Composition, Verification, and Differential Privacy

Justin Hsu

University of Wisconsin–Madison
Lightning recap

Definition (Dwork, McSherry, Nissim, Smith (2006))
An algorithm is \((\varepsilon, \delta)\)-differentially private if, for every two adjacent inputs, the output distributions \(\mu_1, \mu_2\) satisfy:

\[
\text{for all sets of outputs } S, \Pr_{\mu_1}[S] \leq e^\varepsilon \cdot \Pr_{\mu_2}[S] + \delta
\]

Intuitively

Output can’t depend too much on any single individual’s data
Tremendous impact
Tremendous impact

<table>
<thead>
<tr>
<th>Google Scholar</th>
<th>About 2,860,000 results (0.04 sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Articles</td>
<td></td>
</tr>
</tbody>
</table>
| **Differential privacy**: A survey of results  
   C Dwork - International Conference on Theory and Applications of ..., 2008 - Springer |
| Abstract Over the past five years a new approach to privacy-preserving data analysis has born fruit [13, 18, 7, 19, 5, 37, 35, 8, 32]. This approach differs from much (but not all!) of the related literature in the statistics, databases, theory, and cryptography communities, in that ... |
| Cited by 2333  | Related articles  | All 32 versions  | Web of Science: 331  | Cite  | Save  | More |
**Tremendous impact**

<table>
<thead>
<tr>
<th>Scholor</th>
<th>About 2,860,000 results (0.04 sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Articles</td>
<td></td>
</tr>
<tr>
<td>Differential privacy: A survey of results C Dwork - International Conference on Theory and Applications of ..., 2008 - Springer</td>
<td></td>
</tr>
<tr>
<td>Abstract Over the past five years a new approach to privacy-preserving data analysis has born fruit [13, 18, 7, 19, 5, 37, 35, 8, 32]. This approach differs from much (but not all!) of the related literature in the statistics, databases, theory, and cryptography communities, in that ...</td>
<td></td>
</tr>
<tr>
<td>Cited by 2333 Related articles All 32 versions Web of Science: 331 Cite Save More</td>
<td></td>
</tr>
</tbody>
</table>
Tremendous impact

Differential privacy: A survey of results
C Dwork - International Conference on Theory and Applications of ..., 2008 - Springer
Abstract Over the past five years a new approach to privacy-preserving data analysis has born fruit [13, 18, 7, 19, 5, 37, 35, 8, 32]. This approach differs from much (but not all!) of the related literature in the statistics, databases, theory, and cryptography communities, in that ...
Cited by 2333  Related articles  All 32 versions  Web of Science: 331  Cite  Save  More

TPDP 2018 - Theory and Practice of Differential Privacy
Toronto, Canada - 15 October 2018 - part of CCS 2018
Why so popular? Elegant definition

Cleanly carve out a slice of privacy
  ► Mathematically formalize one kind of privacy
  ► “Your data” versus “data about you” (McSherry)

Simple and flexible
  ► Can establish property in isolation
  ► Achievable via rich variety of techniques
Why so popular? Theoretical features

Protects against worst-case scenarios

► Strong adversaries
► Colluding individuals
► Arbitrary side information

Rule out “blatantly” non-private algorithms

► Release data record at random: not private!
Above all, one reason...
Above all, one reason...

Composition!
1. Review and motivate composition properties
2. Case study: formal verification for privacy
3. Case study: advanced composition
A Quick Review:
Composition and Privacy
Theorem
Consider randomized algorithms $M: \mathcal{D} \rightarrow \text{Distr}(\mathcal{R})$ and $M': \mathcal{R} \times \mathcal{D} \rightarrow \text{Distr}(\mathcal{R}')$. If $M$ is $(\epsilon, \delta)$-private and for every $r \in \mathcal{R}$, $M'(r, -)$ is $(\epsilon', \delta')$-private, then the composition $r \sim M(d) \sim M'(r, d) \sim \text{return}(\text{out})$ is $(\epsilon + \epsilon', \delta + \delta')$-private.
Theorem
Consider randomized algorithms $M : D \rightarrow \text{Distr}(R)$ and $M' : R \times D \rightarrow \text{Distr}(R')$. If $M$ is $(\varepsilon, \delta)$-private and for every $r \in R$, $M'(r, -)$ is $(\varepsilon', \delta')$-private, then the composition

$$r \sim M(d); \text{out} \sim M'(r, d); \text{return}(\text{out})$$

is $(\varepsilon + \varepsilon', \delta + \delta')$-private.
Example: post processing

