Making Privacy Technology Accessible: Benchmarks and Platforms

Michael Hay, Colgate University

The opinions expressed in this talk are my own and not those of the U.S. Census Bureau.
### Illustrative Example

<table>
<thead>
<tr>
<th>age</th>
<th>child race</th>
<th>household size</th>
<th>race householder</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>white</td>
<td>3</td>
<td>white</td>
</tr>
<tr>
<td>9</td>
<td>asian</td>
<td>4</td>
<td>white</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Goal**: Produce 2-way marginal between race of child and race of householder, computed under DP
Which DP algorithm should I use?

---

**Paper 1 [2012]**

<table>
<thead>
<tr>
<th>Epsilon</th>
<th>Algorithm A</th>
<th>Algorithm B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0125</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>0.025</td>
<td>10</td>
<td>1000</td>
</tr>
<tr>
<td>0.05</td>
<td>100</td>
<td>10000</td>
</tr>
<tr>
<td>0.1</td>
<td>1000</td>
<td>100000</td>
</tr>
</tbody>
</table>

**Paper 2 [2014]**

<table>
<thead>
<tr>
<th>Epsilon</th>
<th>Algorithm A</th>
<th>Algorithm B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>1000</td>
<td>10000</td>
</tr>
<tr>
<td>0.05</td>
<td>10000</td>
<td>100000</td>
</tr>
<tr>
<td>0.1</td>
<td>100000</td>
<td>1000000</td>
</tr>
<tr>
<td>0.50</td>
<td>1000000</td>
<td>10000000</td>
</tr>
</tbody>
</table>

---

*Simons Workshop on “Data Privacy: From Foundations to Applications” March 2019*
Analysis & Implementation

• Query: 2-way marginal between race of child and race of householder

• Analyst calculates sensitivity

• Analysts finds Laplace RNG

• Friend (DP expert) warns, “Watch out for floating-point precision attack.” [Mironov CCS12]

<table>
<thead>
<tr>
<th>age</th>
<th>race</th>
<th>household size</th>
<th>race household</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>white</td>
<td>3</td>
<td>white</td>
</tr>
<tr>
<td>29</td>
<td>asian</td>
<td>4</td>
<td>white</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Challenges to deployment

• Conflicting empirical results
• Lack of reference implementations
• Risk of subtle bugs (analysis + implementation)
Today’s talk

• Introduction

• DPBench: principled empirical evaluations of accuracy

• Ektelo: framework for private computation

• PrivateSQL: differentially private SQL query engine
Sound evaluation is hard

• Factors affecting performance: setting of epsilon, “amount” of data, tunable algorithm parameters, data pre-processing (cleaning, representation)

• Algorithms can be **data-dependent** because they *adapt* or *introduce statistical bias*.

• Examples: smooth sensitivity [Nissim STOC 2007], DAWA [Li VLDB 2014], Adaptive Grid [Qardaji ICDE 2013], StructureFirst [Xu VLDBJ 2013]
Principled evaluation of DP algorithms [SIGMOD16]

Companion website: dpcomp.org

Finding: no “universal” algorithms: best performance depends on task, input data, epsilon…
Sound evaluation is important!

• How to incentivize community participation?
  • Benchmarks
    *Successful in other communities* TPC-H, Trec, MNIST
  • Contests
    *NIST Differential Privacy Synthetic Data Challenge*
  • Reproducibility requirements
Outline

• Introduction

• DPBench: principled empirical evaluations of accuracy

• Ektelo: framework for private computation

• PrivateSQL: differentially private SQL query engine
Challenges of DP Deployment

• Successful deployments have required a team of privacy experts.

• Limited resources available
  • Few libraries, reference implementations or re-usable tools.
  • Frameworks like PINQ ensure privacy safe computation, but little guidance on accuracy
  • Implementations often start from scratch in arbitrary PL.

• Difficult for privacy non-experts to contribute.
Challenges of DP Deployment

- Privacy: Many points of failure
  - Code must be carefully vetted.

