

# Advanced Probabilistic Couplings for Differential Privacy

Gilles Barthe, Noémie Fong, Marco Gaboardi,  
Benjamin Grégoire, Justin Hsu, Pierre-Yves Strub

October 25, 2016

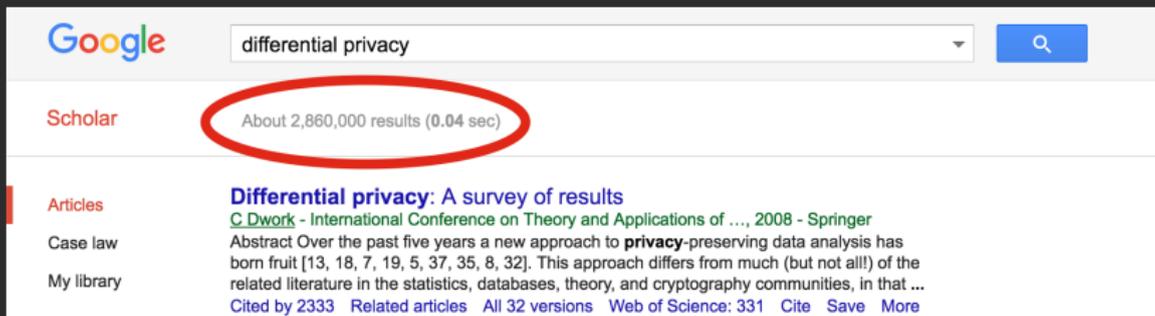
A new approach to formulating privacy goals: the risk to one's privacy, or in general, any type of risk . . . should not substantially increase as a result of participating in a statistical database.

This is captured by differential privacy.

— Cynthia Dwork

# Increasing interest

In research...



The screenshot shows a Google Scholar search interface. The search bar contains the text "differential privacy". Below the search bar, the results are displayed. The "Scholar" section shows "About 2,860,000 results (0.04 sec)", which is circled in red. Below this, the "Articles" section is visible, with a search result titled "Differential privacy: A survey of results" by C. Dwork, published in the "International Conference on Theory and Applications of ...", 2008 - Springer. The abstract for this article is provided, along with citation information: "Cited by 2333", "Related articles", "All 32 versions", "Web of Science: 331", "Cite", "Save", and "More".

Google

differential privacy

Scholar

About 2,860,000 results (0.04 sec)

Articles

**Differential privacy: A survey of results**

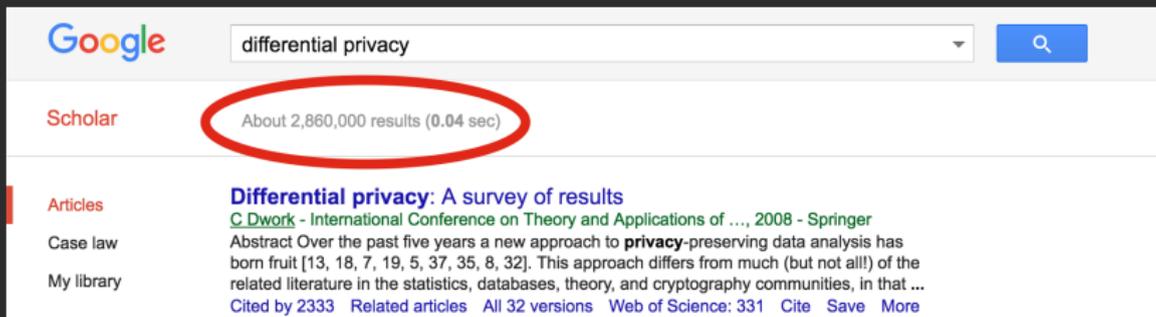
[C. Dwork](#) - International Conference on Theory and Applications of ..., 2008 - Springer

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](#)

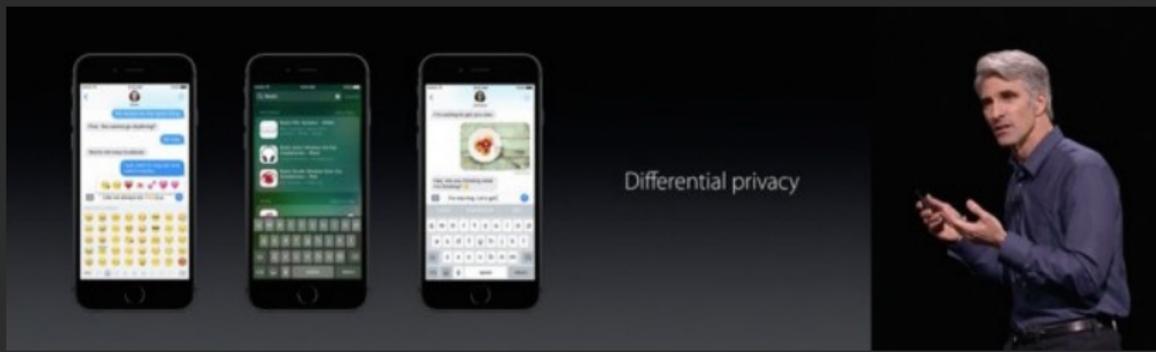
# Increasing interest

In research...



A screenshot of a Google Scholar search for "differential privacy". The search bar contains the text "differential privacy" and a magnifying glass icon. Below the search bar, the results are displayed. The first result is "About 2,860,000 results (0.04 sec)", which is circled in red. Below this, there are sections for "Articles", "Case law", and "My library". The "Articles" section shows a result titled "Differential privacy: A survey of results" by C. Dwork, published in the "International Conference on Theory and Applications of ..." in 2008, published by Springer. The abstract states: "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 ...". The article is cited by 2333, has 32 versions, and is listed in the Web of Science with 331 citations. There are links for "Cite", "Save", and "More".

... and beyond



A presentation slide featuring three smartphones on the left, each displaying a different app interface. The first phone shows a messaging app with a keyboard. The second phone shows a social media app with a green background. The third phone shows a social media app with a food image. To the right of the phones, the text "Differential privacy" is displayed. On the far right, a man in a blue shirt is gesturing with his hands, likely presenting the slide.

D

Alice

Bob

Xavier

Donna

Ernie

Algorithm

Pr [r]

ratio bounded

## Dwork, McSherry, Nissim, and Smith

Let  $\epsilon, \delta \geq 0$  be parameters, and suppose there is a binary adjacency relation  $Adj$  on  $D$ . A randomized algorithm  $M : D \rightarrow \mathbf{Distr}(R)$  is  $(\epsilon, \delta)$ -differentially private if for every set of outputs  $S \subseteq R$  and every pair of adjacent inputs  $d_1, d_2$ , we have

$$\Pr_{x \sim M(d_1)}[x \in S] \leq \exp(\epsilon) \cdot \Pr_{x \sim M(d_2)}[x \in S] + \delta.$$

## Dwork, McSherry, Nissim, and Smith

Let  $\epsilon, \delta \geq 0$  be parameters, and suppose there is a binary adjacency relation  $Adj$  on  $D$ . A randomized algorithm  $M : D \rightarrow \mathbf{Distr}(R)$  is  $(\epsilon, \delta)$ -differentially private if for every set of outputs  $S \subseteq R$  and every pair of adjacent inputs  $d_1, d_2$ , we have