Privacy is preserved

- \( F \) is \((0, 0)\)-private: doesn't use private data
- Result is still \((\varepsilon, \delta)\)-private
Example: post processing

Privacy is preserved

- $F$ is $(0, 0)$-private: doesn’t use private data
- Result is still $(\varepsilon, \delta)$-private
Theorem

Consider randomized algorithms $M_1 : D \rightarrow \text{Distr}(R_1)$ and $M_2 : D \rightarrow \text{Distr}(R_2)$. If $M_1$ and $M_2$ are both $(\epsilon, \delta)$-private, then the parallel composition $(d_1, d_2) \leftarrow \text{split}(d_1); r_1 \sim M_1(d_1); r_2 \sim M_2(d_2); \text{return } (r_1, r_2)$ is $(\epsilon, \delta)$-private.
Theorem
Consider randomized algorithms $M_1 : D \rightarrow \text{Distr}(R_1)$ and $M_2 : D \rightarrow \text{Distr}(R_2)$. If $M_1$ and $M_2$ are both $(\epsilon, \delta)$-private, then the parallel composition

$$(d_1, d_2) \leftarrow \text{split}(d); r_1 \sim M_1(d_1); r_2 \sim M_2(d_2); \text{return}(r_1, r_2)$$

is $(\epsilon, \delta)$-private.
Example: local differential privacy

Each individual adds noise
- Split data among individuals
- Each individual computation achieves privacy

Central computation aggregates noisy data
- Post-processing
Group privacy

Bound output distance when multiple inputs differ

- Inputs databases differ in one individual: \((\varepsilon, 0)\)-privacy
- Inputs databases differ in \(k\) individuals: \((k\varepsilon, 0)\)-privacy

Cast privacy as Lipschitz continuity

- Composes well
- Not so clean for \((\varepsilon, \delta)\)-privacy...
Why You Might Care About Composition
Make definitions easier to use

Easier to prove property

- Privacy proofs are often straightforward
- Don’t need to unfold definition each time

More people can prove privacy

- Don’t need years of PhD training
Increase re-usability

Dramatically increases impact

- One useful algorithm can enable many others
- Repurpose for new, unforeseen applications
Increase re-usability

Dramatically increases impact

- One useful algorithm can enable many others
- Repurpose for new, unforeseen applications

Key algorithms used everywhere

- Laplace, Gaussian, Exponential mechanisms
- Sparse vector technique
- Private counters
- Subsampling
- ...
Build larger algorithms

Scale up private algorithms
- Construct complex private algorithms out of simple pieces
- Composition ensures result is still correct

Enables common toolboxes
- PINQ framework (McSherry)
- PSI project (see Salil’s talk)
Sign of a “good” definition

Not just about generalizing

- More general: must assume less about the pieces
- More specific: must prove more about the whole

Sweet spot between specific and general

- One way of probing robustness of definitions
Case Study: Verifying Privacy
Recap: verification setting

Dynamic

▸ Monitor program as it executes on particular input
▸ Raise error if it violates differential privacy

Static

▸ Take program (maybe written in special language)
▸ Check differential privacy on all inputs
Composition is crucial

Simplify verification task

- Trust a (small) collection of primitives
- Verify components separately

Enable automation

- Generally: enables faster/simpler verification
- So simple, a computer can do it
Privacy-integrated queries (PINQ)

C# library for private queries

- Proposed by Frank McSherry (2006)
- First verification technique for privacy

Dynamic analysis

- User writes PINQ query in C#
- Runtime monitors privacy budget as query runs
The Fuzz family of languages

History

- Reed and Pierce (2010), many subsequent extensions
- Programming language and custom type system

Main concept: function sensitivity

- Equip each type with a metric
- Types can express Lipschitz continuity
The Fuzz family of languages

History

▶ Reed and Pierce (2010), many subsequent extensions
▶ Programming language and custom type system

Main concept: function sensitivity

▶ Equip each type with a metric
▶ Types can express Lipschitz continuity

Example

\(!_k \sigma \rightarrow \tau\) is type of a \(k\)-sensitive function from \(\sigma\) to \(\tau\)
The Fuzz family of languages

Strengths

▶ Static analysis: don’t need to run program
▶ Typechecking/privacy checking can be automated
▶ Can express sequential and parallel composition
▶ Captures kind of group privacy (e.g., $(\varepsilon, 0)$-privacy)

Weaknesses

▶ Can’t verify programs where proof isn’t from composition
▶ Have to use a custom programming language
The Fuzz family of languages

