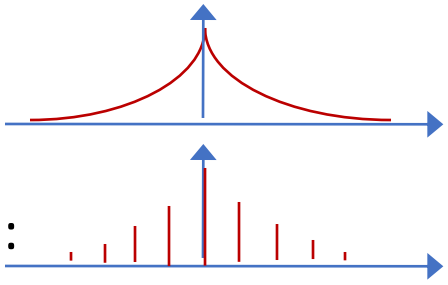
(Global) Sensitivity of publishing:

$s = \max_{D,D'} |F(D) - F(D')|$, D, D' differ by 1 individual

E.g., count individuals satisfying property P: s = 1

For every value that is output:

- Add Laplacian noise Lap(ε/s):
- Or Geometric noise for discrete case:



Simple rules for composition of differentially private outputs:

Given output O_1 that is ε_1-DP and O_2 that is ε_2-DP

- (Sequential composition) If inputs overlap, result is $(\varepsilon_1 + \varepsilon_2)$-DP
- (Parallel composition) If inputs disjoint, result is $\max(\varepsilon_1, \varepsilon_2)$-DP

Trying to Reduce Trust

- Most work on differential privacy assumes a trusted party
  - Data aggregator (e.g., organizations) that sees the true, raw data
  - Can compute exact query answers, then perturb for privacy

- A reasonable question: can we reduce the amount of trust?
  - Can we remove the trusted party from the equation?
  - Users produce locally private output, aggregate to answer queries

- One approach is to use SMC or homomorphic encryption
  - Merge encrypted data, and add noise for privacy inside encryption
  - Complex to get right, and very high computational overhead

Local Differential Privacy

- What about having each user run a DP algorithm on their data?
  - Then combine all the results to get a final answer
- On first glance, this idea seems crazy
  - Each user adds noise to mask their own input
  - So surely the noise will always overwhelm the signal?
- But … noise can cancel out or be subtracted out
  - We end up with the true answer, plus noise which can be smaller
  - However, noise is still larger than in the centralized case

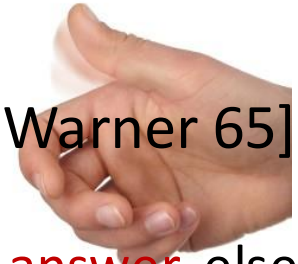
Local Differential Privacy

- We can achieve LDP, and obtain reasonable accuracy (for large N)
  - The error typically scales with $\sqrt{N}$
- Generic approach: apply centralized DP algorithm to local data
  - But error might still be quite large
  - Unclear how to merge private outputs (e.g. private clustering)
- So we seek to design new LDP algorithms
  - Maximize the accuracy of the results
  - Minimize the costs to the users (space, time, communication)
  - Ensure that there is an accurate algorithm for aggregation

Privacy with a coin toss: Randomized Response

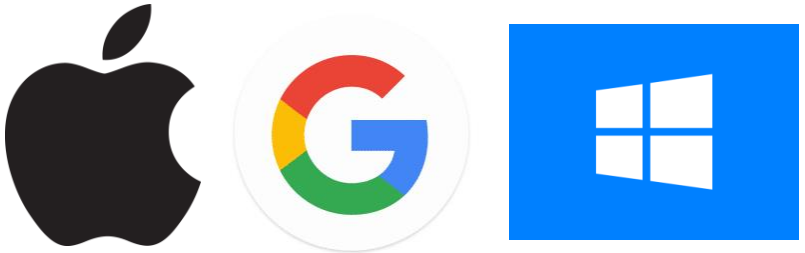
- Each user has a single bit of private information
  - Encoding e.g. political/sexual/religious preference, illness, etc.

- Randomize Response (RR): toss an unbiased coin [Warner 65]
  - If Heads (probability $p = ½$), report the true answer
  - Else, toss unbiased coin again: if Heads, report true answer, else lie



- Collect responses from N users, subtract noise
  - E.g., for average, portion, etc., the error in the estimate is proportional to $1/\sqrt{N}$
  - Simple LDP algorithm with parameter $\varepsilon=\ln((¾)/(¼)) = \ln(3)$
  - Generalization: allow biased coins ($p \neq ½$)

Privacy in practice

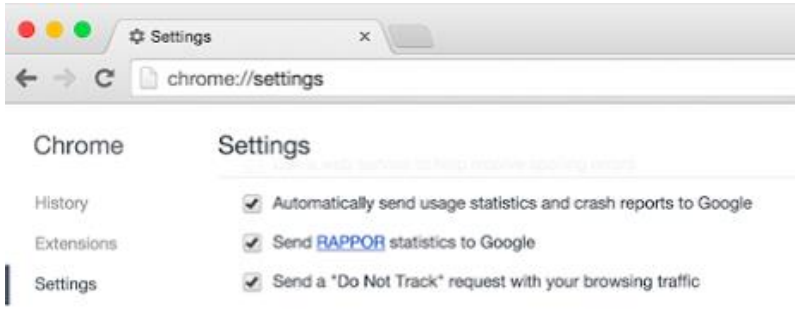


- Differential privacy based on coin tossing is widely deployed!
  - In Google Chrome browser, to collect browsing statistics
  - In Apple iOS and MacOS, to collect typing statistics
  - In Microsoft Windows to collect telemetry data over time
  - From Snap to perform modeling of user preference
  - This yields deployments of over 100 million users each

- All deployments are based on RR, but extend it substantially
  - To handle the large space of possible values a user might have

- Local Differential Privacy is state of the art in 2018
  - Randomized response invented in 1965: five decades ago!

