The Complexity of Verifying Loop-Free Programs as Differentially Private

Marco Gaboardi
Boston University, MA, USA

Kobbi Nissim
Georgetown University, Washington, DC, USA

David Purser
University of Warwick, Coventry, UK
Max Planck Institute for Software Systems, Saarbrücken, Germany

Abstract

We study the problem of verifying differential privacy for loop-free programs with probabilistic choice. Programs in this class can be seen as randomized Boolean circuits, which we will use as a formal model to answer two different questions: first, deciding whether a program satisfies a prescribed level of privacy; second, approximating the privacy parameters a program realizes. We show that the problem of deciding whether a program satisfies ε-differential privacy is $\mathbf{coNP}^{\#\mathbf{P}}$-complete. In fact, this is the case when either the input domain or the output range of the program is large. Further, we show that deciding whether a program is (ε, δ)-differentially private is $\mathbf{coNP}^{\#\mathbf{P}}$-hard, and in $\mathbf{coNP}^{\#\mathbf{P}}$ for small output domains, but always in $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$. Finally, we show that the problem of approximating the level of differential privacy is both $\mathbf{NP}$-hard and $\mathbf{coNP}$-hard. These results complement previous results by Murtagh and Vadhan [35] showing that deciding the optimal composition of differentially private components is $\#\mathbf{P}$-complete, and that approximating the optimal composition of differentially private components is in $\mathbf{P}$.

2012 ACM Subject Classification Security and privacy; Theory of computation → Probabilistic computation

Keywords and phrases differential privacy, program verification, probabilistic programs

Digital Object Identifier 10.4230/LIPIcs.ICALP.2020.129

Category Track B: Automata, Logic, Semantics, and Theory of Programming

Related Version A full version of the paper is available at https://arxiv.org/abs/1911.03272.

Funding M. G. and K. N. were supported by NSF grant No. 1565387 TWC: Large: Collaborative: Computing Over Distributed Sensitive Data. D. P. was supported by the UK EPSRC Centre for Doctoral Training in Urban Science (EP/L016400/1).

Acknowledgements Research partially done while M.G. and K.N. participated in the "Data Privacy: Foundations and Applications" program held at the Simons Institute, UC Berkeley in spring 2019.

1 Introduction

Differential privacy [22] is currently making significant strides towards being used in large scale applications. Prominent real-world examples include the use of differentially private computations by the US Census' OnTheMap project[1], applications by companies such as Google and Apple [24, 36, 4, 18], and the US Census' plan to deploy differentially private releases in the upcoming 2020 Decennial [1].

[1] https://onthemap.ces.census.gov

47th International Colloquium on Automata, Languages, and Programming (ICALP 2020).
Editors: Artur Czumaj, Anuj Dawar, and Emanuela Merelli; Article No. 129; pp. 129:1–129:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

More often than not, algorithms and their implementations are analyzed “on paper” to show that they provide differential privacy. This analysis – a proof that the outcome distribution of the algorithm is stable under the change in any single individual’s information – is often intricate and may contain errors (see [32] for an illuminating discussion about several wrong versions of the sparse vector algorithm which appeared in the literature). Moreover, even if it is actually differentially private, an algorithm may be incorrectly implemented when used in practice, e.g., due to coding errors, or because the analysis makes assumptions which do not hold in finite computers, such as the ability to sample from continuous distributions (see [34] for a discussion about privacy attacks on naive implementations of continuous distributions). Verification tools may help validate, given the code of an implementation, that it would indeed provide the privacy guarantees it is intended to provide. However, despite the many verification efforts that have targeted differential privacy based on automated or interactive techniques (see, e.g.,[37, 9, 40, 25, 7, 44, 6, 2, 14, 15]), little is known about the complexity of some of the basic problems in this area. Our aim is to clarify the complexity of some of these problems.

In this paper, we consider the computational complexity of determining whether programs satisfy (ε, δ)-differential privacy. The problem is generally undecidable, and we hence restrict our attention to probabilistic loop-free programs, which are part of any reasonable programming language supporting random computations. To approach this question formally, we consider probabilistic circuits. The latter are Boolean circuits with input nodes corresponding both to input bits and to uniformly random bits (“coin flips”) where the latter allow the circuit to behave probabilistically (see Figure 1). We consider both decision and approximation versions of the problem, where in the case of decision the input consists of a randomized circuit and parameters ε, δ and in the case of approximation the input is a randomized circuit, the desired approximation precision, and one of the two parameters ε, δ. In both cases, complexity is measured as function of the total input length in bits (circuit and parameters).

Previous works have studied the complexity of composing differentially private components. For any k differentially private algorithms with privacy parameters $(\varepsilon_1, \delta_1), \ldots, (\varepsilon_k, \delta_k)$, it is known that their composition is also differentially private [22, 23, 35], making composition a powerful design tool for differentially private programs. However, not all interesting differentially private programs are obtained by composing differentially private components, and a goal of our work is to understand what is the complexity of verifying that full programs are differentially private, and how this complexity differs from the one for programs which result of composing differentially private components.

Regarding the resulting parameters, the result of composing the k differentially private algorithms above results in $(\varepsilon_g, \delta_g)$-differentially private for a multitude of possible $(\varepsilon_g, \delta_g)$ pairs. Murtagh and Vadhan showed that determining the minimal ε_g given δ_g is #**P**-complete [35]. They also gave a polynomial time approximation algorithm that computes ε_g to arbitrary accuracy, giving hope that for “simple” programs deciding differential privacy or approximating of privacy parameters may be tractable. Unfortunately, our results show that this is not the case.

1.1 Contributions

Following the literature, we refer to the variant of differential privacy where $\delta = 0$ as *pure* differential privacy and to the variant where $\delta > 0$ as *approximate* differential privacy. We contribute in three directions.