- Accuracy: Sophisticated algorithms needed
  - Need to think in new ways to get optimal error

- Context: data analysis workflows are *ad hoc*
  - Need toolkits, not monolithic algorithms.
εktelo execution framework

• Goal: simplify and accelerate development of efficient and accurate differentially private algorithms

• Ektelo supports a library of vetted operators.

• Operators encode (some) best practices from literature

• Differentially private computation expressed as a plan: a sequence of operator calls

Joint work with Gerome Miklau, Ashwin Machanavajhalla, Dan Zhang, Ryan McKenna, Ios Kotsogiannis
persons = ProtectedDataSource(persons_uri)
for level in geo_levels:
    geo_regions = SelectPartition(level)
    splits = persons.SplitByPartition(geo_regions)
    for persons_in_region in splits:
        x = persons_in_region.Vectorize()
        M = SelectMeasurementsHDMM(W)
        y = x.LaplaceMeasure(M, eps)
        x_hat = LeastSquares(M, y)
        ... additional post-processing ...

* Dan Kifer’s presentation “Consistency with External Knowledge: The TopDown Algorithm”
persons = ProtectedDataSource(persons_uri)
for level in geo_levels:
    geo_regions = SelectPartition(level)
    splits = persons.SplitByPartition(geo_regions)
    for persons_in_region in splits:
        x = persons_in_region.Vectorize()
        M = SelectMeasurementsHDMM(W)
        y = x.LaplaceMeasure(M, eps)
        x_hat = LeastSquares(M, y)
        ... additional post-processing ...

Top Down algorithm* implemented as Ektelo plan.

(Artistic rendering)

Plan executed by client, with calls to protected kernel that manages sensitive data
persons = ProtectedDataSource(persons_uri)
for level in geo_levels:
    geo_regions = SelectPartition(level)
    splits = persons.SplitByPartition(geo_regions)
    for persons_in_region in splits:
        x = persons_in_region.Vectorize()
        M = SelectMeasurementsHDMM(W)
        y = x.LaplaceMeasure(M, eps)
        x_hat = LeastSquares(M, y)
        ... additional post-processing ...

Transformations
persons = ProtectedDataSource(persons_uri)
for level in geo_levels:
    geo_regions = SelectPartition(level)
    splits = persons.SplitByPartition(geo_regions)
    for persons_in_region in splits:
        x = persons_in_region.Vectorize()
        M = SelectMeasurementsHDMM(W)
        y = x.LaplaceMeasure(M, eps)
        x_hat = LeastSquares(M, y)
        ... additional post-processing ...

Measurement Selection
persons = ProtectedDataSource(persons_uri)
for level in geo_levels:
    geo_regions = SelectPartition(level)
    splits = persons.SplitByPartition(geo_regions)
    for persons_in_region in splits:
        x = persons_in_region.Vectorize()
        M = SelectMeasurementsHDMM(W)
        y = x.LaplaceMeasure(M, eps)
        x_hat = LeastSquares(M, y)
        ... additional post-processing ...

Top Down algorithm* implemented as Ektelo plan.

(Artistic rendering)
Inference (and other post-processing)

... additional post-processing...

Top Down algorithm*
(Artistic rendering)

implemented as Ektelo plan.

persons = ProtectedDataSource(persons_uri)

for level in geo_levels:
  geo_regions = SelectPartition(level)
  for persons_in_region in splits:
    x = persons_in_region.Vectorize()
    y = x.LaplaceMeasure(M, eps)
    M = SelectMeasurementsHDMM(M)
    x_hat = LeastSquares(M, y)
...
persons = ProtectedDataSource(persons_uri)
for level in geo_levels:
    geo_regions = SelectPartition(level)
splits = persons.SplitByPartition(geo_regions)
for persons_in_region in splits:
    x = persons_in_region.Vectorize()
    M = SelectMeasurementsHDMM(W)
    y = x.LaplaceMeasure(M, eps)
    x_hat = LeastSquares(M, y)
    ... additional post-processing ...