Google's RAPPOR

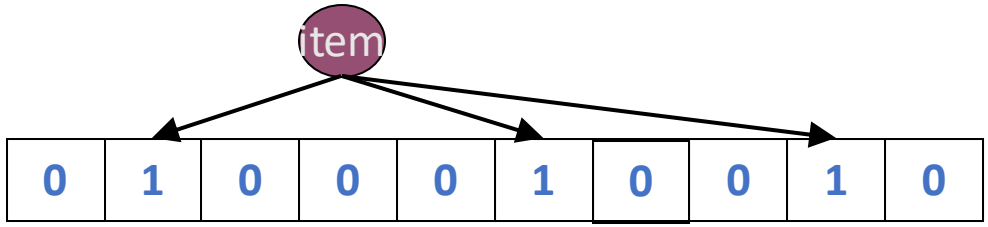


- Each user has one value out of a very large set of possibilities
  - E.g. their favourite URL, www.nytimes.com

- Attempt 1: run RR for all values: google.com? Bing.com? …
  - User has sparse binary vector with one 1: run RR on every bit
  - Satisfies 2ε-LDP: change user's choice, 2 bits change: 1 → 0, 0 → 1
  - Slow: sends 1 bit for every possible index in the vector
  - Limited: can't handle new possibilities being added

- Try to do better by reducing domain size through hashing

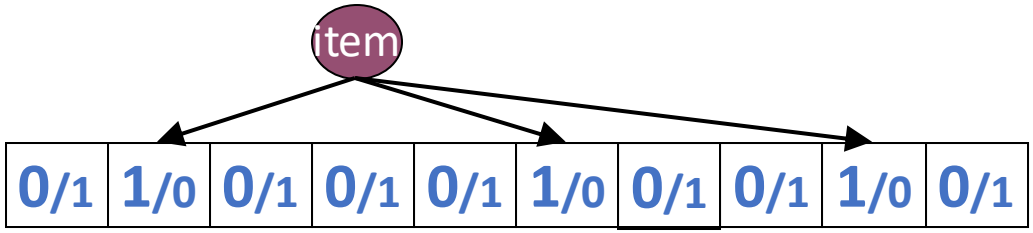
Bloom Filters



- Bloom filters [B70] compactly encode set membership
  - k hash functions map items to m-bit vector k times
  - Update: Set all k entries to **1** to indicate item is present
  - Query: Can lookup items, store set of size n in O(n) bits
  - Analysis: choose k and m to obtain small false positive probability

- Can be merged by OR-ing vectors (of same size)
  - Duplicate insertions do not change Bloom filters

Counting Bloom Filters & Randomized Response



- Idea: apply Randomized Response to the bits in a Bloom filter
  - Not too many bits in the filter compared to all possibilities

- Each user maps their input to at most k bits in the filter
  - Privacy guarantee with parameter $2k\varepsilon$

- Combine all user reports and observe how often each bit is set

Decoding Noisy Counting Bloom Filters

- We have a noisy Bloom filter, where each bit has a probability

- To estimate the frequency of a particular value:
  - Look up its bit locations in the Bloom filter
  - Compute the unbiased estimate of the probability each is 1
  - Take the minimum of these estimates as the frequency
  - More advanced decoding heuristics to decode all at once

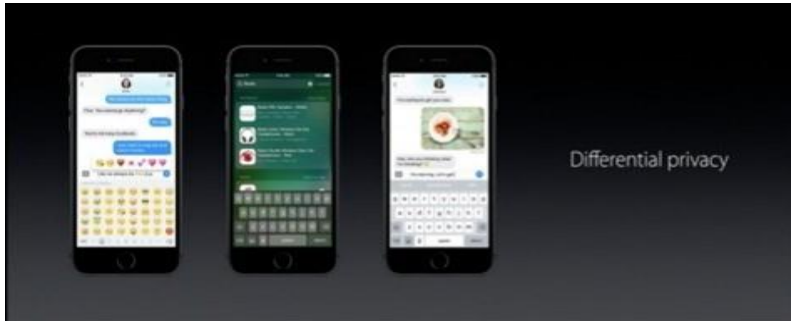
RAPPOR in practice

- The RAPPOR approach is implemented in the Chrome browser
  - Collects data from opt-in users, tens of millions per day
  - Open source implementation available

- Tracks settings in the browser, e.g. home page, search engine
  - Many users unexpectedly change home page → possible malware

- Typical configuration:
  - 128 bit Bloom filter, 2 hash functions, privacy parameter ~0.5
  - Needs about 10K reports to identify a value with confidence

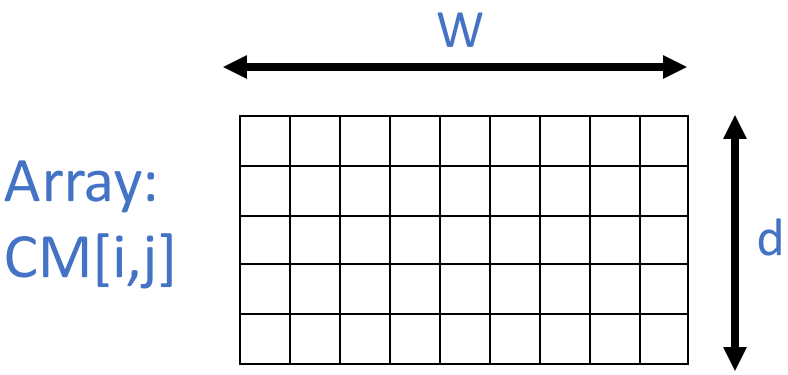
Apple: Sketches and Transforms



- Similar problem to RAPPOR: count frequencies of many items
  - For simplicity, assume that each user holds a single item
  - To reduce burden of collection, can size of summary be reduced?