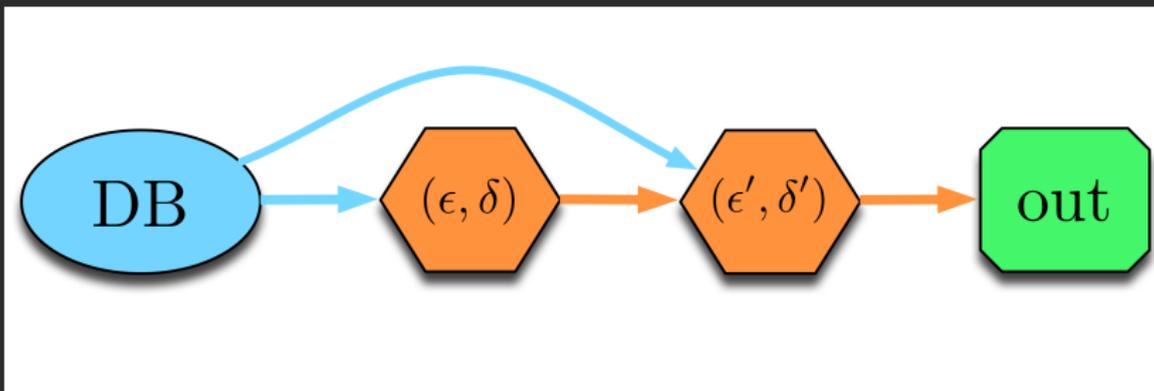
$$\Pr_{x \sim M(d_1)}[x \in S] \leq \exp(\epsilon) \cdot \Pr_{x \sim M(d_2)}[x \in S] + \delta.$$

How to formally verify?

Differential privacy is a:

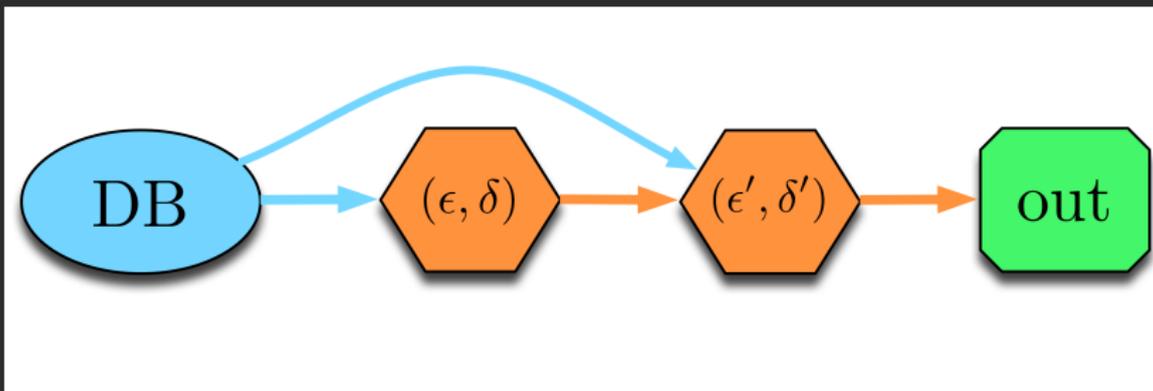
relational property of  
probabilistic programs.

## Composition properties



Program is  $(\epsilon + \epsilon', \delta + \delta')$ -private

## Composition properties



Program is  $(\epsilon + \epsilon', \delta + \delta')$ -private

### Formally

Consider randomized algorithms  $M : D \rightarrow \mathbf{Distr}(R)$  and  $M' : R \rightarrow D \rightarrow \mathbf{Distr}(R')$ . If  $M$  is  $(\epsilon, \delta)$ -private and for every  $r \in R$ ,  $M'(r)$  is  $(\epsilon', \delta')$ -private, then the composition is  $(\epsilon + \epsilon', \delta + \delta')$ -private:

$$r \xleftarrow{\$} M(d); \text{res} \xleftarrow{\$} M'(r, d); \text{return}(\text{res})$$

When privacy follows from composition



## When privacy follows from composition



(Linear types, refinement types, self products, relational Hoare logics, ...)

# When privacy **doesn't** follow from composition



# Complicated privacy proofs

## 3.1 Privacy Proof for Algorithm 1

We now prove the privacy of Algorithm 1. We break the proof down into two steps. To make the proof easier to understand, and more importantly, to point out what combinations likely caused the difference, we present the situation where all  $Q$ 's are processed. In the first step, we analyze the situation when the output is  $\perp$ , a length- $L$  vector  $(\perp, \dots, \perp)$ , indicating that all  $Q$ 's queries are tested to be below the threshold.

**LEMMA 1.** Let  $A$  be Algorithm 1. For any neighboring datasets  $D$  and  $D'$ , and any integer  $L$ , we have

$$\Pr[A(D) = \perp^L] \leq \epsilon + \Pr[A(D') = \perp^L].$$

PROOF. We have

$$\Pr[A(D) = \perp^L] = \int_{\perp^L} f_{D, \perp^L}(D, \perp^L) d\pi, \\ \text{where } f_{D, \perp^L}(D, \perp^L) = \Pr[\pi = \perp^L \mid \Pr[\pi(D) = \perp, \pi(T) = \perp], \text{ (I)} \\ \text{and } L = \{1, 2, \dots, L\}.$$

The probability of outputting  $\perp$  over  $D$  is the summation (or integral) of terms  $f_{D, \perp^L}(D, \perp^L)$ , each of which is the product of  $\Pr[\pi = \perp]$ , the probability that the threshold noise equals  $\pi$ , and  $\Pr[\text{Pr}[g(D, \pi) = \perp \mid \pi] = \perp]$ , the conditional probability that  $\perp$  is the output on  $D$  given that the threshold noise is  $\pi$ . (Note that given  $D, T$ , the queries, and  $\pi$ , whether one query results in  $\perp$  or not depends completely on the noise  $\pi$ , and is independent from whether any other query results in  $\perp$ , if we can prove

$$f_{D, \perp^L}(D, \perp^L) \leq \epsilon^L f_{D', \perp^L}(D', \perp^L), \quad (2)$$

then we have

$$\Pr[A(D) = \perp^L] = \int_{\perp^L} f_{D, \perp^L}(D, \perp^L) d\pi \\ \leq \int_{\perp^L} \epsilon^L f_{D', \perp^L}(D', \perp^L) d\pi \quad \text{from (2)} \\ = \epsilon^L \int_{\perp^L} f_{D', \perp^L}(D', \perp^L) d\pi \quad \text{let } \pi' = \pi + \Delta \\ = \epsilon^L \Pr[A(D') = \perp^L].$$

This proves the lemma. It remains to prove Eq (2). For any query  $q$ , because  $h(D) - \pi(D) \leq \Delta$  and that  $-q(D) \leq \Delta - q(D')$ , we have

$$\Pr[\pi(D) = \perp \mid \pi = \perp] \leq \Pr[\pi \leq \pi(T) - \epsilon(D) + \Delta] \\ \leq \Pr[\pi \leq \pi(T) - \Delta - q(D') + \Delta] \\ = \Pr[\pi(D) = \perp \mid \pi = \perp, \pi(T) = \perp + \Delta]. \quad (3)$$

With (3), we prove (2) as follows

$$f_{D, \perp^L}(D, \perp^L) = \Pr[\pi = \perp^L \mid \Pr[\pi(D) = \perp, \pi(T) = \perp] \\ \leq \epsilon^L \Pr[\pi = \perp \mid \Delta] \prod_{i=1}^L \Pr[\pi_i(D') = \perp, \pi_i(T) = \perp + \Delta] \\ = \epsilon^L f_{D', \perp^L}(D', \perp^L). \quad \square$$

That is, by using a noisy threshold, we are able to bound the probability ratio for all the negative query answers (i.e.,  $\perp$ 's) by  $\epsilon$ , no matter how many negative answers there are.