Top Down algorithm* implemented as Ektelo plan.

(Artistic rendering)

Runs in trusted environment. Impacts sensitivity

Releases noisy measurements; Consumes privacy loss budget

Client-side; no impact on privacy
Operator classes

Transform
Filter, project, group, etc.

Query
Laplace mechanism

Query selection
Strategically choose query sets

Inference
Reconcile inconsistencies in noisy answers

Partition selection
Dimensionality reduction

Transform Partition selection Query selection
TV T-Vectorize PA AHPpartition SI Identity
TP V-SplitByPartition PG Grid ... Mult Weights Query SA AdaptiveGrids
HR Thresholding LM Vector Laplace SQ Quadtree
SW Worst-approx
SPB PrivBayes select
### Operator classes and instances

**Theorem:** if _red_ and _orange_ operators are vetted, then any Ektelop plan satisfies DP

<table>
<thead>
<tr>
<th>Transform</th>
<th>Query selection</th>
<th>Inference</th>
<th>Partition selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>TV T-Vectorize</td>
<td>SI Identity</td>
<td>LS Least squares</td>
<td>PA AHPpartition</td>
</tr>
<tr>
<td>TP V-SplitByPartition</td>
<td>ST Total</td>
<td>NLS Nneg Least squares</td>
<td>PG Grid</td>
</tr>
<tr>
<td>TR V-ReduceByPartition</td>
<td>SP Privelet</td>
<td>MW Mult Weights</td>
<td>PD Dawa</td>
</tr>
<tr>
<td></td>
<td>SH2 H2</td>
<td>HR Thresholding</td>
<td>PW Workload-based</td>
</tr>
<tr>
<td></td>
<td>SHB HB</td>
<td></td>
<td>PS Stripe(attr)</td>
</tr>
<tr>
<td></td>
<td>SG Greedy-H</td>
<td></td>
<td>PM Marginal(attr)</td>
</tr>
<tr>
<td></td>
<td>SU UniformGrid</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SA AdaptiveGrids</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SQ Quadtree</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SW Worst-approx</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SPB PrivBayes select</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Operators**

<table>
<thead>
<tr>
<th>Transform</th>
<th>Query selection</th>
<th>Inference</th>
<th>Partition selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>TV T-Vectorize</td>
<td>LM Vector Laplace</td>
<td>LS Least squares</td>
<td>PA AHPPartition</td>
</tr>
<tr>
<td>TP V-SplitByPartition</td>
<td>SI Identity</td>
<td>NLS Nneg Least squares</td>
<td>PG Grid</td>
</tr>
<tr>
<td>TR V-ReduceByPartition</td>
<td>ST Total</td>
<td>MW Multi Weights</td>
<td>PD Dawa</td>
</tr>
<tr>
<td>SH2 H2</td>
<td>SP Privulet</td>
<td>HR Thresholding</td>
<td>PW Workload-based</td>
</tr>
<tr>
<td>SHB HB</td>
<td>SH Greedy-H</td>
<td></td>
<td>PS Stripe(attr)</td>
</tr>
<tr>
<td>SG Greedy-H</td>
<td>SHB PrivBayes b</td>
<td></td>
<td>PM Marginal(attr)</td>
</tr>
<tr>
<td>SU UniformGrid</td>
<td>SH2 Quadtree</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA AdaptiveGrids</td>
<td>SQ Quadtree</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SW Worst-approx</td>
<td>SW Worst-approx</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SPB PrivBayes select</td>
<td>SPB PrivBayes select</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Algorithms as Ektelo plans**