- Instead of Bloom Filter, make use of sketches
  - Similar idea, but better suited to capturing frequencies

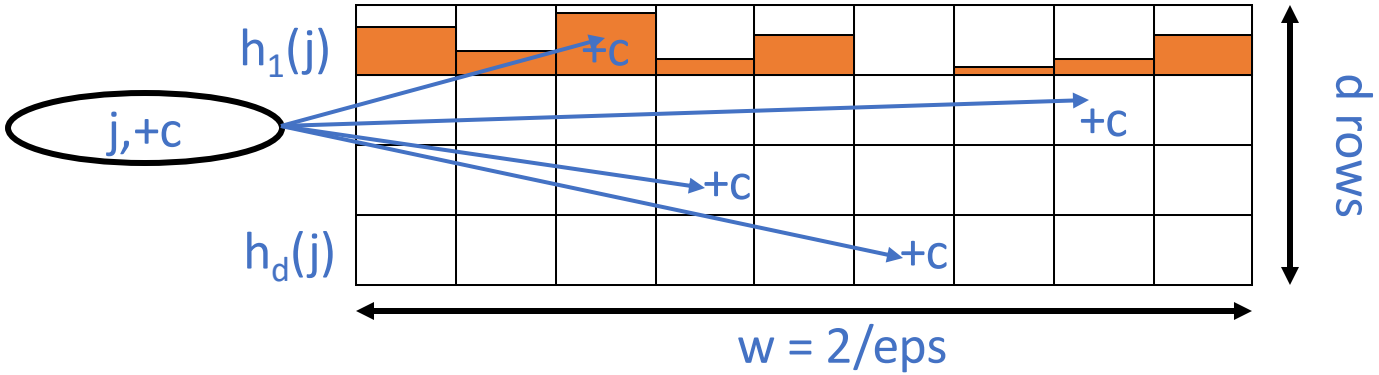
Count-Min Sketch [CM04]



- Count-Min sketch encodes item counts
  - Allows estimation of frequencies (e.g. for selectivity estimation)
  - Some similarities in appearance to Bloom filters

- Model input data as a sparse frequency vector x of dimension U
  - Create a small summary as an array of w × d in size
  - Use d hash function to map vector entries to [1..w]

Count-Min Sketch



- Update: each entry in vector x is mapped to one bucket per row
- Query: estimate x[j] by taking $\min_k CM[k,h_k(j)]$
  - Guarantees error less than $eps*\|x\|_1$ in size O(1/eps)
- Merge two sketches by entry-wise summation

Count-Min Sketch + Randomized Response

- Each user encodes their (unit) input with a Count-Min sketch
  - Then applies randomized response to each (binary) entry

- Aggregator adds sketches, unbiases entries, estimates using mean
  - More robust than taking min when there is random noise
  - Error is a random variable with variance proportional to $\|x\|_2^2/(wdn)$
  - I.e. (absolute) error decreases proportional to $1/\sqrt{n}$, $1/\sqrt{\text{sketch size}}$

- Bigger sketch → more accuracy
  - But we want smaller communication

Hadamard Transform

- The distribution of interest could be sparse and spiky
  - This distribution is preserved under CM sketch
  - But, if we don't report the whole sketch, we might lose information

$$\begin{bmatrix} H^* & H^* \\ H^* & -H^* \end{bmatrix} = \begin{bmatrix} -1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 1 & 1 & -1 \\ -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 \\ 1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 \end{bmatrix}.$$

- Idea: Transform sketched data to "spread out" the signal
  - Use Hadamard transform (a discrete Fourier transform)

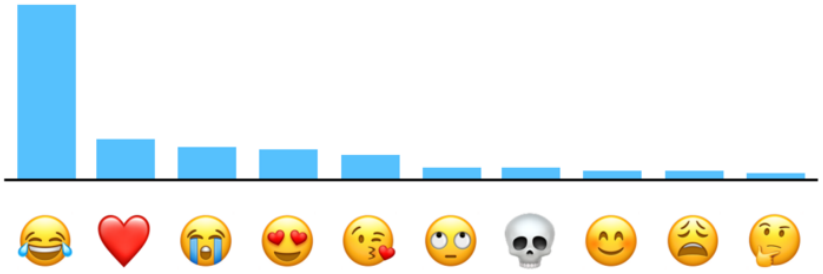
Low Communication with Hadamard Transform

$$\begin{bmatrix} H^* & H^* \\ H^* & -H^* \end{bmatrix} = \begin{bmatrix} -1 & 1 & 1 & 1 & -1 & 1 & 1 & 1 \\ 1 & -1 & 1 & 1 & 1 & -1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 & -1 & 1 \\ 1 & 1 & 1 & -1 & 1 & 1 & 1 & -1 \\ -1 & 1 & 1 & 1 & 1 & -1 & -1 & -1 \\ 1 & -1 & 1 & 1 & -1 & 1 & -1 & -1 \\ 1 & 1 & -1 & 1 & -1 & -1 & 1 & -1 \\ 1 & 1 & 1 & -1 & -1 & -1 & -1 & 1 \end{bmatrix}.$$