We can obtain a similar result for positive query answers in the same way.

$$\text{Let } f_{D, \perp^L}(D, \perp^L) = \Pr[\pi = \perp^L \mid \Pr[\pi(D) = \perp, \pi(T) = \perp + \Delta].$$

$$\text{We have } f_{D, \perp^L}(D, \perp^L) \leq \epsilon^L f_{D', \perp^L}(D', \perp^L), \text{ and then} \\ \Pr[A(D) = \perp^L] \leq \epsilon^L \Pr[A(D') = \perp^L].$$

This likely contributes to the misadventures behind Algorithms 3 and 8, which turn positive and negative answers exactly the same way. The problem is that while one is free to choose a bound positive or negative side, one cannot bound both.

We also observed that the proof of Lemma 1 will go through if noise is added to the query answers, i.e.,  $\pi_i = 0$ , because Eq (3) holds even when  $\pi_i = 0$ . It is likely because of this observation that Algorithm 3 adds no noise to query answers. However, when considering outcomes that include both positive answers ( $\perp$ 's) and negative answers ( $\perp$ 's), one has to add noise to the query answers, as we show below.

### THEOREM 2. Algorithm 2 is $\epsilon$ -DP.

PROOF. Consider any output vector  $\pi \in \{L, \perp, T\}^L$ . Let  $\pi_0 = (\pi_1, \dots, \pi_n) = \pi$ ,  $\pi' = (\pi'_1, \dots, \pi'_n) = \pi$ , and  $\pi'' = (\pi''_1, \dots, \pi''_n) = \pi$ . Clearly,  $\pi'' \leq \pi \leq \pi'$ .

$$\Pr[A(D) = \pi] = \int_{\pi} f_{D, \pi}(D, \pi) d\pi, \text{ where} \\ g(D, \pi) = \Pr[\pi = \pi \mid \Pr[\pi(D) = \pi, \pi(T) = \pi]] \leq \Pr[A(D) = \pi, \pi(T) = \pi].$$

We want to show that  $g(D, \pi) \leq \epsilon^L g(D', \pi + \Delta)$ . This suffices to prove that  $\Pr[A(D) = \pi] \leq \epsilon^L \Pr[A(D') = \pi]$ . Note that  $g(D, \pi)$  can be written as

$$g(D, \pi) = f_{D, \pi}(D, \pi) \prod_{i=1}^L \Pr[\pi_i(D) = \pi_i \mid \pi_i(T) = \pi_i + \Delta].$$

Following the proof of Lemma 1, we can show that  $f_{D, \pi}(D, \pi) \leq \epsilon^L f_{D', \pi + \Delta}(D', \pi + \Delta)$  and it remains to show

$$\prod_{i=1}^L \Pr[\pi_i(D) = \pi_i \mid \pi_i(T) = \pi_i + \Delta] \leq \prod_{i=1}^L \Pr[\pi_i(D') = \pi_i \mid \pi_i(T) = \pi_i + \Delta]. \quad (4)$$

Because  $\pi_0 = \text{Lap}(\pi''_0)$  and  $\|\pi(D) - \pi(D')\| \leq \Delta$ , we have

$$\Pr[\pi_i(D) = \pi_i \mid \pi_i(T) = \pi_i + \Delta] = \Pr[\pi_i \geq \pi_i(T) - \Delta - \pi_i(D)] \\ \leq \Pr[\pi_i \geq \pi_i(T) - \Delta - \pi_i(D')] \quad (5) \\ \leq \epsilon^L \Pr[\pi_i \geq \pi_i(T) + \Delta - \pi_i(D') + 2\Delta] \\ = \epsilon^L \Pr[\pi_i(D') = \pi_i \mid \pi_i(T) = \pi_i + \Delta].$$

Eq (4) is because  $\|\pi(D) - \Delta - \pi_i(D)\|$  and Eq (5) is from the Laplace distribution's property. This proves Eq (2).  $\square$

The basic idea of the proof is that when comparing  $g(D, \pi)$  with  $g(D', \pi + \Delta)$ , we can bound the probability ratio for all outputs of  $\perp$ , to no more than  $\epsilon$ , by using a noisy threshold, no matter how many such outputs there are. To bound the ratio for the  $T$ 's outputs to no more than  $\epsilon$ , we need to add sufficient Laplace noises, which should scale with  $\epsilon$ , to the number of positive outputs.

Now we turn to Algorithm 3 to clarify what are wrong with their privacy proofs and to give their DP properties.

Figure 1: A Selection of SVT Variants

### Input/Output shared by all SVT Algorithms

**Input:** A private database  $D$ , a stream of queries  $Q = \{q_1, q_2, \dots\}$  each with sensitivity no more than  $\Delta$ , either a sequence of thresholds  $T = \{T_1, T_2, \dots\}$  or a single threshold  $T$  (see footnote 1), and  $\epsilon$ , the maximum number of queries to be answered with  $T$ .

**Output:** A stream of answers  $a_1, a_2, \dots$ , where each  $a_i \in \{T, \perp, \Delta\}$  and  $R$  and  $R'$  denotes the set of all mid numbers.

#### Algorithm 1 An instantiation of the SVT proposed in this paper.

```

Input:  $D, Q, \Delta, T, \epsilon$ .
1:  $\mu = \text{Lap}(\frac{\Delta T}{\epsilon})$ ,  $\text{covert} = 0$ 
2: for each query  $q_i \in Q$  do
3:  $\pi_i = \text{Lap}(\frac{\Delta T}{\epsilon})$ 
4: if  $q_i(D) + \pi_i \geq T_i + \mu$  then
5:  $\text{Output } a_i = T_i$ 
6:  $\text{covert} = \text{covert} + 1$ , Abort if  $\text{covert} \geq \epsilon$ .
7: else
8:  $\text{Output } a_i = \perp$ .
9: end if
10: end for
    
```

#### Algorithm 2 SVT in Dwork and Roth 2014 [8].

```

Input:  $D, Q, \Delta, T, \epsilon$ .
1:  $\mu = \text{Lap}(\frac{\Delta T}{\epsilon})$ ,  $\text{covert} = 0$ 
2: for each query  $q_i \in Q$  do
3:  $\pi_i = \text{Lap}(\frac{\Delta T}{\epsilon})$ 
4: if  $q_i(D) + \pi_i \geq T_i + \mu$  then
5:  $\text{Output } a_i = T_i$ ,  $\mu = \text{Lap}(\frac{\Delta T}{\epsilon})$ 
6:  $\text{covert} = \text{covert} + 1$ , Abort if  $\text{covert} \geq \epsilon$ .
7: else
8:  $\text{Output } a_i = \perp$ .
9: end if
10: end for
    
```