<table>
<thead>
<tr>
<th>ID</th>
<th>Cite</th>
<th>Algorithm name</th>
<th>Plan signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[8]</td>
<td>Identity</td>
<td>SI LM</td>
</tr>
<tr>
<td>2</td>
<td>[39]</td>
<td>Privulet</td>
<td>SP LM LS</td>
</tr>
<tr>
<td>3</td>
<td>[17]</td>
<td>Hierarchical (H2)</td>
<td>SH2 LM LS</td>
</tr>
<tr>
<td>4</td>
<td>[34]</td>
<td>Hierarchical Opt (HB)</td>
<td>SHB LM LS</td>
</tr>
<tr>
<td>5</td>
<td>[22]</td>
<td>Greedy-H</td>
<td>SG LM LS</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>Uniform</td>
<td>ST LM LS</td>
</tr>
<tr>
<td>7</td>
<td>[15]</td>
<td>MWEM</td>
<td>I:( SW LM MW )</td>
</tr>
<tr>
<td>8</td>
<td>[42]</td>
<td>AHP</td>
<td>PA TR SI LM LS</td>
</tr>
<tr>
<td>9</td>
<td>[22]</td>
<td>DAWA</td>
<td>PD TR SG LM LS</td>
</tr>
<tr>
<td>10</td>
<td>[6]</td>
<td>Quadtree</td>
<td>SQ LM LS</td>
</tr>
<tr>
<td>11</td>
<td>[33]</td>
<td>UniformGrid</td>
<td>SU LM LS</td>
</tr>
<tr>
<td>13</td>
<td>NEW</td>
<td>DAWA-Striped</td>
<td>PS TP[ PD TR SG LM] LS</td>
</tr>
<tr>
<td>14</td>
<td>NEW</td>
<td>HB-Striped</td>
<td>PS TP[ SHB LM] LS</td>
</tr>
<tr>
<td>15</td>
<td>NEW</td>
<td>PrivBayesLS</td>
<td>SPB LM LS</td>
</tr>
<tr>
<td>16</td>
<td>NEW</td>
<td>MWEM variant b</td>
<td>I:( SW SH2 LM MW )</td>
</tr>
<tr>
<td>17</td>
<td>NEW</td>
<td>MWEM variant c</td>
<td>I:( SW LM NLS )</td>
</tr>
<tr>
<td>18</td>
<td>NEW</td>
<td>MWEM variant d</td>
<td>I:( SW SH2 LM NLS )</td>
</tr>
</tbody>
</table>

**Benefits**

- Reuse: existing algorithms implemented with reusable operators
- Reduces code verification effort
- Improved operator implementations
- New variants of algorithm easy to construct (improved accuracy!)

---

Simons Workshop on “Data Privacy: From Foundations to Applications” March 2019
Architecture for Private Computation?

• Separate concerns:
  • Transformations
  • Measurement selection
  • Measurement
  • Post-processing (consistency, synthetic data, inference)

• Benefits of modularity:
  • Reduce scope of privacy verification
  • Diverse contributors: relevant expertise differs by component
Outline

• Introduction

• DPBench: principled empirical evaluations of accuracy

• Ektelo: framework for private computation

• PrivateSQL: differentially private SQL query engine
Motivations for Private SQL

• Towards a declarative interface for query answering

• Complex queries over multi-relational data

• Privacy at multiple resolutions

Joint work with Gerome Miklau, Ashwin Machanavajjhala, Ios Kotsogiannis, Yuchao Tao, Xi He, Maryam Fanaeepour
Statistics Released by US Census Bureau

Census Summary File 1 (SF-1)
- “Number of males between 18 and 21 years old”, …
- “Number of people living in owned houses of size 3 where the householder is a married Hispanic male”, …

At all levels of geography (state, county, tract, block)
Complex Queries

• Linear queries on households

SELECT COUNT(*)
FROM ( SELECT hid, COUNT(*) AS CNT
FROM Persons p, (SELECT hid
FROM Persons p1, Persons p2
WHERE p1.hid = p2.hid
AND p1.Rel = 'householder'
AND p2.Rel = 'spouse'
AND ((p1.sex= 'M' AND p2.sex = 'F')
OR (p1.sex= 'F' AND p2.sex = 'M'))
GROUP BY hid) AS h
WHERE p.hid = h.hid AND p Rel = 'child'
AND p.Age < 18
GROUP BY hid)
WHERE CNT >= 1
Complex Queries