- Now the user just samples one entry in the transformed sketch
  - No danger of missing the important information – it's everywhere!
  - User only has to send one bit of information → low communication

- Aggregator can invert the transform to get the sketch back
  - Variance is essentially unchanged from previous case

Apple's Differential Privacy in Practice



The Count Mean Sketch technique allows Apple to determine the most popular emoji to help design better ways to find and use our favorite emoji. The top emoji for US English speakers contained some surprising favorites.

- Apple uses their system to collect data from iOS and OS X users
  - Popular emojis: (heart) (laugh) (smile) (crying) (sadface)
  - "New" words: bruh, hun, bae, tryna, despacito, mayweather
  - Which websites to mute, which to autoplay audio on!

- Deployment settings: w=1000, d=1000, ε=2-8 (some criticism)

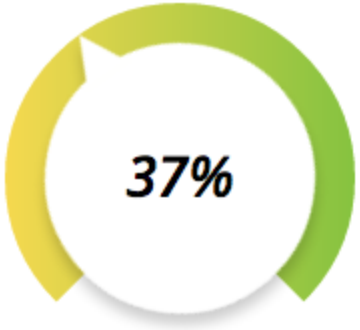
Microsoft telemetry data collection

- Microsoft want to collect data on app usage
  - How much time was spent on a particular app today?
  - Allows finding patterns over time

- Makes use of multiple subroutines:
  - 1BitMean to collect numeric data
  - dBitFlip to collect (sparse) histogram data
  - Memoization and output perturbation to allow repeated probing

- Has been implemented in Windows since 2017

1BitMean



- Problem 1: Want to estimate the mean f value of all users
  - Each user has a value f between 0 and 1

- Simple approach: build a histogram
  - How to decide bucket boundaries? Introduces rounding error

- 1bitMean: modify randomized response
  - User outputs 1 with probability linear in f: $(1 + f\cdot(e^{\varepsilon} - 1))/(e^{\varepsilon} + 1)$
  - f=0 gives $Pr[1] = 1/(e^{\varepsilon} + 1)$, while f=1 gives $Pr[1]= e^{\varepsilon}/(e^{\varepsilon} + 1)$
  - Taking the average of (unbiased version of) reports gives mean
  - Error can be bounded proportional to $1/\varepsilon\sqrt{n}$

dBitFlip

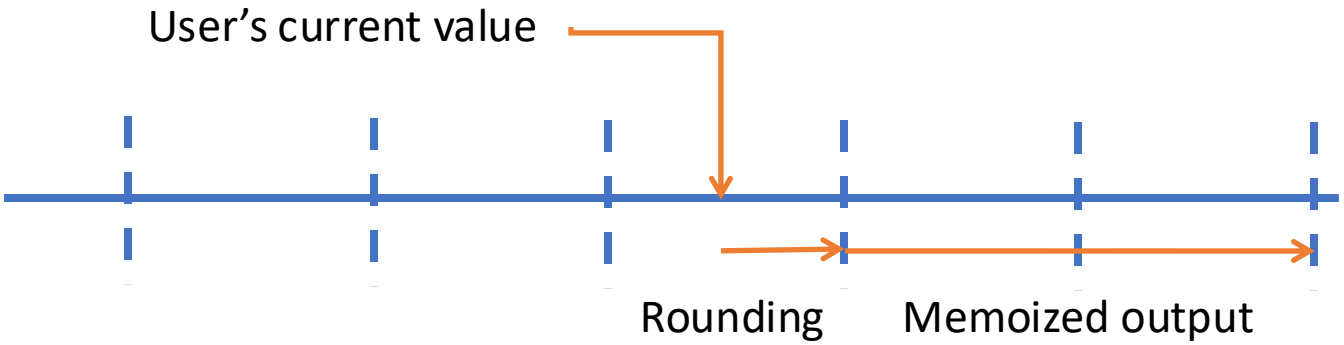
- Problem 2: build a histogram over data items from a population
  - Each user has a single item value out of k possibilities
  - Essentially the same problem that Google and Apple tackle

- Simple approach: apply RR to every bucket, costly when k is large

- dBitFlip: apply Randomized Response to a sample of buckets
  - User samples d buckets without replacement, applies RR to each
  - Aggregator sums and unbiases the noisy reports
  - Error proportional to √(k/d): trades off error for speed
  - Many alternative solutions for this “frequency oracle” problem

Memoization

- Collecting data continuously erodes the privacy guarantee
  - E.g. if we know a user's value didn't change, we can estimate it



- Memoization: a heuristic to reduce privacy loss
  - Each user rounds their current value to a discrete range
  - Each user precomputes their output for each range
  - Choosing a fixed random shift for each user reduces error