#### Algorithm 3 SVT in Roth's 2011 Lecture Notes [15].

```

Input:  $D, Q, \Delta, T, \epsilon$ .
1:  $\mu = \text{Lap}(\frac{\Delta T}{\epsilon})$ ,  $\text{covert} = 0$ 
2: for each query  $q_i \in Q$  do
3:  $\pi_i = \text{Lap}(\frac{\Delta T}{\epsilon})$ 
4: if  $q_i(D) + \pi_i \geq T_i + \mu$  then
5:  $\text{Output } a_i = q_i(D) + \pi_i$ 
6:  $\text{covert} = \text{covert} + 1$ , Abort if  $\text{covert} \geq \epsilon$ .
7: else
8:  $\text{Output } a_i = \perp$ .
9: end if
10: end for
    
```

#### Algorithm 4 SVT in Lee and Clifton 2014 [13].

```

Input:  $D, Q, \Delta, T, \epsilon$ .
1:  $\mu = \text{Lap}(\frac{\Delta T}{\epsilon})$ ,  $\text{covert} = 0$ 
2: for each query  $q_i \in Q$  do
3:  $\pi_i = \text{Lap}(\frac{\Delta T}{\epsilon})$ 
4: if  $q_i(D) + \pi_i \geq T_i + \mu$  then
5:  $\text{Output } a_i = T_i$ 
6:  $\text{covert} = \text{covert} + 1$ , Abort if  $\text{covert} \geq \epsilon$ .
7: else
8:  $\text{Output } a_i = \perp$ .
9: end if
10: end for
    
```

#### Algorithm 5 SVT in Stockdale et al. 2014 [18].

```

Input:  $D, Q, \Delta, T$ .
1:  $\mu = \text{Lap}(\frac{\Delta T}{\epsilon})$ 
2: for each query  $q_i \in Q$  do
3:  $\pi_i = 0$ 
4: if  $q_i(D) + \pi_i \geq T_i + \mu$  then
5:  $\text{Output } a_i = T_i$ 
6:  $\text{Output } a_i = T_i$ 
7: else
8:  $\text{Output } a_i = \perp$ 
9: end if
10: end for
    
```

#### Algorithm 6 SVT in Chen et al. 2015 [1].

```

Input:  $D, Q, \Delta, T = T_1, T_2, \dots$ .
1:  $\mu = \text{Lap}(\frac{\Delta T}{\epsilon})$ 
2: for each query  $q_i \in Q$  do
3:  $\pi_i = \text{Lap}(\frac{\Delta T}{\epsilon})$ 
4: if  $q_i(D) + \pi_i \geq T_i + \mu$  then
5:  $\text{Output } a_i = T_i$ 
6:  $\text{Output } a_i = T_i$ 
7: else
8:  $\text{Output } a_i = \perp$ 
9: end if
10: end for
    
```

	Alg.1	Alg.2	Alg.3	Alg.4	Alg.5	Alg.6
Scale of threshold noise $\mu$	$\frac{\Delta T}{\epsilon}$					
Reset $\mu$ after each output of $T$	No	Yes	No	Yes	No	No
Scale of query noise $\pi_i$	$\frac{\Delta T}{\epsilon}$					
Outputting $\pi_i + \pi$ instead of $T$	No	No	Yes	No	No	No
Stop after coarsening $\epsilon/\epsilon'$	Yes	Yes	Yes	Yes	Yes	No
Privacy Property	$\epsilon$ -DP					

Figure 2: Differences among Algorithms 1-6.

# Complicated privacy proofs

## 3.1 Privacy Proof for Algorithm 1

We now prove the privacy of Algorithm 1. We break the proof down into two steps, to make the proof easier to understand, and, more importantly, to point out what conditions likely caused the different non-private variants of SVT to be proposed. In the first step, we analyze the situation where the output is  $\perp$ , a single  $\perp$  output  $(\perp, \dots, \perp)$ , indicating that all  $Q$  queries are tested to be below the threshold.

**LEMMA 1.** Let  $\Delta$  be Algorithm 1. For any neighboring datasets  $D$  and  $D'$ , and any integer  $t$ , we have

$$\Pr[A(D) = \perp] \leq \epsilon + \Pr[A(D') = \perp].$$

**PROOF.** We have

$$\Pr[A(D) = \perp] = \int_{\perp} f_{D, \Delta}(x) dx, \text{ where } f_{D, \Delta}(x) = \Pr[x \leq \tau] \Pr[x \leq \tau + \Delta], \text{ (1)}$$

$$\text{and } L = \{x \mid x \leq -\Delta\}.$$

The probability of outputting  $\perp$  over  $D$  is the summation (or integral) of terms  $f_{D, \Delta}(x)$ , each of which is the product of  $\Pr[x \leq \tau]$  (the probability that the threshold noise equals  $x$ ) and  $\Pr[x \leq \tau + \Delta]$  (the conditional probability that  $\perp$  is the output on  $D'$  given that the threshold noise is  $x$ ). Note that given  $D, T$ , the queries, and  $\mu$ , whether the query results in  $\perp$  or not depends completely on the noise  $x$  and is independent from whether any other query results in  $\perp$ . If we can prove

$$f_{D, \Delta}(x) \leq \epsilon + f_{D', \Delta}(x + \Delta), \text{ (2)}$$

then we have

$$\Pr[A(D) = \perp] = \int_{\perp} f_{D, \Delta}(x) dx \leq \int_{\perp} \epsilon + f_{D', \Delta}(x + \Delta) dx \text{ does (2)}$$

$$\leq \epsilon + \int_{\perp} f_{D', \Delta}(x) dx \text{ let } x' = x + \Delta$$

$$\leq \epsilon + \Pr[A(D') = \perp].$$

This proves the lemma. It remains to prove Eq (2). For any query  $q$ , because  $\delta \leq |D| - |D'| \leq \Delta$  and that  $-q(D) \leq \Delta - q(D')$ , we have

$$\Pr[q(D) = \perp] + \epsilon \leq \Pr[q(D') = \perp] + \epsilon + \Delta \leq \Pr[q(D') = \perp] + \epsilon + \Delta \text{ (3)}$$

With (3), we prove (2) as follows

$$f_{D, \Delta}(x) = \Pr[x = \mu] \Pr[x \leq \tau + \Delta] \leq \epsilon + \Pr[x = \mu + \Delta] \Pr[x \leq \tau + \Delta + \Delta] = \epsilon + f_{D', \Delta}(x + \Delta).$$

□

That is, by using a noisy threshold, we are able to bound the probability ratio for all the negative query answers (i.e.,  $\perp$ 's) by  $\epsilon$ , i.e., we make how many negative answers there are.

We can obtain a similar result for positive query answers in the same way.

$$\text{Let } f_{D, \Delta}(x) = \Pr[x = \mu] \prod_{i \in Q} \Pr[x_i(D) \geq \tau_i + \Delta].$$

$$\text{We have } f_{D, \Delta}(x) \leq \epsilon + f_{D', \Delta}(x - \Delta), \text{ and then } \Pr[A(D) = \top] \leq \epsilon + \Pr[A(D') = \top].$$

This likely contributes to the misinterpretation behind Algorithms 3 and 8, which turn positive and negative answers exactly the same way. The problem is that while one is free to choose a bound positive or negative side, one cannot bound both.

