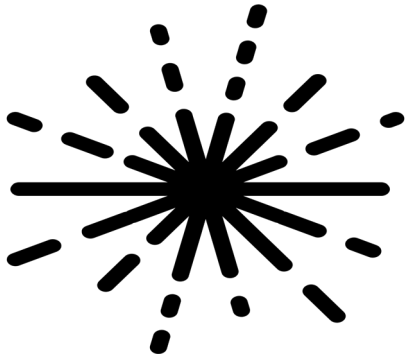


CS208: Applied Privacy for Data Science
Programming Frameworks

James Honaker, Priyanka Nanayakkara, Zachary Ratliff, Salil Vadhan
School of Engineering & Applied Sciences
Harvard University

March 31, 2025

Programming Frameworks for DP

Goal: make it easier for a data custodian or analyst to write programs that are DP, and be confident that they actually are DP.

Common approach (starting with PinQ [McSherry `09]):

- (Small) set of trusted DP subroutines: (Lap, Geo, ExpMech, …) only channel for info to flow from dataset to rest of program.
- Track privacy budget consumption: using composition of DP, with either a runtime monitor or static analysis.
- Allow "stable" data transformations: (recursively) track impact on privacy consumption.

Hierarchical Interactivity in PinQ

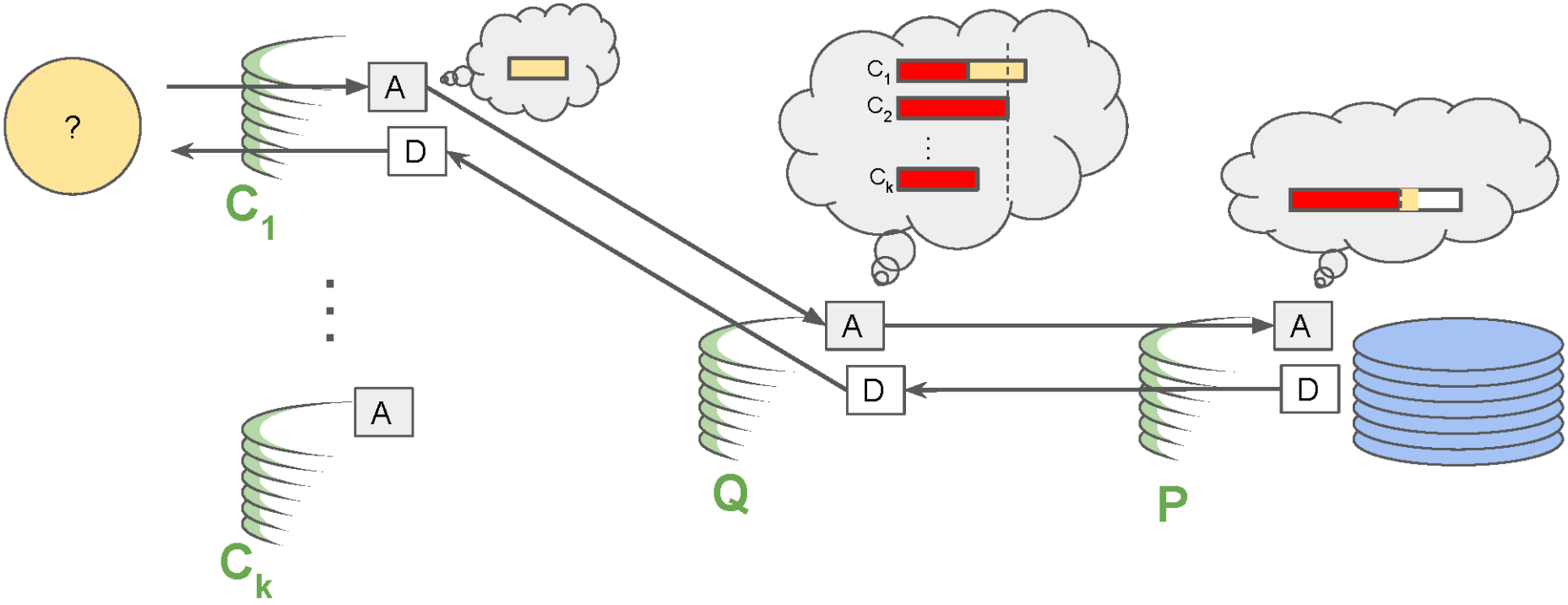
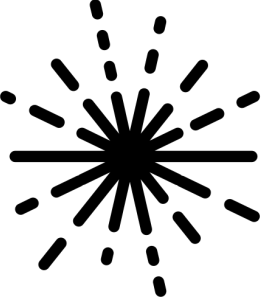


diagram from [Gaboardi et al. `24]