Output Perturbation

- Note however "memoization violates differential privacy"!
  - Can tell the difference between change and no change
  - But at least this process hides what the value was

- Output perturbation aims to address this problem
  - For each user output bit, flip it with small probability γ, e.g. 0.2
  - Increases error by a factor of $1/(1 - 2\gamma)$ (so 5/3 for $\gamma=0.2$)

- Increases uncertainty about exactly when a change happened
  - After long enough, can be certain that some change happened

MS Telemetry Collection in Practice

- Deployed in Windows 10 Fall Creators Update (October 2017)
  - Collects number of seconds users spend in different apps
  - Parameters: $\varepsilon = 1$ and $\gamma = 0.2$
  - Collection period: every 6 hours

- Collects data on all app usage, not just one at a time
  - Can analyze based on the fact that total time spent is limited
  - Gives overall guarantee of $\varepsilon = 1.672$ for a round of collection

Theoretical Foundations

From DP to LDP: Formal Definition

Idea of DP: Any output should be about as likely regardless of whether or not I am in the dataset

A randomized algorithm $\boldsymbol{A}$ satisfies $\boldsymbol{\varepsilon}$-differential privacy, iff for any two neighboring datasets $\boldsymbol{D}$ and $\boldsymbol{D'}$ and for any output $\boldsymbol{O}$ of $\boldsymbol{A}$,

$$\Pr[\boldsymbol{A}(\boldsymbol{D}) = \boldsymbol{O}] \leq \exp(\boldsymbol{\varepsilon}) \cdot \Pr[\boldsymbol{A}(\boldsymbol{D'}) = \boldsymbol{O}]$$

A randomized algorithm $\boldsymbol{A}$ satisfies $\boldsymbol{\varepsilon}$-local differential privacy, iff for any two inputs $\boldsymbol{x}$ and $\boldsymbol{x'}$ and for any output $\boldsymbol{y}$ of $\boldsymbol{A}$,

$$\Pr[\boldsymbol{A}(\boldsymbol{x}) = \boldsymbol{y}] \leq \exp(\boldsymbol{\varepsilon}) \cdot \Pr[\boldsymbol{A}(\boldsymbol{x'}) = \boldsymbol{y}]$$

Run by

ε is also called privacy budget
Smaller ε ➔ stronger privacy

person

Idea of LDP: Any output should be about as likely regardless of my secret

Properties of (Centralized) DP

A randomized algorithm $\boldsymbol{A}$ satisfies $\boldsymbol{\varepsilon}$-differential privacy, iff for any two neighboring datasets $\boldsymbol{D}$ and $\boldsymbol{D'}$ and for any output $\boldsymbol{O}$ of $\boldsymbol{A}$,

$$\Pr[\boldsymbol{A}(\boldsymbol{D}) = \boldsymbol{O}] \leq \exp(\boldsymbol{\varepsilon}) \cdot \Pr[\boldsymbol{A}(\boldsymbol{D'}) = \boldsymbol{O}]$$

- Post-processing (of the output) is free
  - does not c
- Parallel composition
  - partition the dataset into subsets, each applying an ε_i-DP algorithm, the overall result satisfies $\max(\varepsilon_i)$-DP
- Sequential composition
  - apply k DP algorithms, each using ε_i, result satisfies $\sum \varepsilon_i$-DP

What about LDP?

Properties of LDP

A randomized algorithm $\boldsymbol{A}$ satisfies $\boldsymbol{\varepsilon}$-local differential privacy, iff for any two inputs $\boldsymbol{x}$ and $\boldsymbol{x'}$ and for any output $\boldsymbol{y}$ of $\boldsymbol{A}$,

$$\Pr[\boldsymbol{A}(\boldsymbol{x}) = \boldsymbol{y}] \leq \exp(\boldsymbol{\varepsilon}) \cdot \Pr[\boldsymbol{A}(\boldsymbol{x'}) = \boldsymbol{y}]$$

- Post-processing is also free
  - does not consume privacy budget
- No direct parallel composition
  - because each user only has one record, which cannot be partitioned
  - but one can apply different questions to different subsets of users
- Sequential composition
  - apply k LDP algorithms, each using ε_i, result satisfies $\sum \varepsilon_i$-LDP