We also observed that the proof of Lemma 1 goes through if noise is added to the query answers, i.e.,  $(x_i, 0)$ , because Eq (2) holds even when  $x_i = 0$ . It is likely because of this observation that Algorithm 3 adds no noise to query answers. However, when considering outcomes that include both positive answers ( $\top$ 's) and negative answers ( $\perp$ 's), one has to add noise to the query answers, as we show below.

### THEOREM 2. Algorithm 2 is $\epsilon$ -DP.

**PROOF.** Consider any output vector  $o \in \{L, \perp, \top\}^Q$ . Let  $\mu = (0, \dots, 0)$ ,  $\tau = (T, \dots, T)$ , and  $\tau'_i = (\epsilon + \mu, \dots, \mu)$ . Clearly,  $\tau'_i \leq \mu$ . We have

$$\Pr[A(D) = o] = \int_{o} f_{D, \Delta}(x) dx, \text{ where } f_{D, \Delta}(x) = \Pr[x = \mu] \prod_{i \in Q} \Pr[x_i(D) + \tau'_i \leq \tau_i + \Delta].$$

We want to show that  $f_{D, \Delta}(x) \leq \epsilon + f_{D', \Delta}(x + \Delta)$ . This suffices to prove that  $\Pr[A(D) = o] \leq \epsilon + \Pr[A(D') = o]$ . Note that  $f_{D, \Delta}(x)$  can be written as

$$f_{D, \Delta}(x) = f_{D, \Delta}(x, \tau'_i) \prod_{i \in Q} \Pr[x_i(D) + \tau_i \geq \tau_i + \Delta].$$

Following the proof of Lemma 1, we can show that  $f_{D, \Delta}(x, \tau'_i) \leq \epsilon + f_{D', \Delta}(x + \Delta, \tau'_i)$  and it remains to show

$$\prod_{i \in Q} \Pr[x_i(D) + \tau_i \geq \tau_i + \Delta] \leq \prod_{i \in Q} \Pr[x_i(D') + \tau_i \geq \tau_i + \Delta]. \text{ (4)}$$

Because  $\mu_i = \text{Lap}(\tau'_i)$  and  $\|\tau'_i - \tau_i\|_2 \leq \Delta$ , we have

$$\Pr[x_i(D) + \tau_i \geq \tau_i + \Delta] = \Pr[x_i \geq \tau_i + \Delta - \mu_i] \leq \Pr[x_i \geq \tau_i + \Delta - \mu_i] \leq \epsilon + \Pr[x_i \geq \tau_i + \Delta - \mu_i + \Delta] = \epsilon + \Pr[x_i(D') + \tau_i \geq \tau_i + \Delta].$$

Eq (4) is because  $\|\tau'_i - \tau_i\|_2 \leq \Delta - \mu_i(D)$ , and Eq (4) is from the Laplace distribution's property. This proves Eq (4). □

The basic idea of the proof is that when comparing  $f_{D, \Delta}(x)$  with  $f_{D', \Delta}(x + \Delta)$ , we can bound the probability ratio for all outputs of  $\perp$  to be more than  $\epsilon$  by using a noisy threshold, so neither how many such outputs there are. To bound the ratio for the  $\top$  outputs to be more than  $\epsilon$ , we need to add sufficient Laplace noises, which should scale with  $\epsilon$ , the number of positive outputs.

Now we turn to Algorithm 3 to clarify what was wrong with their privacy proofs and to give their DP properties.

Figure 1: A Selection of SVT Variants

### Input/Output shared by all SVT Algorithms

**Input:** A private database  $D$ , a stream of queries  $Q = \{q_1, q_2, \dots\}$  each with sensitivity no more than  $\Delta$ , either a sequence of thresholds  $T = \{T_1, T_2, \dots\}$  or a single threshold  $T$  (see footnote 1), and  $\epsilon$ , the maximum number of queries to be answered with  $\top$ .

**Output:** A stream of answers  $a_1, a_2, \dots$ , where each  $a_i \in \{\perp, \top, \Delta\}$  and  $R$  and  $R'$  denotes the set of all real numbers.

**Algorithm 1** An instantiation of the SVT proposed in this paper.

**Input:**  $D, Q, \Delta, T, \epsilon$ .  
 1:  $\mu = \text{Lap}(\tau)$ ,  $\text{count} = 0$   
 2: **for** each query  $q_i \in Q$  **do**  
 3:  $x_i = \text{Lap}(\Delta \cdot \mu)$   
 4: **if**  $q_i(D) + x_i \geq T_i + \mu$  **then**  
 5:  $\text{Output } a_i = \top$   
 6:  $\text{count} = \text{count} + 1$ , **Abort if**  $\text{count} \geq \epsilon$ .  
 7: **else**  
 8:  $\text{Output } a_i = \perp$ .  
 9: **end if**  
 10: **end for**

**Algorithm 2 SVT in Dwork and Roth 2014 [8].**

**Input:**  $D, Q, \Delta, T, \epsilon$ .  
 1:  $\mu = \text{Lap}(\tau)$ ,  $\text{count} = 0$   
 2: **for** each query  $q_i \in Q$  **do**  
 3:  $x_i = \text{Lap}(\Delta \cdot \mu)$   
 4: **if**  $q_i(D) + x_i \geq T_i + \mu$  **then**  
 5:  $\text{Output } a_i = \top$ ,  $\mu = \text{Lap}(\Delta \cdot \mu)$   
 6:  $\text{count} = \text{count} + 1$ , **Abort if**  $\text{count} \geq \epsilon$ .  
 7: **else**  
 8:  $\text{Output } a_i = \perp$ .  
 9: **end if**  
 10: **end for**

**Algorithm 3 SVT in Roth's 2011 Lecture Notes [15].**

**Input:**  $D, Q, \Delta, T, \epsilon$ .  
 1:  $\mu = \text{Lap}(\tau)$ ,  $\text{count} = 0$   
 2: **for** each query  $q_i \in Q$  **do**  
 3:  $x_i = \text{Lap}(\Delta \cdot \mu)$   
 4: **if**  $q_i(D) + x_i \geq T_i + \mu$  **then**  
 5:  $\text{Output } a_i = q_i(D) + \mu$   
 6:  $\text{count} = \text{count} + 1$ , **Abort if**  $\text{count} \geq \epsilon$ .  
 7: **else**  
 8:  $\text{Output } a_i = \perp$ .  
 9: **end if**  
 10: **end for**

**Algorithm 4 SVT in Lee and Clifton 2014 [13].**

**Input:**  $D, Q, \Delta, T, \epsilon$ .  
 1:  $\mu = \text{Lap}(\tau)$ ,  $\text{count} = 0$   
 2: **for** each query  $q_i \in Q$  **do**  
 3:  $x_i = \text{Lap}(\Delta \cdot \mu)$   
 4: **if**  $q_i(D) + x_i \geq T_i + \mu$  **then**  
 5:  $\text{Output } a_i = \top$   
 6:  $\text{count} = \text{count} + 1$ , **Abort if**  $\text{count} \geq \epsilon$ .  
 7: **else**  
 8:  $\text{Output } a_i = \perp$ .  
 9: **end if**  
 10: **end for**