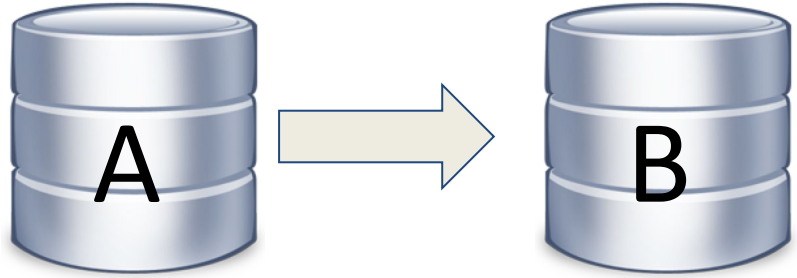
OpenDP Programming Framework

- Generality in privacy definitions & algorithms
  - Pure DP, approximate DP, concentrated DP, f-DP, etc.
  - Node-level privacy in graphs, user-level privacy in streams, etc.
- Generality in privacy calculus
  - Composition, amplification by subsampling, group privacy, etc.
- Safe extensions of framework with vetted contributions
  - Clear spec for each component's privacy-relevant properties
- Interactive DP algorithms as first-class citizens
  - Adaptive composition, sparse vector, etc.
  - Still in implementation!
- Implementation in Rust w/Python bindings

Transformations and Measurements

Transformations:

Function from data(sets) to data(sets).

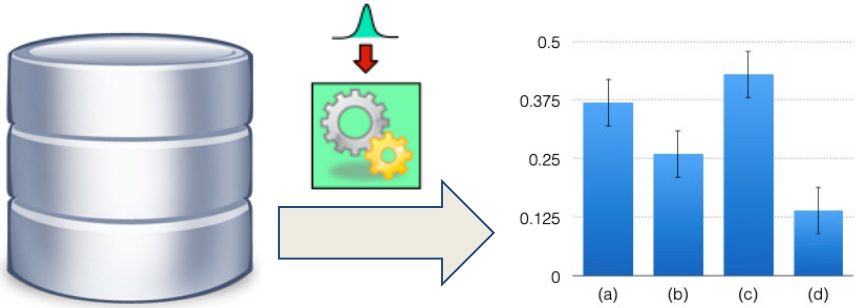


Transformation Attributes

- Input domain
- Input metric
- Output domain
- Output metric
- Function
- Stability relation

Measurements:

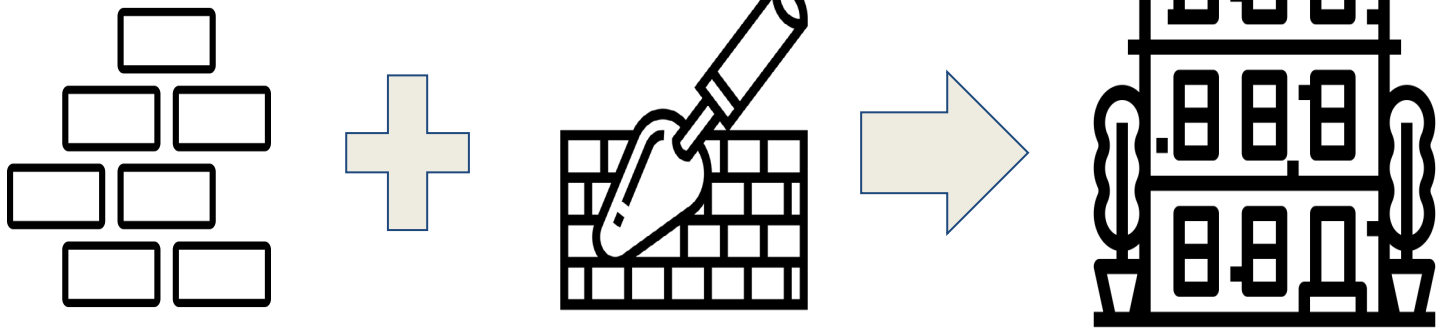
Randomized functions from data(sets) to outputs.