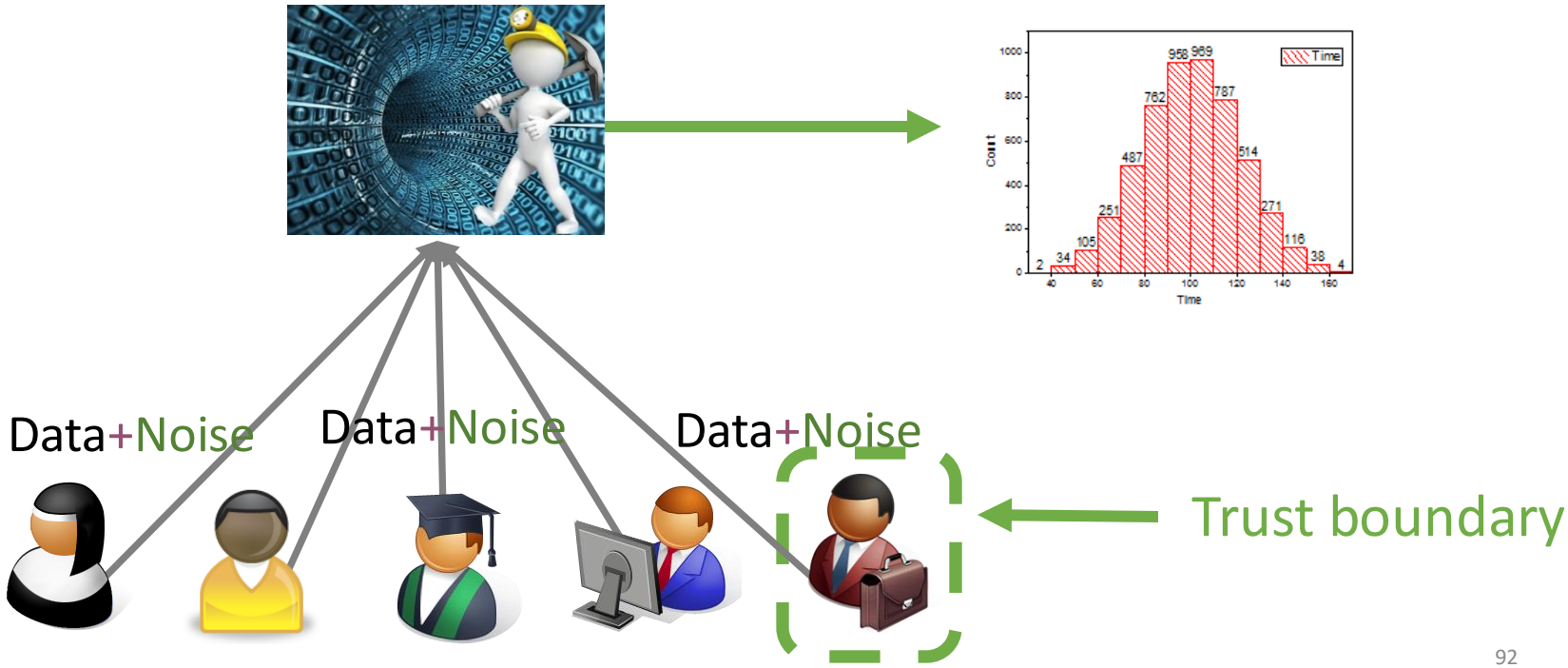
Key difference between DP and LDP

- DP concerns two neighboring datasets
- LDP concerns any two values
- As a result, the amount of noise is different: In aggregated result for counting queries
  - Noise in DP is $\Omega(1)$ (sensitivity is constant)
  - But in LDP, even noise for each user is constant, the aggregated result is $\Omega(\sqrt{n})$ [1]
  - If the result is normalized (divide the result with n), noise is $\Omega\left(\frac{1}{n}\right)$ versus $\Omega\left(\frac{1}{\sqrt{n}}\right)$

[1] Optimal lower bound for differentially private multi-party aggregation by T.-H. H. Chan, E. Shi, and D. Song

Frequency Estimation

- Assumption: each user has a single value x from a categorical domain D
- Goal: Estimate the frequency of any value in D



Frequency Oracle Framework

- $x := E(v)$

takes input value v from domain D and outputs an encoded value x

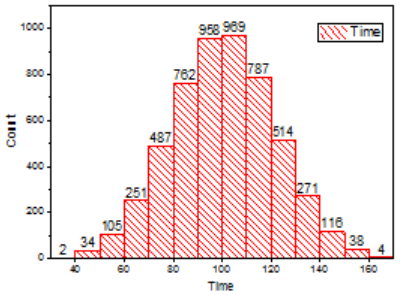
- $y := P(x)$

takes an encoded value x and outputs y.

y

- $c := Est(\{y\})$

takes reports $\{y\}$ from all users and outputs estimations $c(v)$ for any value v in domain D



P is ε -LDP iff for any v and v' from D, and any valid output y,

$$\frac{\Pr[P(E(v))=y]}{\Pr[P(E(v'))=y]} \leq e^{\varepsilon}$$

Random Response (Warner'65)

- Survey technique for private questions
- Survey people:
  - "Do you a
- Each person
  - Flip a secre
  - Answer truth if head (w/p 0.5)
  - Answer
  - E.g., a pa w/p 25%
- To get unbiased estimation of the distribution:
  - If n_v out of n people have the disease, we expect to see

$$E[\,I_v] = 0.75n_v + 0.25(n - n_v)$$ "yes" answers

  - $c(n_v) = \frac{I_v - 0.25n}{0.5}$ is the unbiased estimation of number of patients

For any $\boldsymbol{v}$ and $\boldsymbol{v}'$ from "yes" and "no",

$$\frac{\Pr[P(\boldsymbol{v}) = \boldsymbol{v}]}{\Pr[P(\boldsymbol{v}') = \boldsymbol{v}]} \leq 3 = e^{\varepsilon}$$

seeing answer, not certain about the secret.

This only handles binary attribute.
We will handle the more general setting.