**Algorithm 5 SVT in Stockdale et al. 2014 [18].**

**Input:**  $D, Q, \Delta, T$ .  
 1:  $\mu = \text{Lap}(\tau)$   
 2: **for** each query  $q_i \in Q$  **do**  
 3:  $x_i = \mu$   
 4: **if**  $q_i(D) + x_i \geq T_i + \mu$  **then**  
 5:  $\text{Output } a_i = \top$   
 6: **else**  
 7:  $\text{Output } a_i = \perp$   
 8: **end if**  
 9: **end if**  
 10: **end for**

**Algorithm 6 SVT in Chen et al. 2015 [1].**

**Input:**  $D, Q, \Delta, T, \tau, T', T_1, \dots$ .  
 1:  $\mu = \text{Lap}(\tau)$   
 2: **for** each query  $q_i \in Q$  **do**  
 3:  $x_i = \text{Lap}(\Delta \cdot \mu)$   
 4: **if**  $q_i(D) + x_i \geq T_i + \mu$  **then**  
 5:  $\text{Output } a_i = \top$   
 6: **else**  
 7:  $\text{Output } a_i = \perp$   
 8: **end if**  
 9: **end if**  
 10: **end for**

	Alg. 1	Alg. 2	Alg. 3	Alg. 4	Alg. 5	Alg. 6
Scale of threshold noise $\mu$	$2\Delta/\epsilon$	$2\Delta/\epsilon$	$2\Delta/\epsilon$	$4\Delta/\epsilon$	$2\Delta/\epsilon$	$2\Delta/\epsilon$
Reset $\mu$ after each output of $\top$	No	Yes	No	Yes	No	No
Scale of query noise $x_i$	$4\Delta/\epsilon$	$4\Delta/\epsilon$	$4\Delta/\epsilon$	$2\Delta/\epsilon$	$4\Delta/\epsilon$	$2\Delta/\epsilon$
Outputting $\mu + x_i$ instead of $\top$	No	Yes	No	Yes	No	No
Stop after outputting $\epsilon \cdot \mu$	Yes	Yes	Yes	Yes	Yes	No
Privacy Property	$\epsilon$ -DP	$\epsilon$ -DP	$\epsilon$ -DP	$(13\epsilon/\epsilon)$ -DP	$\epsilon$ -DP	$\epsilon$ -DP

Figure 2: Differences among Algorithms 1-6.

Lyu, Su, Dong

# How to verify these proofs?

## Recent progress (2016)

Differential privacy  $\approx$  Approximate couplings

## Recent progress (2016)

Differential privacy  $\approx$  Approximate couplings

Approximate couplings  $\approx$  Proofs in the logic apRHL

## Recent progress (2016)

Differential privacy  $\approx$  Approximate couplings

Approximate couplings  $\approx$  Proofs in the logic apRHL

Only proofs beyond composition  
for  $(\epsilon, 0)$ -privacy

## Enhance the logic

New coupling constructions



New proof rules



Richer formal proofs of privacy

Our work: formal privacy proofs with:

Accuracy-dependent privacy

Advanced composition

Adaptive inputs

Our work: formal privacy proofs with:

Accuracy-dependent privacy

Advanced composition

Adaptive inputs



# A crash course: the program logic apRHL [BKOZB]

Imperative language with random sampling

$$x \stackrel{\$}{\leftarrow} \mathcal{L}_\epsilon(e)$$

# A crash course: the program logic apRHL [BKOZB]

Imperative language with random sampling

$$x \stackrel{\$}{\leftarrow} \mathcal{L}_\epsilon(e)$$

approximate probabilistic Relational Hoare Logic

$$\vdash \{\Phi\} c_1 \sim_{(\epsilon, \delta)} c_2 \{\Psi\}$$

# A crash course: the program logic apRHL [BKOZB]

Imperative language with random sampling

$$x \stackrel{\$}{\leftarrow} \mathcal{L}_\epsilon(e)$$

approximate probabilistic Relational Hoare Logic

$$\vdash \{ \Phi \} c_1 \sim_{(\epsilon, \delta)} c_2 \{ \Psi \}$$

Non-probabilistic, relational ( $x_1 = x_2$ )

# A crash course: the program logic apRHL [BKOZB]

Imperative language with random sampling

$$x \stackrel{\$}{\leftarrow} \mathcal{L}_\epsilon(e)$$

approximate probabilistic Relational Hoare Logic

$$\vdash \{\Phi\} c_1 \sim_{(\epsilon, \delta)} c_2 \{\Psi\}$$

Numeric index

## Approximate couplings [BKOZB, BO]

### Definition

Let  $R \subseteq A \times A$  be a relation and  $\epsilon, \delta \geq 0$ . Two distributions  $\mu_1, \mu_2 \in \mathbf{Distr}(A)$  are related by an  $(\epsilon, \delta)$ -approximate coupling with support  $R$  if there exists  $\mu_L, \mu_R \in \mathbf{Distr}(A \times A)$  with:

## Approximate couplings [BKOZB, BO]

### Definition

Let  $R \subseteq A \times A$  be a relation and  $\epsilon, \delta \geq 0$ . Two distributions  $\mu_1, \mu_2 \in \mathbf{Distr}(A)$  are related by an  $(\epsilon, \delta)$ -approximate coupling with support  $R$  if there exists  $\mu_L, \mu_R \in \mathbf{Distr}(A \times A)$  with:

- ▶ support in  $R$  ;

# Approximate couplings [BKOZB, BO]

## Definition

Let  $R \subseteq A \times A$  be a relation and  $\epsilon, \delta \geq 0$ . Two distributions  $\mu_1, \mu_2 \in \mathbf{Distr}(A)$  are related by an  $(\epsilon, \delta)$ -approximate coupling with support  $R$  if there exists  $\mu_L, \mu_R \in \mathbf{Distr}(A \times A)$  with:

- ▶ support in  $R$  ;
- ▶  $\pi_1(\mu_L) = \mu_1$  and  $\pi_2(\mu_R) = \mu_2$  ;

# Approximate couplings [BKOZB, BO]

## Definition

Let  $R \subseteq A \times A$  be a relation and  $\epsilon, \delta \geq 0$ . Two distributions  $\mu_1, \mu_2 \in \mathbf{Distr}(A)$  are related by an  $(\epsilon, \delta)$ -approximate coupling with support  $R$  if there exists  $\mu_L, \mu_R \in \mathbf{Distr}(A \times A)$  with:

- ▶ support in  $R$  ;
- ▶  $\pi_1(\mu_L) = \mu_1$  and  $\pi_2(\mu_R) = \mu_2$  ;
- ▶ for every  $S \subseteq A \times A$ ,

$$\Pr_{z \sim \mu_L}[z \in S] \leq \exp(\epsilon) \cdot \Pr_{z \sim \mu_R}[z \in S] + \delta$$

# Approximate couplings [BKOZB, BO]

## Definition

Let  $R \subseteq A \times A$  be a relation and  $\epsilon, \delta \geq 0$ . Two distributions  $\mu_1, \mu_2 \in \mathbf{Distr}(A)$  are related by an  $(\epsilon, \delta)$ -approximate coupling with support  $R$  if there exists  $\mu_L, \mu_R \in \mathbf{Distr}(A \times A)$  with:

- ▶ support in  $R$  ;
- ▶  $\pi_1(\mu_L) = \mu_1$  and  $\pi_2(\mu_R) = \mu_2$  ;
- ▶ for every  $S \subseteq A \times A$ ,

$$\Pr_{z \sim \mu_L}[z \in S] \leq \exp(\epsilon) \cdot \Pr_{z \sim \mu_R}[z \in S] + \delta$$

# Approximate couplings [BKOZB, BO]

## Definition

Let  $R \subseteq A \times A$  be a relation and  $\epsilon, \delta \geq 0$ . Two distributions  $\mu_1, \mu_2 \in \mathbf{Distr}(A)$  are related by an  $(\epsilon, \delta)$ -approximate coupling with support  $R$  if there exists  $\mu_L, \mu_R \in \mathbf{Distr}(A \times A)$  with:

- ▶ support in  $R$  ;
- ▶  $\pi_1(\mu_L) = \mu_1$  and  $\pi_2(\mu_R) = \mu_2$  ;
- ▶ for every  $S \subseteq A \times A$ ,

$$\Pr_{z \sim \mu_L}[z \in S] \leq \exp(\epsilon) \cdot \Pr_{z \sim \mu_R}[z \in S] + \delta$$

Write:  $\mu_1 \quad R_{(\epsilon, \delta)}^\# \quad \mu_2$

## Interpreting judgments

$$\vdash \{ \Phi \} \quad c_1 \sim_{(\epsilon, \delta)} c_2 \quad \{ \Psi \}$$

## Interpreting judgments

$$\vdash \{ \Phi \} \quad c_1 \sim_{(\epsilon, \delta)} c_2 \quad \{ \Psi \}$$

Two memories related by  $\Phi$

## Interpreting judgments

$$\vdash \{ \Phi \} \quad c_1 \sim_{(\epsilon, \delta)} c_2 \quad \{ \Psi \}$$

Two memories related by  $\Phi$



Two distributions related by  $\Psi_{(\epsilon, \delta)}^\#$

## Differential privacy in apRHL

$$\vdash \{Adj(d_1, d_2)\} \ c \sim_{(\epsilon, \delta)} \ c \ \{res_1 = res_2\}$$

## Differential privacy in apRHL

$$\vdash \{Adj(d_1, d_2)\} \ c \sim_{(\epsilon, \delta)} \ c \ \{res_1 = res_2\}$$

$(\epsilon, \delta)$ -differential privacy

## Proof rules

Proof rule  $\approx$  Recipe to combine couplings

Proof rule  $\approx$  Recipe to combine couplings

Sequence rule  $\approx$  standard composition of privacy

$$\text{SEQ} \frac{\vdash \{\Phi\} c_1 \sim_{(\epsilon, \delta)} c_2 \{\Psi\} \quad \vdash \{\Psi\} c'_1 \sim_{(\epsilon', \delta')} c'_2 \{\Theta\}}{\vdash \{\Phi\} c_1; c'_1 \sim_{(\epsilon + \epsilon', \delta + \delta')} c_2; c'_2 \{\Theta\}}$$

Proof rule  $\approx$  Recipe to combine couplings

Sequence rule  $\approx$  standard composition of privacy

$$\text{SEQ} \frac{\vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta)} \ c_2 \ \{\Psi\} \quad \vdash \{\Psi\} \ c'_1 \sim_{(\epsilon', \delta')} \ c'_2 \ \{\Theta\}}{\vdash \{\Phi\} \ c_1; c'_1 \sim_{(\epsilon + \epsilon', \delta + \delta')} \ c_2; c'_2 \ \{\Theta\}}$$

Our work: formal privacy proofs with:

Accuracy-dependent privacy

Advanced composition

Adaptive inputs



# Accuracy-dependent privacy



# Accuracy-dependent privacy

## Rough intuition

- ▶ Think of  $\delta$  in  $(\epsilon, \delta)$ -privacy as **failure probability**
- ▶ “Algorithm is private except with small probability  $\delta$ ”
- ▶ “If the noise added is not too large, then . . .”

## Similar to **up-to-bad** reasoning

- ▶ Common tool in crypto proofs
- ▶ “If **bad event** doesn't happen, then protocol is safe”

## In apRHL: up-to-bad rule

$$\text{U}_{\text{TB}} \frac{\begin{array}{l} \vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta)} c_2 \ \{\neg \Psi \langle 1 \rangle \rightarrow x_1 = x_2\} \\ \models m \in \Theta \implies \Pr_{\llbracket c_1 \rrbracket(m_1)} [\Psi \langle 1 \rangle] < \delta' \end{array}}{\vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta + \delta')} c_2 \ \{x_1 = x_2\}}$$

## In apRHL: up-to-bad rule

$$\text{U}_{\text{TB}} \frac{\begin{array}{c} \vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta)} c_2 \ \{\neg \Psi \langle 1 \rangle \rightarrow x_1 = x_2\} \\ \models m \in \Theta \implies \Pr_{\llbracket c_1 \rrbracket(m_1)} [\Psi \langle 1 \rangle] < \delta' \end{array}}{\vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta + \delta')} c_2 \ \{x_1 = x_2\}}$$

### Notes

- ▶  $\Psi \langle 1 \rangle$  is “bad event”, only mentions  $c_1$

## In apRHL: up-to-bad rule

$$\text{UTB} \frac{\begin{array}{c} \vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta)} c_2 \ \{\neg \Psi \langle 1 \rangle \rightarrow x_1 = x_2\} \\ \models m \in \Theta \implies \Pr_{\llbracket c_1 \rrbracket(m_1)} [\Psi \langle 1 \rangle] < \delta' \end{array}}{\vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta + \delta')} c_2 \ \{x_1 = x_2\}}$$

### Notes

- ▶  $\Psi \langle 1 \rangle$  is “bad event”, only mentions  $c_1$
- ▶ If bad event doesn't happen, have **privacy**

## In apRHL: up-to-bad rule

$$\text{UTB} \frac{\begin{array}{l} \vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta)} c_2 \ \{\neg \Psi \langle 1 \rangle \rightarrow x_1 = x_2\} \\ \models m \in \Theta \implies \Pr_{\llbracket c_1 \rrbracket(m_1)} [\Psi \langle 1 \rangle] < \delta' \end{array}}{\vdash \{\Phi\} \ c_1 \sim_{(\epsilon, \delta + \delta')} c_2 \ \{x_1 = x_2\}}$$

### Notes

- ▶  $\Psi \langle 1 \rangle$  is “bad event”, only mentions  $c_1$
- ▶ If bad event doesn't happen, have privacy
- ▶ **Bound probability** of  $\Psi$  after  $c_1$



*Advanced*  
**COMPOSITION**

# Advanced composition theorem

Compose  $n$  mechanisms, each  $(\epsilon, \delta)$ -private

- ▶ Standard composition:  $(n \cdot \epsilon, n \cdot \delta)$ -private
- ▶ Advanced composition:  $(\epsilon^*, \delta^*)$ -private

$$\epsilon^* \approx \sqrt{n} \cdot \epsilon \quad \text{and} \quad \delta^* \approx n \cdot \delta + \delta'$$