Measurement Attributes

- Input domain
- Input metric
- Output measure
- Function
- Privacy relation

Combinators: Chaining, Composition and Post-processing

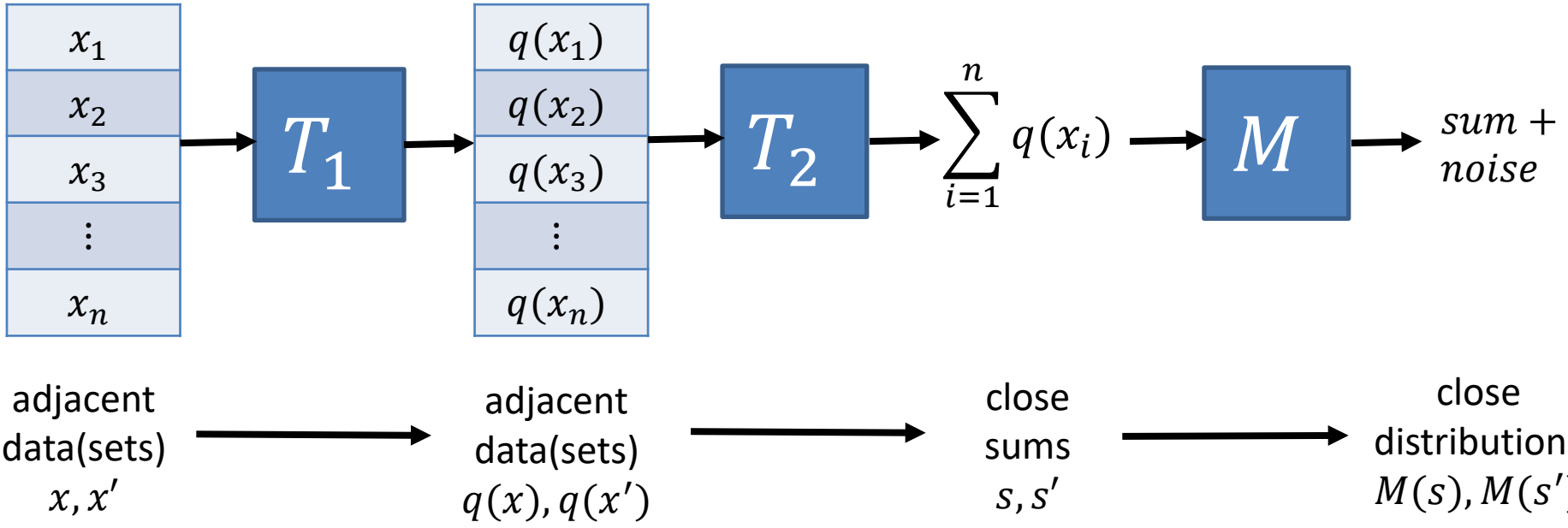


Measurements & Transformations

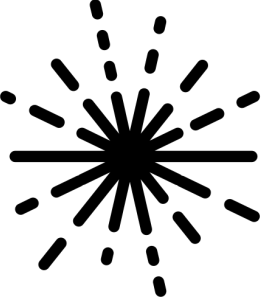
Combinators, e.g. Chaining, Composition, Post-processing

Complex DP programs

Chaining Transformations & Measurements



The result is a new measurement $M' = T_1 \gg T_2 \gg M$



Privacy calculus: privacy and stability relations

To implement a privacy calculus based on the idea of stability we have:

- privacy maps in measurements to capture several notions of privacy. E.g. DP, approx. DP, Renyi DP, zCDP, f-DP.
- stability maps in transformations to capture general aggregate operations. E.g. bounded joins.
- combination of these maps by means of combinators such as chaining and composition.

Measurement attributes

- Input domain
- Input metric
- ***Output measure***
- Function
- ***Privacy map***

Transformation attributes

- Input domain
- Input metric
- Output domain
- Output metric
- Function
- ***Stability map***

d_{out} =map(d_{in}) should imply:
if two inputs are "d_{in}-close",
then the corresponding outputs (or distributions) are "d_{out}-close".

Other Issues in Programming DP

- Multi-relational databases
  - Need to define input metric/adjacency carefully
  - Standard joins have unbounded stability constant, so need to truncate results or use “local sensitivity” approximations.
- Side-channel attacks
  - Info can be leaked through timing, approx. of real numbers, global state, exceptions, etc.
  - Constrain language & implementation to match model better.
- Verifying DP building blocks or more complex DP algs
  - Specialized programming languages.
  - Annotate programs with types to assist automated verification of DP.
  - Tradeoff between usability and expressiveness.
  - Now can even synthesize DP algorithms from examples!
- Guidance on Accuracy & Privacy Budgeting
  - Next time!
- Choice of Programming Model (e.g. SQL vs. MapReduce vs. R)
- Machine Learning Frameworks
  - Main ones are Opacus, Jax-Privacy, TensorFlow-Privacy

Takeaways

- Should not build DP applications from scratch, better to use an existing framework.

- We'll illustrate with OpenDP because we
  - Are most familiar with OpenDP
  - Hope that some of you become OpenDP contributors

- But feel free to use any existing tools out there that are a good fit for your project!