Concrete Example (Let's do math)

A patient will answer "yes" w/p 75%, and "no" w/p 25%

	truth	->yes	->no
yes	80	40+20	0+20
no	20	0+5	10+5

$$c(n_v) = \frac{I_v - 0.25n}{0.5}$$

observed	65	35
estimate	80	20

(Simple) Proofs

- $E[c(n_v)] = n_v$
- We have
  - $c(n_v) = \frac{I_v - 0.25n}{0.5}$
  - $E[\, I_v] = 0.75n_v + 0.25(n - n_v)$
- $E[c(n_v)] = \frac{E[I_v] - 0.25n}{0.5} = \frac{0.75n_v + 0.25(n - n_v) - 0.25n}{0.5} = n_v$
- Can be extended to other protocols
- Variance can be derived similarly

Probabilistic Analysis

Compare the result $c(v)$ with the ground truth n_v.

- $c(v)$ is a random variable
- Show that $c(v)$ is unbiased: $E[c(n_v)] = n_v$
- Compute the variance of $c(v)$: $Var[c(v)]$
- Use appropriate inequality to bound the error
  - Bernstein or Hoeffding inequalities
- Transform from variance to error bound
  - Since $c(v)$ is a binomial variable (sum of iid Bernoulli variables)

LDP Applications

Beyond heavy hitters and simple statistics, many LDP applications:

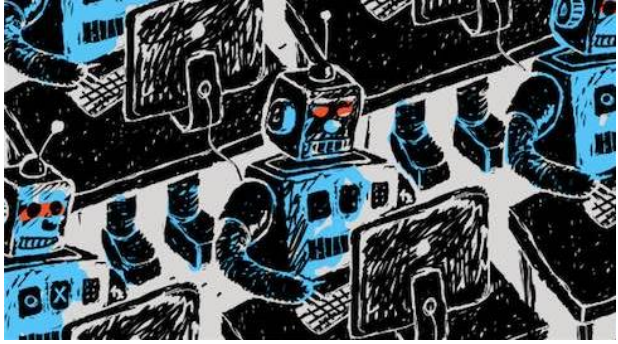
- Text and language modelling
- Marginal distributions and correlation
- Spatial data
- Graphs and social networks
- Machine learning

Across all these LDP applications, some common features emerge

Social Network Data

- Social graphs may have a lot of sensitive valuable information about a person's ties.
- As for central DP, there are alternative LDP definitions for graphs:
  - Node LDP: LDP definition applied to whole adjacency list of a **user**
  - Edge LDP: LDP restricted to adjacency lists that differ in one **edge**
- Goal: Release social network data as graph models
  - Hard enough in the central DP model, so only harder with LDP
  - Aim to release parameters of a statistical graph model
- Results so far only for the (more relaxed) edge LDP definition

Synthetic Social Graphs



- **Generating Synthetic Decentralized Social Graphs with Local Differential Privacy** [Qin et al. CCS 2017] considers two baselines:

1. Randomized neighbour list (RNL) on user's adjacency list
   - Meets LDP, but introduces a lot of fake edges
   - Need a model where parameters have more support
2. Degree-based graph generation (DGG)
   - Compute degree of each node (accurate under edge LDP)
   - Sample from a graph model that only needs degrees (eg BTER)

- Neither baseline is very satisfying
  - Goldilocks effect: RNL too detailed, DGG too coarse

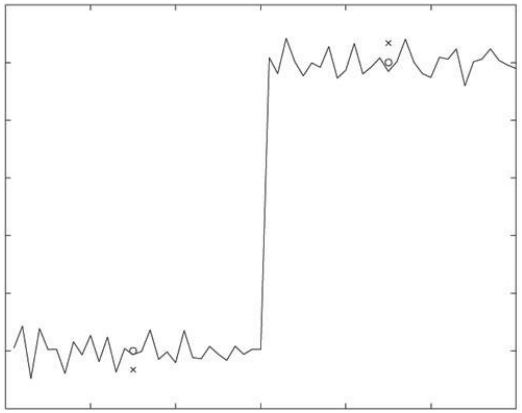
Open Problems and Closing Thoughts

Immediate Open Problems for LDP

- Take any data type or analysis goal and ask "Can we handle this under the LDP model?":
  - Sentiment analysis for (private) reviews: LDP-TripAdvisor?
  - Trajectory analysis of GPS movements: LDP-RunKeeper?
  - Social network data analysis: LDP-Facebook?
- More challenging to apply to richer, unstructured data
  - E.g. Free text in written medical notes
  - E.g. Multimedia – images and videos
  - The semantics of privacy are not yet well defined here!

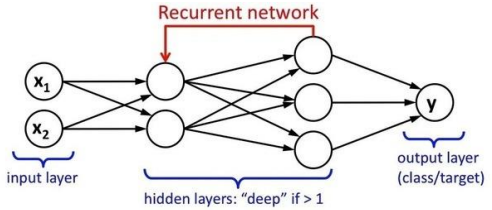
- Algorithm engineering: make LDP widely available
  - RAPPOR is open source, but there's no easy toolkit for LDP yet

Evolving Data



- Often want to collect data multiple times
  - Regular update of usage statistics e.g. daily
- Current deployments offer only weak longitudinal guarantees
  - Google & Microsoft: memoization and re-randomization heuristics
  - Apple: privacy budget ‘replenished’ every day [Tang et al ArXiv 17]
- LDP for Evolving Data [Joseph, Roth, Ullman, Waggoner 18] gives a privacy guarantee for repeated release of a statistic
  - Time divided into ‘epochs’ of multiple timesteps
  - In each epoch, subsets of users ‘vote’ to refresh the global statistic
  - Voting thresholds make cost proportional to number of changes
- More work needed to implement and make practical!