- **Bounding pure differential privacy.** We show that determining whether a randomized circuit is ε-differentially private is $\mathbf{coNP}^{\#\mathbf{P}}$-complete.[2] To show hardness in $\mathbf{coNP}^{\#\mathbf{P}}$ we consider a complement to the problem E-MAJ-SAT [31], which is complete for $\mathbf{NP}^{\#\mathbf{P}}$ [13]. In the complementary problem, ALL-MIN-SAT, given a formula ϕ over $n+m$ variables the task is to determine if for all allocations $\boldsymbol{x} \in \{0,1\}^n$, $\phi(\boldsymbol{x},\boldsymbol{y})$ evaluates to true on no more than $\frac{1}{2}$ of allocations to $\boldsymbol{y} \in \{0,1\}^m$.
- **Bounding approximate differential privacy.** Turning to the case where $\delta > 0$, we show that determining whether a randomized circuit is (ε,δ)-differentially private is $\mathbf{coNP}^{\#\mathbf{P}}$-complete when the number of output bits is small relative to the total size of the circuit and otherwise between $\mathbf{coNP}^{\#\mathbf{P}}$ and $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$.
- **Approximating the parameters ε and δ.** Efficient approximation algorithms exist for optimal composition [35], and one might expect the existence of polynomial time algorithms to approximate ε or δ in randomized circuits. We show this is $\mathbf{NP}$-hard and $\mathbf{coNP}$-hard, and therefore an efficient algorithm does not exist (unless $\mathbf{P} = \mathbf{NP}$).

Our results show that for loop-free programs with probabilistic choice directly verifying whether a program is differentially private is intractable. These results apply to programs in any reasonable programming language supporting randomized computations. Hence, they set the limits on where to search for automated techniques for these tasks.

The relation to quantitative information flow

Differential privacy shares similarities with quantitative information flow [17, 27], which is an entropy-based theory measuring how secure a program is. Alvim et al. [3] showed that a bound on pure differential privacy implies a bound on quantitative information flow. So, one could hope that bounding differential privacy could be easier than bounding quantitative information flow. Yasuoka and Terauchi [42] have shown that bounding quantitative information flow for loop free boolean programs with probabilistic choice is $\mathbf{PP}$-hard (but in $\mathbf{PSPACE}$). In contrast, our results show that bounding pure differential privacy is $\mathbf{coNP}^{\#\mathbf{P}}$-complete. Chadha et al. [11] showed the problem to be $\mathbf{PSPACE}$-complete for boolean programs with loops and probabilistic choice (notice that this would be not true for programs with integers). We leave the analogous question for future works.

2 Preliminaries

Numbers

By a *number given as a rational* we mean a number of the form $\frac{x}{y}$ where x, y are given as binary integers.

2.1 Loop-free probabilistic programs

We consider a simple loop-free imperative programming language built over Booleans, and including probabilistic choice.

$$
\begin{array}{lcll}
x & ::= & [\mathrm{a{-}z}]^{+} & \text{(variable identifiers)} \\
b & ::= & \texttt{true} \mid \texttt{false} \mid \texttt{random} \mid x \mid b \wedge b \mid b \vee b \mid \neg b & \text{(boolean expressions)} \\
c & ::= & \texttt{SKIP} \mid x := b \mid c;\ c \mid \texttt{if}\ b\ \texttt{then}\ c\ \texttt{else}\ c & \text{(commands)} \\
t & ::= & x \mid t, x & \text{(list of variables)} \\
p & ::= & \texttt{input}(t);\ c;\ \texttt{return}(t) & \text{(programs)}
\end{array}
$$

[2] The class $\mathbf{coNP}^{\#\mathbf{P}}$ is contained in $\mathbf{PSPACE}$ and contains the polynomial hierarchy (as, per Toda's Theorem, $\mathbf{PH} \subseteq \mathbf{P}^{\#\mathbf{P}}$).

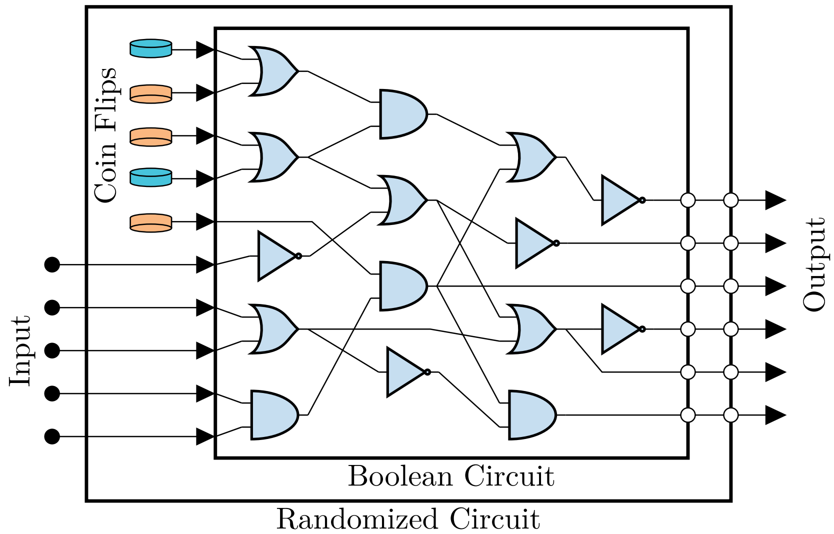

Figure 1 Example randomized circuit.

Probabilistic programs [30] extend standard programs with the addition of coin tosses; this is achieved by the probabilistic operation `random`, which returns either `true` or `false` with equal probability. A standard operation, sometimes denoted by $c \oplus c$, which computes one of the two expressions with probability $\frac{1}{2}$ each is achieved with `if random then` c `else` c. The notation $c \oplus c$ is avoided as $\oplus$ refers to *exclusive or* in this paper.

The semantics of the programming language are standard and straight forward. Without loss of generality, each variable assignment is final, that is, each assignment must go to a fresh variable. Looping behaviour is not permitted, although bounded looping can be encoded by unrolling the loop.

▶ Remark 1. Our results also hold when the language additionally supports integers and the associated operations (e.g. $+, \times, -, \geq, =$ etc.), providing the integers are of a bounded size. Such a language is equally expressive as the language presented here. Further details are given in the full version of the paper.