• Linear queries on households

```
SELECT COUNT(*)
FROM (
  SELECT hid, COUNT(*) AS CNT
  FROM Persons p, (SELECT hid
    FROM Persons p1, Persons p2
    WHERE p1.hid = p2.hid
    AND p1.Rel = 'householder'
    AND p2.Rel = 'spouse'
    AND ((p1.sex = 'M' AND p2.sex = 'F')
    OR (p1.sex = 'F' AND p2.sex = 'M'))
    GROUP BY hid)
  WHERE p.hid = h.hid AND p.Rel = 'child'
  AND p.Age < 18
  GROUP BY hid)
WHERE CNT >= 1
```
Complex Queries

• Queries on people living in households

SELECT COUNT(*)
FROM Person p
WHERE p.Age < 18 AND
  p.hID in (SELECT hID
              FROM Person p
              WHERE p.Rel = "householder"
              AND p.Race = "Asian")
Complex Queries

• Queries on people living in households

```sql
SELECT COUNT(*)
FROM Person p
WHERE p.Age < 18 AND p.hID in (SELECT hID
FROM Person p
WHERE p.Rel = "householder"
AND p.Race = "Asian")
```

Count of the number of people under 18 living in households with an Asian householder
Complex queries

- Degree distribution query or count of count histogram

```
SELECT cnt, COUNT(*)
FROM (SELECT hID, COUNT(*) as cnt
     FROM Person p
     GROUP BY hID)
GROUP BY cnt
ORDER BY cnt
```
Complex queries

- Degree distribution query or count of count histogram

```sql
SELECT cnt, COUNT(*)
FROM (SELECT hID, COUNT(*) as cnt
     FROM Person p
     GROUP BY hID)
GROUP BY cnt
ORDER BY cnt
```

For every household size, release the number of households of that size.
Motivations for Private SQL

• Complex queries over multi-relational data

• Privacy at multiple resolutions
Privacy requirement

• Title 13 Section 9

Neither the secretary nor any officer or employee …

… make any publication whereby the data furnished
by any particular establishment or individual
under this title can be identified …

• In some data products, only properties of people need to
be hidden, and in other products, properties of households
also need to be hidden.
Privacy at multiple resolutions

**Person-privacy**: hide properties of people

**Household-privacy**: hide properties of households and the people within them.

**Edge-privacy**: hide the presence of an edge

**Node-privacy**: hide the presence of a node and all edges incident to it.

**Event-privacy**: hide sensor reading

**Window-privacy**: hide readings in \((t-w, t]\)

**User-privacy**: hide all sensor readings
Goals of Private SQL

• *Automatically* generates differentially private code to accurately answer the queries specified in a high level language (SQL)

• Ensures a *fixed privacy budget* across all queries posed by the analyst.

• Enables privacy to be specified at *multiple resolutions*. 
1. Queries answered on live-DB one at a time

Example: FLEX [VLDB18]
- Deployed at Uber.
1. Queries answered on live-DB one at a time

Unbounded Privacy Loss
- Unless the system decides to shut off future queries, the privacy loss keeps increasing.

Inflexible privacy semantics (for Flex specifically)
- Hides any row in DB, but this may not align with privacy in particular context.