Deep Learning and Privacy



- Methods based on 'deep learning' (neural networks) are currently very popular for accurate machine learning
  - This is challenging for privacy: they can (over)fit the data
- 'Federated learning' promises some privacy gains
  - The model is learned with the help of distributed users
  - Each selected user updates the current model based on their data
  - Data never leaves the user – but gradients could still be revealing?
- Some examples in other DP models are promising, **e.g. Learning differentially private language models without losing accuracy** [McMahan et al. ICLR 18]
  - Have large number of users participate in local training.
  - Collect their updates and bound the sensitivity of total update, add proportional Gaussian noise and update the global model.

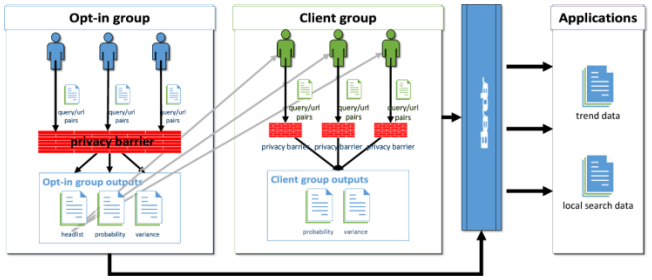
Theoretical Underpinnings

- LDP (re)emerged most recently from the theory literature
  - **What can we learn privately?** [Kasiviswanathan et al, FOCS 08] Showed the model is equivalent to Kearns' model of 'statistical queries' – up to polynomial blow up in population size n!
  - **Local Privacy and Statistical Minimax Rates** [Duchi et al, FOCS 13] Showed asymptotic optimality of basic LDP algorithms such as Laplace mechanism and Randomized Response
- Still many theoretical questions about LDP:
  - Can we show more lower bounds for the accuracy tradeoff?
  - What problems do not have efficient LDP protocols?
  - Can all LDP algorithms have each user output only 1 (noisy) bit? (Some good reasons to believe so)

Generalizing LDP

- Additive error probability
  - Centralized DP allows an additional parameter δ, which relaxes the requirement to $\Pr[X \in S|D] \leq e^{\varepsilon} \Pr[X \in S|D'] + \delta$
  - LDP can be relaxed similarly, but it makes no difference (in theory)
  - What are the practical consequences of this relaxation?
- Multiple rounds of data collection
  - Some LDP protocols collect data once, others interact with users
    - Queries in subsequent rounds depend on prior rounds
  - What is the tradeoff between rounds and accuracy/communication?

Hybrid Privacy Models



- We can imagine 'hybrid' models of privacy
  - Users run LDP with different (personalized) values of ε
  - Some users entrust their raw data to the aggregator
- Blender [Avent et al UsenixSecurity 17] fleshes out this model
  - Combines data obtained from the 'opt-in' group O with the 'regular' (LDP) clients C
  - E.g. use opt-in data to generate list of possible query answers
  - Use combined data from both groups to rank the list
- Several natural questions about theory and applications:
  - Quantify benefits of hybrid privacy as a function of |O| and |C|
  - What is the 'killer app' for this model of privacy?
  - Other hybrids: each user chooses what to reveal non-privately

MPC and DP

- The main weakness of LDP is the magnitude of the error added
  - Compared to central DP, where the noise is essentially a constant
- Could we obtain central DP error with a local guarantee?
- Change the game: use secure multiparty computation (MPC)
  - Each participant encrypts their data, which is aggregated
  - Each user also adds some carefully chosen noise
  - The noise combines to be DP-level noise: "computational DP"
  - We can finally decrypt the answer with central DP noise
  - **Private Nearest Neighbors Classification in Federated Databases (Schoppmann, Gascon and Balle 2018)**
- Several papers give protocols to achieve this
  BUT computational overheads are **very high**
  - Hence why Google, Apple, Microsoft went the LDP route?

Reflecting on LDP

- Local Differential Privacy is a big success for privacy research
  - Adopted by Google, Apple, Microsoft and more for deployment
  - Deployments affecting (hundreds of) millions of users
  - In contrast, centralized DP has smaller success (but, see US Census)
- However, there are reasons to pause and reflect:
  - LDP only works when you can rely on millions of active participants
  - Companies using LDP also gather vast amounts of raw data directly!
  - Privacy settings are not very tight: deployed ε ranges from 0.5 to 8+

Opening up Private Data

- Deployments of LDP are still tightly controlled by aggregator
  - Google/MSR/Apple controls the code on your device
  - User reports typically sent over secure channel to aggregator
- Could there be a more 'open' implementation of LDP?
  - Users could publish their LDP outputs
  - Multiple aggregators could perform independent analysis
  - Would require a big change in architecture and adoption!

LDP data

Conclusions

- Private data release is a challenging and exciting problem!
- The model of Local Differential Privacy has been widely adopted
  - Initially for gathering counting statistics
  - LDP comes at a cost: need many more users than centralized model
- Lots of research excitement developing new applications
  - Practical impact of these is still to be seen
- Lots of opportunity for new work:
  - Designing optimal mechanisms for local differential privacy
  - Extend far beyond simple counts and marginals
  - Structured data: graphs, movement patterns
  - Unstructured data: text, images, video?