2.2 Probabilistic circuits

▶ **Definition 2.** *A Boolean circuit ψ with n inputs and ℓ outputs is a directed acyclic graph $\psi = (V, E)$ containing n input vertices with zero in-degree, labeled $X_1, \dots, X_n$ and ℓ output vertices with zero out-degree, labeled $O_1, \dots, O_\ell$. Other nodes are assigned a label in $\{\wedge, \vee, \neg\}$, with vertices labeled $\neg$ having in-degree one and all others having in-degree two. The size of ψ, denoted $|\psi|$, is defined to be $|V|$. A* randomized circuit *has m additional random input vertices labeled $R_1, \dots, R_m$.*

Given an input string $\boldsymbol{x} = (x_1, \dots, x_n) \in \{0,1\}^n$, the circuit is evaluated as follows. First, the values $x_1, \dots, x_n$ are assigned to the nodes labeled $X_1, \dots, X_n$. Then, m bits $\boldsymbol{r} = (r_1, \dots, r_m)$ are sampled uniformly at random from $\{0,1\}^m$ and assigned to the nodes labeled $R_1, \dots, R_m$. Then, the circuit is evaluated in topological order in the natural way. E.g., let v be a node labeled $\wedge$ with incoming edges $(u_1, v), (u_2, v)$ where u_1, u_2 were assigned values z_1, z_2 then v is assigned the value $z_1 \wedge z_2$. The outcome of ψ is $(o_1, \dots, o_\ell)$, the concatenation of values assigned to the ℓ output vertices $O_1, \dots, O_\ell$.

For input $\boldsymbol{x} \in \{0,1\}^n$ *and event* $E \subseteq \{0,1\}^\ell$ *we have*

$$\Pr[\psi(\boldsymbol{x}) \in E] = \frac{|\{\boldsymbol{r} \in \{0,1\}^m \ : \ \psi(\boldsymbol{x}, \boldsymbol{r}) \in E\}|}{2^m}.$$

▶ Remark 3. The operators, $\land, \lor$ and $\neg$ are functionally complete. However, we will also use $\oplus$ (exclusive or), such that $p \oplus q \iff (p \lor q) \land \neg(p \land q)$.

2.3 Equivalence of programs and circuits

▶ **Lemma 4.** *A loop-free probabilistic program can be converted into an equivalent probabilistic boolean circuit in linear time in the size of the program (and vice-versa).*

Proof sketch. It is clear that a probabilistic circuit can be expressed as a probabilistic program using just boolean operations by expressing a variable for each vertex after sorting the vertices in topological order.

To convert a probabilistic Boolean program into a probabilistic circuit, each of the commands can be handled using a fixed size sub-circuit, each of which can be composed together appropriately. ◀

Given the equivalence between loop-free probabilistic programs and probabilistic circuits, the remainder of the paper will use probabilistic circuits.

2.4 Differential privacy in probabilistic circuits

Let X be any input domain. An input to a differentially private analysis would generally be an array of elements from a data domain X, each corresponding to the information of an individual, i.e., $\boldsymbol{x} = (x_1, \dots, x_n) \in X^n$.

The definition of differential privacy depends on adjacency between inputs, we define *neighboring* inputs.

▶ **Definition 5.** *Inputs* $\boldsymbol{x} = (x_1, \dots, x_n)$ *and* $\boldsymbol{x}' = (x'_1, \dots, x'_n) \in X^n$ *are called* neighboring *if there exist* $i \in [n]$ *s.t. if* $j \neq i$ *then* $x_j = x'_j$.

In this work, we will consider input domains with finite representation. Without loss of generality we set $X = \{0,1\}^k$ and hence an array $x = (x_1, \dots, x_n)$ can be written as a sequence of nk bits, and given as input to a (randomized) circuit with nk inputs. Our lower bounds work already for for $k = 1$ and our upper bounds are presented using $k = 1$ but generalise to all k.

▶ **Definition 6** (Differential Privacy [22, 21])**.** *A probabilistic circuit* ψ *is* (ε, δ)*-differentially private if for all neighboring* $\boldsymbol{x}, \boldsymbol{x}' \in X^n$ *and for all* $E \subseteq \{0,1\}^\ell$*,*

$$\Pr[\psi(\boldsymbol{x}) \in E] \le e^\varepsilon \cdot \Pr[\psi(\boldsymbol{x}') \in E] + \delta.$$

Following common use, we refer to the case where $\delta = 0$ as *pure* differential privacy and to the case where $\delta > 0$ as *approximate* differential privacy. When omitted, δ is understood to be zero.

2.5 Problems of deciding and approximating differential privacy

We formally define our three problems of interest.

▶ **Definition 7.** *The problem* DECIDE-ε-DP *asks, given ε and ψ, if ψ is ε-differentially private. We assume ε is given by the input e^{ε} as a rational number.*

▶ **Definition 8.** *The problem* DECIDE-ε,δ-DP *asks, given ε, δ and ψ, if ψ is (ε,δ)-differentially private. We assume ε is given by the input e^{ε} as a rational number.*

▶ **Definition 9.** *Given an approximation error $\gamma > 0$, the* APPROXIMATE-δ *problem and the* APPROXIMATE-ε *problem, respectively, ask:*

- *Given ε, find $\hat{\delta} \in [0,1]$, such that $0 \leq \hat{\delta} - \delta \leq \gamma$, where δ is the minimal value such that ψ is (ε,δ)-differentially private.*
- *Given δ, find $\hat{\varepsilon} \geq 0$, such that $0 \leq \hat{\varepsilon} - \varepsilon \leq \gamma$, where ε is the minimal value such that ψ is (ε,δ)-differentially private.*

2.6 The class $\mathbf{coNP}^{\#\mathbf{P}}$

The complexity class $\#\mathbf{P}$ is the counting analogue of $\mathbf{NP}$ problems. In particular #SAT, the problem of counting the number of satisfying assignments of a given a boolean formula ϕ on n variables, is complete for $\#\mathbf{P}$. Similarly #CIRCUITSAT, the problem of counting the satisfying assignments of a circuit with a single output, is complete for $\#\mathbf{P}$.