Other concerns: inconsistency between answers, side channel attacks
2. Query answering on a synthetic version of base tables

Examples:
HDMM [VLDB18], MWEM [NIPS12] ...
- Output a histogram tunes to query workload
PrivBayes [SIGMOD14], Private Synthetic Data using GANs [NIST Challenge 18]
- Generates a synthetic database in the same schema as input
2. Query answering on a synthetic version of base tables

No support for multi-relational tables

Joins computed on synthetic tables have very high error.
Defining privacy at multiple resolutions

**Edge-privacy:** hide the presence of an edge

**Node-privacy:** hide the presence of a node and all edges incident to it.

**Person-privacy:** hide properties of people

**Household-privacy:** hide properties of households and the people within them.

### Person

<table>
<thead>
<tr>
<th>ID</th>
<th>Sex</th>
<th>...</th>
<th>HID</th>
</tr>
</thead>
<tbody>
<tr>
<td>122</td>
<td>M</td>
<td>...</td>
<td>H6</td>
</tr>
<tr>
<td>123</td>
<td>F</td>
<td>...</td>
<td>H6</td>
</tr>
<tr>
<td>124</td>
<td>M</td>
<td>...</td>
<td>H7</td>
</tr>
<tr>
<td>125</td>
<td>M</td>
<td>...</td>
<td>H8</td>
</tr>
<tr>
<td>126</td>
<td>F</td>
<td>...</td>
<td>H8</td>
</tr>
</tbody>
</table>

### Household

<table>
<thead>
<tr>
<th>HID</th>
<th>...</th>
<th>Geo</th>
</tr>
</thead>
<tbody>
<tr>
<td>H6</td>
<td>...</td>
<td>CA</td>
</tr>
<tr>
<td>H7</td>
<td>...</td>
<td>FL</td>
</tr>
<tr>
<td>H8</td>
<td>...</td>
<td>NC</td>
</tr>
</tbody>
</table>
Multi-resolution privacy in PrivateSQL

- **Policy:** A specification of the base relation that is the *primary private object*.

- **Neighboring Databases:**
  - Add or remove a row $r$ in the primary private relation
  - Add or remove all rows in other tables that *transitively refer* to the row $r$ in the primary private relation
Multi-resolution privacy in PrivateSQL

**Person-privacy:**
- Person is the primary private relation
- Adding or removing an person record does not affect the household table.

**Household-privacy:**
- Household is the primary private relation
- Adding or removing a row $r$ from household removes all rows in person that refer to $r$ in household table.
The PrivateSQL system

Private Database

Privacy Budget

ε

Query Workload

Analyst
The PrivateSQL system

Private Database

View Selector

Query Workload

Views selected so that analyst queries are linear over views (no joins)

Example view

| age | race | household size | ...
|-----|------|----------------|-----
| 34  | white| 3              |     |
| 29  | asian| 4              |     |
|     |      |                |     |
The PrivateSQL system

- Private Database
- View Selector
- Synopsis Generator
- Query Workload

Synopsis may consist of a tuples or histograms

Analyst
The PrivateSQL system

Private Database

View Selector

Budget Allocator

Synopsis Generator

Query Workload

Split privacy budget across views
The PrivateSQL system

Private Database

View Selector

Budget Allocator

Synopsis Generator

Query Answering Engine

Query Workload

Purely postprocessing. No effect on privacy

Query 1

Result 1

Query 2

Result 3

Analyst
The PrivateSQL system

Quantifies the number of rows that change in the view if one row changes in the input.
Addressing view sensitivity

- View is complex SQL query; evaluation is hard
  [Arapinis et al. ICALP16]

Rule-based sensitivity bound calculator
(builds on PINQ, Flex, with new rules: joins on keys)
Addressing view sensitivity

- View is complex SQL query; evaluation is hard [Arapinis et al. ICALP16]

- Global sensitivity may be high / unbounded

Rule-based sensitivity bound calculator (builds on PINQ, Flex, with new rules: joins on keys)

<table>
<thead>
<tr>
<th>Example view</th>
</tr>
</thead>
<tbody>
<tr>
<td>age</td>
</tr>
<tr>
<td>34</td>
</tr>
<tr>
<td>29</td>
</tr>
<tr>
<td>...</td>
</tr>
</tbody>
</table>
Addressing view sensitivity