# Advanced composition theorem

Compose  $n$  mechanisms, each  $(\epsilon, \delta)$ -private

- ▶ Standard composition:  $(n \cdot \epsilon, n \cdot \delta)$ -private
- ▶ Advanced composition:  $(\epsilon^*, \delta^*)$ -private

$$\epsilon^* \approx \sqrt{n} \cdot \epsilon \quad \text{and} \quad \delta^* \approx n \cdot \delta + \delta'$$

Trade off  $\epsilon$  and  $\delta$

# Advanced composition theorem

Compose  $n$  mechanisms, each  $(\epsilon, \delta)$ -private

- ▶ Standard composition:  $(n \cdot \epsilon, n \cdot \delta)$ -private
- ▶ Advanced composition:  $(\epsilon^*, \delta^*)$ -private

$$\epsilon^* \approx \sqrt{n} \cdot \epsilon \quad \text{and} \quad \delta^* \approx n \cdot \delta + \delta'$$

Trade off  $\epsilon$  and  $\delta$

# Advanced composition theorem

Compose  $n$  mechanisms, each  $(\epsilon, \delta)$ -private

- ▶ Standard composition:  $(n \cdot \epsilon, n \cdot \delta)$ -private
- ▶ Advanced composition:  $(\epsilon^*, \delta^*)$ -private

$$\epsilon^* \approx \sqrt{n} \cdot \epsilon \quad \text{and} \quad \delta^* \approx n \cdot \delta + \delta'$$

Trade off  $\epsilon$  and  $\delta$

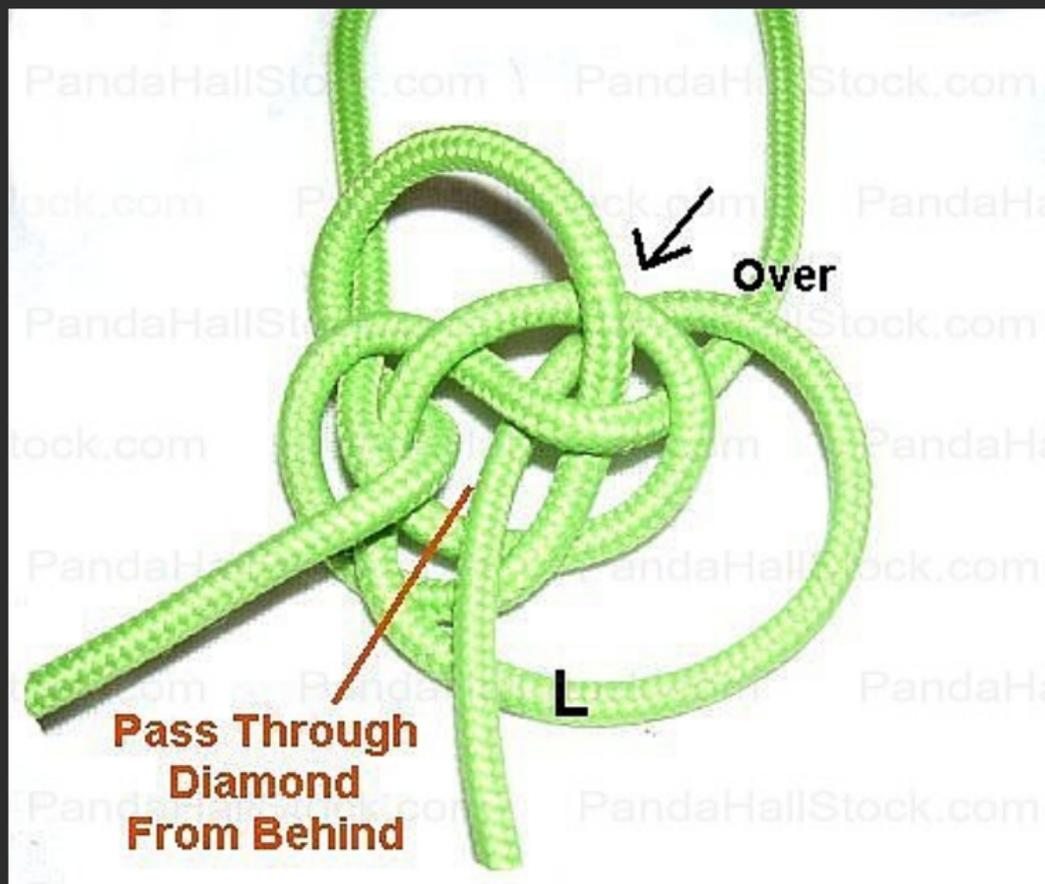
## In pRHL: new **while** rule

$$\text{AC} \frac{\models \Theta \rightarrow e\langle 1 \rangle = e\langle 2 \rangle \quad \vdash \{\Theta \wedge e\langle 1 \rangle\} \quad c_1 \sim_{(\epsilon, \delta)} c_2 \quad \{\Theta\} \quad \text{while } e_1 \text{ do } c_1 \text{ executes at most } n \text{ iterations}}{\vdash \{\Theta\} \quad \text{while } e_1 \text{ do } c_1 \sim_{(\epsilon^*, \delta^*)} \text{while } e_2 \text{ do } c_2 \quad \{\Theta \wedge \neg e\langle 1 \rangle\}}$$

### Notes

- ▶ Surprising: generalization to approximate couplings
- ▶ More surprising: privacy composition directly generalizes

## Putting it all together



# A brief preview: the Between Thresholds algorithm

Variant of a mechanism by Bun, Steinke, Ullman (2016)

```
ASVbt(a, b, M, N, d) :=  
i ← 0; l ← [];  
u  $\xleftarrow{\$}$   $\mathcal{L}_{\epsilon/2}(0)$ ;  
A ← a - u; B ← b + u;  
while i < N ∧ |l| < M do  
  i' ← i; hd ← -1;  
  while i' < N do  
    if (hd = -1)  
      q ←  $\mathcal{A}(l)$ ;  
      S  $\xleftarrow{\$}$   $\mathcal{L}_{\epsilon'/3}(\text{evalQ}(q, d))$ ;  
      if (A ≤ S ≤ B) then hd ← i;  
      i ← i + 1;  
    i' ← i' + 1;  
  if (hd ≠ -1) then l ← hd :: l;  
return l
```

Formalized  $(\epsilon, \delta)$ -privacy in EasyCrypt

## Formal proof combines many different features:

- ▶ Accuracy-dependent privacy
- ▶ Advanced composition
- ▶ Adaptively chosen inputs
- ▶ “Subset” coupling

## Formal proof combines many different features:

- ▶ Accuracy-dependent privacy
- ▶ Advanced composition
- ▶ Adaptively chosen inputs
- ▶ “Subset” coupling

Formal proof combines many different features:

- ▶ Accuracy-dependent privacy
- ▶ Advanced composition
- ▶ Adaptively chosen inputs
- ▶ “Subset” coupling

Please see the paper!

Our work: formal privacy proofs with:

Accuracy-dependent privacy

Advanced composition

Adaptive inputs

Our work: formal privacy proofs with:

Accuracy-dependent privacy

Advanced composition

Adaptive inputs