A language L is in $\mathbf{coNP}^{\#\mathbf{P}}$ if membership in L can be refuted using a polynomial time non-deterministic Turing machine with access to a $\#\mathbf{P}$ oracle. It is easy to see that $\mathbf{coNP}^{\#\mathbf{P}} = \mathbf{coNP}^{\mathbf{PP}}$, and $\mathbf{PH} \subseteq \mathbf{coNP}^{\#\mathbf{P}} \subseteq \mathbf{PSPACE}$, where $\mathbf{PH} \subseteq \mathbf{coNP}^{\#\mathbf{P}}$ follows by Toda's theorem ($\mathbf{PH} \subseteq \mathbf{P}^{\#\mathbf{P}}$) [39].

The following decision problem is complete for $\mathbf{NP}^{\#\mathbf{P}}$ [13]:

▶ **Definition 10.** *E-MAJ-SAT asks, given ϕ a quantifier free formula over $n+m$ variables if there exist an allocation $\boldsymbol{x} \in \{0,1\}^n$ such that there are strictly greater than $\frac{1}{2}$ of allocations to $\boldsymbol{y} \in \{0,1\}^m$ where $\phi(\boldsymbol{x},\boldsymbol{y})$ evaluates to true.*

The complementary problem ALL-MIN-SAT, is complete for $\mathbf{coNP}^{\#\mathbf{P}}$: a formula ϕ is ALL-MIN-SAT, if ϕ is not E-MAJ-SAT. That is, ϕ a quantifier free formula over $n+m$ variables is ALL-MIN-SAT if for all allocations $\boldsymbol{x} \in \{0,1\}^n$ there are no more than $\frac{1}{2}$ of allocations to $\boldsymbol{y} \in \{0,1\}^m$ where $\phi(\boldsymbol{x},\boldsymbol{y})$ evaluates to true.

3 The complexity of deciding pure differential privacy

In this section we classify the complexity of deciding ε-differential privacy, for which we show the following theorem:

▶ **Theorem 11.** DECIDE-ε-DP *is* $\mathbf{coNP}^{\#\mathbf{P}}$*-complete.*

It will be convenient to consider the well-known simpler reformulation of the definition of pure differential privacy in finite ranges to consider specific outcomes $\boldsymbol{o} \in \{0,1\}^{\ell}$ rather than events $E \subseteq \{0,1\}^{\ell}$.

▶ **Reformulation 12** (Pure differential privacy)**.** A probabilistic circuit ψ is ε-differentially private if and only if for all neighboring $\boldsymbol{x},\boldsymbol{x'} \in X^n$ and for all $\boldsymbol{o} \in \{0,1\}^{\ell}$,

$$\Pr[\psi(\boldsymbol{x}) = \boldsymbol{o}] \leq e^{\varepsilon} \cdot \Pr[\psi(\boldsymbol{x'}) = \boldsymbol{o}].$$

3.1 Decide-ε-DP is in $\mathbf{coNP}^{\#\mathbf{P}}$

We show a non-deterministic Turing machine which can "refute" ψ being ε-differentially private in (non-deterministic) polynomial time with a $\#\mathbf{P}$ oracle. A circuit ψ is shown not to be ε-differentially private by exhibiting a combination $\boldsymbol{x}, \boldsymbol{x'}, \boldsymbol{o}$ such that $\Pr[\psi(\boldsymbol{x}) = \boldsymbol{o}] > e^\varepsilon \cdot \Pr[\psi(\boldsymbol{x'}) = \boldsymbol{o}]$. The witness to the non-deterministic Turing machine would be a sequence of $2n$ bits parsed as neighboring inputs $\boldsymbol{x}, \boldsymbol{x'} \in \{0,1\}^n$ and ℓ bits describing an output $\boldsymbol{o} \in \{0,1\}^\ell$. The constraint can then be checked in polynomial time, using the $\#\mathbf{P}$ oracle to compute $\Pr[\psi(\boldsymbol{x}) = \boldsymbol{o}]$ and $\Pr[\psi(\boldsymbol{x'}) = \boldsymbol{o}]$.

To compute $\Pr[\psi(\boldsymbol{x}) = \boldsymbol{o}]$ in $\#\mathbf{P}$ we create an instance to #CircuitSat, which will count the number of allocations to the m probabilistic bits consistent with this output. We do this by extending ψ with additional gates reducing to a single output which is true only when the input is fixed to $\boldsymbol{x}$ and the output of ψ was $\boldsymbol{o}$.

3.2 $\mathbf{coNP}^{\#\mathbf{P}}$-hardness of Decide-$\varepsilon$-DP

To show $\mathbf{coNP}^{\#\mathbf{P}}$-hardness of Decide-$\varepsilon$-DP we show a reduction from All-Min-Sat in Lemma 14; together with the inclusion result above, this entails that Decide-ε-DP is $\mathbf{coNP}^{\#\mathbf{P}}$-complete (Theorem 11).

Randomized response [41] is a technique for answering sensitive Yes/No questions by flipping the answer with probability $p \leq 0.5$. Setting $p = \frac{1}{1+e^\varepsilon}$ gives ε-differential privacy. Thus $p = 0$ gives no privacy and $p = 0.5$ gives total privacy (albeit no utility).

▶ **Definition 13** (Randomized Response).

$$RR_\varepsilon(x) = \begin{cases} x & w.p. \quad \frac{e^\varepsilon}{1+e^\varepsilon} \\ \neg x & w.p. \quad \frac{1}{1+e^\varepsilon} \end{cases}$$

▶ **Lemma 14.** All-Min-Sat *reduces in polynomial time to* Decide-ε-DP.

Proof. We will reduce from All-Min-Sat to Decide-ε-DP using randomized response. We will take a boolean formula ϕ and create a probabilistic circuit that is ε-differentially private if and only if ϕ is All-Min-Sat.