Recent developments: extending to $(\varepsilon, \delta)$-privacy

- Idea: cast $(\varepsilon, \delta)$-privacy as sensitivity property
- For inputs that are two apart, output distributions are $(\varepsilon, \delta)$-related via some intermediate distribution
- So-called path metric construction
- Incorporate $(\varepsilon, \delta)$-privacy into Fuzz framework
Privacy as an approximate coupling

History
- Arose from work on verifying cryptographic protocols via game-based techniques, comparing pairs of hybrids
- Target more familiar, imperative programming language

Main concept: prove privacy by constructing a coupling
- Consider program run on two adjacent inputs
- Approximately couple sampling instructions
- Establish relation between coupled outputs
Privacy as an approximate coupling

**Strengths**
- Static analysis: don’t need to run program
- Can verify examples beyond composition
- Sparse vector, propose-test-release, ...
- No issue handling $(\varepsilon, \delta)$-privacy

**Weaknesses**
- Checks proof automatically, but doesn’t build proof
- Human expert must provide proof, manual process
Privacy as an approximate coupling

Recent developments: automate proof construction

- Encode proof requirement as a logical constraint
- Use techniques from program synthesis to find valid proofs
- Automatically verify sophisticated algorithms
- Sparse vector, report-noisy-max, between thresholds, ...
Brilliant collaborators
Case Study:
Advanced Composition
Recap: advanced composition

Sequentially compose $k$ mechanisms

- Each $(\varepsilon, \delta)$-private
- Basic analysis: result is $(k\varepsilon, k\delta)$-private
Recap: advanced composition

Sequentially compose $k$ mechanisms

- Each $(\varepsilon, \delta)$-private
- Basic analysis: result is $(k\varepsilon, k\delta)$-private

Better analysis

- Proposed by Dwork, Rothblum, and Vadhan (2010)
- For any $\delta'$, result is $(\varepsilon', k\delta + \delta')$-private for

$$
\varepsilon' = \varepsilon \sqrt{2k \ln(1/\delta')} + k\varepsilon(e^\varepsilon - 1)
$$
Extremely useful, but seems a bit off...

Intuitively

- Slow growth of $\varepsilon$ by increasing $\delta$ a bit more
- Privacy loss is "usually" much less than $k\varepsilon$

Composition is not so clean

- Best bounds if applied to a block of $k$ mechanisms
- Weaker if repeatedly applied pairwise
Improving the definitions: RDP and zCDP

History

- “Concentrated DP”: Dwork and Rothblum (2016)
- “Zero-Concentrated DP”: Bun and Steinke (2016)
- “Rényi DP”: Mironov (2017)
- Bound Rényi divergence between output distributions
- Refinement of \((\varepsilon, \delta)\)-privacy
Cleaner composition

**Theorem (Mironov (2017))**

Consider randomized algorithms $M : D \rightarrow Distr(R)$ and $M' : R \times D \rightarrow Distr(R')$. If $M$ is $(\alpha, \varepsilon)$-RDP and for every $r \in R$, $M'(r, -)$ is $(\alpha, \varepsilon')$-RDP, then the composition

$$r \sim M(d); \quad \text{out} \sim M'(r, d); \quad \text{return(out)}$$

is $(\alpha, \varepsilon + \varepsilon')$-RDP.

**Benefits**

- Composing pairwise or $k$-wise: same bounds
- Closure under post-processing
- Improved formulation of advanced composition
Enable formal verification

- Extensions of techniques for imperative languages
- Also works for programs in functional languages
- Opens the way to automated proofs
Wrapping Up
Success of privacy is a success of composition

Key factor behind high interest

- Make proofs easy enough for all
- The world has only so many TCS researchers
- Trivial to adapt privacy to new applications
- Ancillary benefit: enable computer verification
Composition matters!

Often not easy, but...

- Difference between a theoretically interesting definition, and a practically usable one
- Worth extra work and trouble to achieve

Compare to situation in cryptography

- Immense need for this technology, but poor composition
- Implementation still tricky, subtle errors
- “Don’t roll your own cryptography”
Trend towards “formal engineering”

Security is too hard for humans

- Want formal guarantees from our systems
- Rule out classes of attacks (subject to assumptions...)
- Principled construction of safe software

Compositional definitions are critical to this vision

- Needed to reason about large systems
- Only way to manage complexity
As I once heard from a famous systems researcher...
As I once heard from a famous systems researcher...

Without modularity, there is no civilization.
As I once heard from a famous systems researcher...

Without modularity, there is no civilization.

(Or at least, the going is pretty tough.)
Composition, Verification,
and Differential Privacy

Justin Hsu
University of Wisconsin–Madison