- View is complex SQL query; evaluation is hard
  [Arapinis et al. ICALP16]
  ➡️ Rule-based sensitivity bound calculator
  (builds on PINQ, Flex, with new rules: joins on keys)

- Global sensitivity may be high / unbounded
  ➡️ Truncate “outliers”

- Calculation depends on privacy resolution level
  (e.g., person vs. household)
  ➡️ View rewriting
The PrivateSQL system

For acyclic dependencies, different policies can be handled by appropriately rewriting the view.
Empirical evaluation

• Dataset: A synthetic census dataset
  – person(id, sex, gender, age, race, relationship, hid) and household(hid, location)
  – Restricted to the state of NC
  – 5.4 million people and 2.7 million households

• Queries: 3493 counting queries from the 2010 Summary file 1.
  – “Number of males between 18 and 21 years old.”
  – “Number of people living in owned houses of size 3 where the householder is a married Hispanic male.”

• Views: PrivateSQL generated 17 views
Overall Error

Privacy Budget: 1.0
Policy: Hiding a row in person table.

Outputting 0 for all queries gives relative error 1.

For queries with sufficiently large answers, the relative error is small.

Stratified by size of query answer

<table>
<thead>
<tr>
<th>Query Answer Size</th>
<th>Relative Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0 - 10^3$</td>
<td>$10^{-1}$</td>
</tr>
<tr>
<td>$10^3 - 10^4$</td>
<td>$10^{-2}$</td>
</tr>
<tr>
<td>$10^4 - 10^5$</td>
<td>$10^{-3}$</td>
</tr>
<tr>
<td>$&gt; 10^5$</td>
<td>$10^{-4}$</td>
</tr>
</tbody>
</table>

Overall Error Distribution

- Overall error is small for queries with sufficiently large answers.
- Outputting 0 for all queries gives a relative error of 1.
- Sensitivity bounds are used to avoid the need for smoothing.
- The de-correlation techniques of SensCalc support queries with correlated subqueries.
- Queries are grouped together based on their computed sensitivity.
- Flex systems support queries, whereas PrivateSQL does not.
- SensCalc provides an improvement of up to 4 orders of magnitude in error compared to Flex.
- SensCalc allows us to decouple improvements in error estimates from the choice of sensitivity computation.
- SensCalc is more general and provides a better support for workloads compared to the alternatives.
Comparison to one-query-at-a-time approach

Privacy Budget: 1.0
Competitor: A baseline based on FLEX [VLDB18]

Improvement over FLEX can be attributed to:
- Tighter sensitivity bounds
- Truncation instead of smoothing
- Better composition (across queries sharing view)
Key highlights of PrivateSQL

- **View Selection + Synopsis Generation** gets us away from one query at a time answering
  - Bounded privacy loss, consistent answers, avoids some side channel attacks

- **Privacy can be defined at multiple resolutions**
  - Able to specify a rich set of policies, and automatically rewrite views based on policy

- **Computing sensitivity for complex SQL queries is challenging**
  - Our techniques give an order of magnitude tighter bounds on sensitivity than prior work.

- **Modular architecture allows independent innovation in each component**
Some Open Questions

• More sophisticated truncation [Raskhodnikova FOCS 16; Chen, SIGMOD13]

• Theoretical characterization of bias-variance tradeoff of truncation

• Quantifying error in the answers
Summary

• Benchmarks can provide valuable insight and focus research community

• Modular architectures like Ektelo can simplify and accelerate algorithm development.

• PrivateSQL towards declarative interface for complex queries over multi-relational data
Thanks


Other related work:

[SIGMOD09] McSherry, “Privacy Integrated Queries”


[TODS17] Zhang et al, “PrivBayes: Private Data Release via Bayesian Networks”


[JPC17] Ebadi and Sands. Featherweight PINQ.


[SIGMOD13] Chen and Zhou, “Recursive mechanism: towards node differential privacy and unrestricted joins”