Consider the circuit ψ which takes as input the value $z \in \{0,1\}$. It probabilistically chooses a value of $\boldsymbol{x} \in \{0,1\}^n$ and $\boldsymbol{y} \in \{0,1\}^m$ and one further random bit p_1 and computes $b = z \oplus \neg(p_1 \vee \phi(\boldsymbol{x}, \boldsymbol{y}))$. The circuit outputs $(\boldsymbol{x}, b)$.

▷ Claim 15. ψ is $\ln(3)$-differentially private if and only if ϕ is All-Min-Sat.

Suppose $\phi \in$ All-Min-Sat then, no matter the choice of $\boldsymbol{x}$,

$$0 \leq \Pr_y[\phi(\boldsymbol{x}, \boldsymbol{y}) = 1] \leq \frac{1}{2},$$

and hence

$$\frac{1}{4} \leq \Pr_{y,p_1}[\neg(p_1 \vee \phi(\boldsymbol{x}, \boldsymbol{y})) = 1] \leq \frac{1}{2}.$$

We conclude the true answer z is flipped between $\frac{1}{4}$ and $\frac{1}{2}$ of the time, observe this is exactly the region in which randomized response gives us the most privacy. In the worst case $p = \frac{1}{4} = \frac{1}{1+e^\varepsilon}$, gives $e^\varepsilon = 3$, so $\ln(3)$-differential privacy.

In the converse, suppose $\phi \in$ E-Maj-Sat, then for some $\boldsymbol{x}$

$$\frac{1}{2} < \Pr_{y}[\phi(\boldsymbol{x}, \boldsymbol{y}) = 1] \leq 1,$$

and then

$$\Pr_{y,p_1}[\neg(p_1 \vee \phi(\boldsymbol{x}, \boldsymbol{y})) = 1] < \frac{1}{4},$$

in which case the randomized response does not provide $\ln(3)$-differential privacy. ◀

▶ Remark 16. We skew the result so that in the positive case (when $\phi \in$ All-Min-Sat) the proportion of accepting allocations is between $\frac{1}{4}$ and $\frac{1}{2}$, resulting in the choice of $\ln(3)$-differentially privacy. Alternative skews, using more bits akin to p_1, shows hardness for other choices of ε.

Hardness by circuit shape

In our proof of the upper-bound we use **coNP** to resolve the non-deterministic choice of both input and output. We show this is necessary in the sense **coNP** is still required for either large input or large output. The hardness proof used in Lemma 14 shows that when $|\psi| = n$ the problem is hard for $\Omega(1)$-bit input and $\Omega(n)$-bit output.

We can also prove this is hard for $\Omega(n)$-bit input and $\Omega(1)$-bit output. Intuitively a counter example to differential privacy has two choices: a pair of adjacent input and a given output upon which the relevant inequality will hold. So to “refute” All-Min-Sat the counterexample of the All choice (i.e. $\boldsymbol{x}$) can be selected in the input, rather than the output as in our case. Since the input is now non-trivial we must take care of what happens when the adjacent bit is in the choice of $\boldsymbol{x}$. Details are given in the full version.

Further the problem is in $\mathbf{P}^{\#\mathbf{P}}$ for $O(\log(n))$-bit input and $O(\log(n))$-bit output, as in this case, the choices made by **coNP** can instead be checked deterministically in polynomial time. In this case we show **PP**-hardness, which applies even when there is 1-bit input and 1-bit output.

4 On the complexity of deciding approximate differential privacy

It is less clear whether deciding (ε, δ)-differential privacy can be done in $\mathbf{coNP}^{\#\mathbf{P}}$. First we consider restrictions to the shape of the circuit so that $\mathbf{coNP}^{\#\mathbf{P}}$ can be recovered, and then show that in general the problem is in $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$.

Recall that in the case of ε-differential privacy it was enough to consider singleton events $\{\boldsymbol{o}\}$ where $\boldsymbol{o} \in \{0,1\}^\ell$, however in the definition of (ε, δ)-differential privacy we must quantify over output events $E \subseteq \{0,1\}^\ell$. If we consider circuits with one output bit ($\ell = 1$), then the event space essentially reduces to $E \in \{\emptyset, \{0\}, \{1\}, \{0,1\}\}$ and we can apply the same technique.

We expand this to the case when the number of outputs bits is logarithmic $\ell \leq \log(|\psi|)$. To cater to this, rather than guessing a violating $E \in \{0,1\}^\ell$, we consider a violating subset of events $E \subseteq \{0,1\}^\ell$. Given such an event E we create a circuit ψ_E on ℓ inputs and a single output which indicates whether the input is in the event E. The size of this circuit is exponential in ℓ, thus polynomial in $|\psi|$. Composing $\psi_E \circ \psi$, we check the conditions hold for this event E, with just one bit of output.

▷ Claim 17. Decide-ε, δ-DP, restricted to circuits ψ with ℓ bit outputs where $\ell \leq \log(|\psi|)$, is in $\mathbf{coNP}^{\#\mathbf{P}}$ (and hence $\mathbf{coNP}^{\#\mathbf{P}}$-complete).

The claim trivially extends to $\ell \leq c \cdot \log(|\psi|)$ for any *fixed* $c > 0$.

4.1 Decide-$\varepsilon, \boldsymbol{\delta}$-DP is in $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$

We now show that Decide-ε, δ-DP in the most general case can be solved in $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$. We will assume $e^\varepsilon = \alpha$ is given as a rational, with $\alpha = \frac{u}{v}$ for some integers u and v. Recall we use n, ℓ and m to refer to the number of input, output and random bits of a circuit respectively. While we will use non-determinism to choose inputs leading to a violating event, unlike in Section 3 it would not be used for finding a violating event E, as an (explicit) description of such an event may be of super-polynomial length. It would be useful for us to use a reformulation of approximate differential privacy, using a sum over potential individual outcomes.

▶ **Reformulation 18** (Pointwise differential privacy [7]). A probabilistic circuit ψ is (ε, δ)-differentially private if and only if for all neighboring $\boldsymbol{x}, \boldsymbol{x}' \in X^n$ and for all $\boldsymbol{o} \in \{0,1\}^\ell$,

$$\sum_{\boldsymbol{o} \in \{0,1\}^\ell} \delta_{\boldsymbol{x},\boldsymbol{x}'}(\boldsymbol{o}) \le \delta,$$

where $\delta_{\boldsymbol{x},\boldsymbol{x}'}(\boldsymbol{o}) = \max\left(\Pr[\psi(\boldsymbol{x}) = \boldsymbol{o}] - e^\varepsilon \cdot \Pr[\psi(\boldsymbol{x}') = \boldsymbol{o}], 0\right).$

We define $\mathcal{M}$, a non-deterministic Turing Machine with access to a $\#\mathbf{P}$-oracle, and where each execution branch runs in polynomial time. On inputs a probabilistic circuit ψ and neighboring $\boldsymbol{x}, \boldsymbol{x}' \in X^n$ the number of accepting executions of $\mathcal{M}$ would be proportional to $\sum_{\boldsymbol{o} \in \{0,1\}^\ell} \delta_{\boldsymbol{x},\boldsymbol{x}'}(\boldsymbol{o})$.

In more detail, on inputs ψ, $\boldsymbol{x}$ and $\boldsymbol{x}'$, $\mathcal{M}$ chooses $\boldsymbol{o} \in \{0,1\}^\ell$ and an integer $C \in \{1, 2, \dots, 2^{m+\lceil \log(v) \rceil}\}$ (this requires choosing $\ell + m + \lceil \log(v) \rceil$ bits). Through a call to the $\#\mathbf{P}$ oracle, $\mathcal{M}$ computes

$$a = |\{\boldsymbol{r} \in \{0,1\}^m : \psi(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{o}\}|$$

and

$$b = |\{\boldsymbol{r} \in \{0,1\}^m : \psi(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{o}\}|\,.$$

Finally, $\mathcal{M}$ accepts if $v \cdot a - u \cdot b \ge C$ and otherwise rejects.

▶ **Lemma 19.** *Given two inputs $\boldsymbol{x}, \boldsymbol{x}' \in X^n$, $\mathcal{M}(\psi, \boldsymbol{x}, \boldsymbol{x}')$ has exactly $v \cdot 2^m \sum_{\boldsymbol{o} \in \{0,1\}^\ell} \delta_{\boldsymbol{x},\boldsymbol{x}'}(\boldsymbol{o})$ accepting executions.*

Proof. Let $\mathbb{1}\{X\}$ be the indicator function, which is one if the predicate X holds and zero otherwise.

$$\begin{aligned}
v \cdot 2^m \sum_{\boldsymbol{o} \in \{0,1\}^\ell} \delta_{\boldsymbol{x},\boldsymbol{x}'}(\boldsymbol{o}) &= \sum_{\boldsymbol{o} \in \{0,1\}^\ell} v \cdot 2^m \max\left(\Pr[\psi(\boldsymbol{x}) = \boldsymbol{o}] - \alpha \Pr[\psi(\boldsymbol{x}') = \boldsymbol{o}], 0\right) \\
&= \sum_{\boldsymbol{o} \in \{0,1\}^\ell} v 2^m \max\left(\frac{1}{2^m} \sum_{\boldsymbol{r} \in \{0,1\}^m} \mathbb{1}\{\psi(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{o}\} - \alpha \frac{1}{2^m} \sum_{\boldsymbol{r} \in \{0,1\}^m} \mathbb{1}\left\{\psi(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{o}\right\}, 0\right) \\
&= \sum_{\boldsymbol{o} \in \{0,1\}^\ell} \max\left(v \sum_{\boldsymbol{r} \in \{0,1\}^m} \mathbb{1}\{\psi(\boldsymbol{x}, \boldsymbol{r}) = \boldsymbol{o}\} - v\alpha \sum_{\boldsymbol{r} \in \{0,1\}^m} \mathbb{1}\left\{\psi(\boldsymbol{x}', \boldsymbol{r}) = \boldsymbol{o}\right\}, 0\right)
\end{aligned}$$

$$\begin{aligned}
\dots &= \sum_{\boldsymbol{o}\in\{0,1\}^\ell} \max\left(v \sum_{\boldsymbol{r}\in\{0,1\}^m} \mathbb{1}\{\psi(\boldsymbol{x},\boldsymbol{r})=\boldsymbol{o}\} - u \sum_{\boldsymbol{r}\in\{0,1\}^m} \mathbb{1}\big\{\psi(\boldsymbol{x}',\boldsymbol{r})=\boldsymbol{o}\big\}, 0\right)\\
&= \sum_{\boldsymbol{o}\in\{0,1\}^\ell} \sum_{C=1}^{2^{\lceil\log(v)\rceil+m}} \mathbb{1}\left\{ \max\left(v \sum_{\boldsymbol{r}\in\{0,1\}^m} \mathbb{1}\{\psi(\boldsymbol{x},\boldsymbol{r})=\boldsymbol{o}\} - u \sum_{\boldsymbol{r}\in\{0,1\}^m} \mathbb{1}\big\{\psi(\boldsymbol{x}',\boldsymbol{r})=\boldsymbol{o}\big\}, 0\right) \geq C \right\}\\
&= \text{number of accepting executions in } \widehat{\mathcal{M}}
\end{aligned}$$ ◀

We can now describe our $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$ procedure for DECIDE-ε,δ-DP. The procedure takes as input a probabilistic circuit ψ.

1. Non-deterministically choose neighboring $\boldsymbol{x}$ and $\boldsymbol{x}' \in \{0,1\}^n$ (i.e., $2n$ bits).
2. Let $\mathcal{M}$ be the non-deterministic Turing Machine with access to a $\#\mathbf{P}$-oracle as described above. Create a machine $\widehat{\mathcal{M}}$ with no input that executes $\mathcal{M}$ on $\psi, \boldsymbol{x}, \boldsymbol{x}'$.
3. Make an $\#\mathbf{P}^{\#\mathbf{P}}$ oracle call for the number of accepting executions $\widehat{\mathcal{M}}$ has.
4. Reject if the number of accepting executions is greater than $v \cdot 2^m \cdot \delta$ and otherwise accept.

By Lemma 19, there is a choice $\boldsymbol{x}, \boldsymbol{x}'$ on which the procedure rejects if and only if ψ is not (ε,δ)-differentially private.

4.2 Hardness

Theorem 11 shows that DECIDE-ε-DP is $\mathbf{coNP}^{\#\mathbf{P}}$-complete, in particular $\mathbf{coNP}^{\#\mathbf{P}}$-hard and since DECIDE-$\varepsilon$-DP is a special case of DECIDE-ε,δ-DP, this is also $\mathbf{coNP}^{\#\mathbf{P}}$-hard. Nevertheless the proof is based on particular values of ε and in the full version we provide an alternative proof of hardness based on δ. This proof result will apply for any ε (even for $\varepsilon = 0$) and for a large range of δ (but not $\delta = 0$).

The proof proceeds by first considering the generalisation of ALL-MIN-SAT to the version where *minority*, i.e. less than $\frac{1}{2}$ of the assignments, is replaced with another threshold. This problem is also $\mathbf{coNP}^{\#\mathbf{P}}$-hard for a range of thresholds. Note however, if this threshold is exactly 1 the problem is true for all formulae, and if the threshold is 0 the problem is simply asks if the formula is unsatisfiable (a **coNP** problem).

This generalised problem can then be reduced to deciding DECIDE-ε,δ-DP, where the threshold corresponds exactly to δ. It will turn out in the resulting circuit ε does not change the status of differential privacy, i.e. it is (ε,δ)-differentially private for all ε, or not.

The proof shows hardness for $\Omega(n)$-input bits and 1-output bit; the case in which there also exists a $\mathbf{coNP}^{\#\mathbf{P}}$ upper-bound. Hence, showing hardness in a higher complexity class, e.g., $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$, would require a reduction to a circuit with more output bits.

5 Inapproximability of the privacy parameters ε, δ

Given the difficulty of deciding if a circuit is differentially private, one might naturally consider whether approximating ε or δ could be efficient. We show that these tasks are both **NP**-hard and **coNP**-hard.

We show that distinguishing between (ε,δ), and (ε',δ')-differential privacy is **NP**-hard, by reduction from a problem we call NOT-CONSTANT which we also show is **NP**-hard. A boolean formula is in NOT-CONSTANT if it is satisfiable and not also a tautology.

▶ **Lemma 20.** *NOT-CONSTANT is* **NP***-complete. (hence CONSTANT is* **coNP***-complete).*

Proof. Clearly, Not-Constant $\in$ **NP**, the witness being a pair of satisfying and non-satisfying assignments. We reduce 3-SAT to Not-Constant. Given a Boolean formula ϕ over variables $x_1, \ldots, x_n$ let $\phi'(x_1, \ldots, x_n, x_{n+1}) = \phi(x_1, \ldots, x_n) \wedge x_{n+1}$. Note that ϕ' is never a tautology as $\phi'(x_1, \ldots, x_n, 0) = 0$. Furthermore, ϕ' is satisfiable iff ϕ is. ◀

In Definition 13 we used randomized response in the pure differential privacy setting. We now consider the approximate differential privacy variant $RR_{\varepsilon,\delta} : \{0,1\} \to \{\top, \bot\} \times \{0,1\}$ defined as follows:

$$RR_{\varepsilon,\delta}(x) = \begin{cases} (\top, x) & \text{w.p. } \delta \\ (\bot, x) & \text{w.p. } (1-\delta)\frac{\alpha}{1+\alpha} \\ (\bot, \neg x) & \text{w.p. } (1-\delta)\frac{1}{1+\alpha} \end{cases} \quad \text{where} \quad \alpha = e^{\varepsilon}$$

I.e., with probability δ, $RR_{\varepsilon,\delta}(x)$ reveals x and otherwise it executes $RR_{\varepsilon}(x)$. The former is marked with "$\top$" and the latter with "$\bot$". This mechanism is equivalent to the one described in [35] and is (ε, δ)-differentially private.

▶ **Definition 21.** *Let $0 \leq \varepsilon \leq \varepsilon'$, $0 \leq \delta \leq \delta' \leq 1$, with either $\varepsilon < \varepsilon'$ or $\delta < \delta'$. The problem* Distinguish-$(\varepsilon, \delta), (\varepsilon', \delta')$-DP *takes as input a circuit ψ, guaranteed to be either (ε, δ)-differentially private, or (ε', δ')-differentially private. The problem asks whether ψ is (ε, δ)-differentially private or (ε', δ')-differentially private.*

▶ **Lemma 22.** *Distinguish-$(\varepsilon, \delta), (\varepsilon', \delta')$-DP is* **NP***-hard (and* **coNP***-hard).*

Proof. We reduce Not-Constant to Distinguish-$(\varepsilon, \delta), (\varepsilon', \delta')$-DP. Given the boolean formula $\phi(\boldsymbol{x})$ on n bits, we create a probabilistic circuit ψ. The input to ψ consists of the n bits $\boldsymbol{x}$ plus a single bit y. The circuit ψ has four output bits (o_1, o_2, o_3, o_4) such that $(o_1, o_2) = RR_{\varepsilon,\delta}(y)$ and $(o_3, o_4) = RR_{\varepsilon',\delta'}(\phi(\boldsymbol{x}))$.

Observe that $(o_1, o_2) = RR_{\varepsilon,\delta}(y)$ is always (ε, δ) differentially private. As for $(o_3, o_4) = RR_{\varepsilon',\delta'}(\phi(\boldsymbol{x}))$, if $\phi \in$ Not-Constant then there are adjacent $\boldsymbol{x}, \boldsymbol{x}'$ such that $\phi(\boldsymbol{x}) \neq \phi(\boldsymbol{x}')$. In this case, $(o_3, o_4) = RR_{\varepsilon',\delta'}(\phi(\boldsymbol{x}))$ is (ε', δ')-differentially private, and, because $(\varepsilon, \delta) < (\varepsilon', \delta')$, so is ψ . On the other hand, if $\phi \notin$ Not-Constant then $\phi(\boldsymbol{x})$ does not depend on $\boldsymbol{x}$ and hence (o_3, o_4) does not affect privacy, in which case we get that ψ is (ε, δ) differentially private.

The same argument also gives **coNP**-hardness. ◀

Notice that the above theorem holds when $\delta = \delta'$ and $\varepsilon < \varepsilon'$ (similarly, $\varepsilon = \varepsilon'$ and $\delta < \delta'$), which entails the following theorem:

▶ **Theorem 23.** *Assuming* **P** $\neq$ **NP***, for any approximation error $\gamma > 0$, there does not exist a polynomial time approximation algorithm that given a probabilistic circuit ψ and δ computes some $\hat{\varepsilon}$, where $|\hat{\varepsilon} - \varepsilon| \leq \gamma$ and ε is the minimal such that ψ is (ε, δ)-differentially private within error γ. Similarly, given ε, no such $\hat{\delta}$ can be computed polynomial time where $|\hat{\delta} - \delta| \leq \gamma$ and δ is minimal.*

▶ Remark 24. The result also applies when approximating within a given ratio $\rho > 1$ (e.g. in the case of approximating ε, to find $\hat{\varepsilon}$ such that $\frac{\hat{\varepsilon}}{\varepsilon} \leq \rho$). Moreover, the result also holds when approximating pure differential privacy, that is when $\delta = 0$.

6 Related work

Differential privacy was introduced in [22]. It is a definition of privacy in the context of data analysis capturing the intuition that information specific to an individuals is protected if every single user's input has a bounded influence on the computation's outcome distribution, where the bound is specified by two parameters, usually denoted by ε, δ. Intuitively, these parameters set an upperbound on privacy loss, where the parameter ε limits the loss and the parameter δ limits the probability in which the loss may exceed ε.

Extensive work has occurred in the computer-assisted or automated of verification of differential privacy. Early work includes, PINQ [33] and Airavat [38] which are systems that keep track of the privacy budgets (ε and δ) using trusted privacy primitives in SQL-like and MapReduce-like paradigms respectively. In other work, programming languages were developed, that use the type system to keep track of the sensitivity and ensure the correct level of noise is added [37, 9, 16, 8]. Another line of work uses proof assistants to help prove that an algorithm is differentially private [7]; although much of this work is not automated, recent work has gone in this direction [2, 44].

These techniques focuses on "soundness", rather than "completeness" thus are not amenable to complexity analysis. In the constrained case of verifying differential privacy on probabilistic automata and Markov chains there are bisimulation based techniques [40, 12]. Towards complexity analysis; [15] shows that computing the optimal value of δ for a finite labelled Markov chain is undecidable. Further [14] and [15] provides distances, which are (necessarily) not tight, but can be computed in polynomial time with an **NP** oracle and a weaker bound in polynomial time. Recent works have focused on developing techniques for finding violations of differential privacy [19, 10]. The methods proposed so far have been based on some form of testing. Our result limits also the tractability of these approaches. Finally, [5] proposes an automated technique for proving differential privacy or finding counterexamples. This paper studies a constrained class of programs extending the language we presented here, and provides a "complete" procedure for deciding differential privacy for them. The paper does not provide any complexity guarantee for the proposed method and we expect our results to apply also in their setting.

As we already discussed, Murtagh and Vadhan [35] showed that finding the optimal values for the privacy parameters when composing different algorithms in a black-box way is #**P**-complete, but also that approximating the optimal values can be done efficiently. In contrast, our results show that when one wants to consider programs as white-box, as often needed to achieve better privacy guarantees (e.g. in the case of the sparse vector technique), the complexity is higher.

Several works have explored different property testing related to differential privacy [20, 29, 26], including verification [26]. In the standard model used in property testing, a user has only black-box access to the function and the observable outputs are the ones provided by a privacy mechanism. In contrast, our work is based on the program description and aim to provide computational limits to the design of techniques for program analyses for differential privacy.

We already discussed some works on quantitative information flow. In addition to those, it was shown that comparing the quantitative information flow of two programs on inputs coming from the uniform distribution is #**P**-hard [43]. However, when quantifying over all distributions the question is **coNP**-complete [43].

As we remarked earlier, our language is equally expressive when integers of a fixed size are added. Recently Jacomme, Kremer and Barthe [28] show deciding equivalence of two such programs, operating over a fixed finite field, is $\mathbf{coNP}^{\mathbf{C_{=}P}}$-complete and the majority problem, which is similar to pure differential privacy, is $\mathbf{coNP}^{\mathbf{PP}}$-complete – matching the class we show for deciding ε-differential privacy. Further the universal equivalence problem, which shows the programs are equivalent over all field extensions, is decidable in 2-**EXP**; the universal majority problem is not know to be decidable.

7 Conclusions and future work

Verifying differential privacy of loop-free probabilistic boolean programs

We have shown the difficulty of verifying differential privacy in loop-free probabilistic boolean programs through their correspondence with probabilistic circuits. Deciding ε-differential privacy is $\mathbf{coNP}^{\#\mathbf{P}}$-complete and (ε, δ)-differential privacy is $\mathbf{coNP}^{\#\mathbf{P}}$-hard and in $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$ (a gap that we leave for future work). Both problems are positioned in the counting hierarchy, in between the polynomial hierarchy **PH** and **PSPACE**.

Verifying differential privacy of probabilistic boolean programs

One interesting question that our work leaves open is the characterization of the complexity of deciding differential privacy problems for probabilistic boolean programs, including loops. Similarly to the works on quantitative information flow [11], we expect these problems to be decidable and we expect them to be in **PSPACE**. However, this question requires some further investigation that we leave for future work.

Solvers mixing non-determinism and counting

Returning to our motivation for this work – developing practical tools for verifying differential privacy – our results seem to point to a deficiency in available tools for model checking. The model checking toolkit includes well established Sat solvers for **NP** (and **coNP**) problems, solvers for further quantification in **PH**, solvers for #Sat (and hence for #**P** problems[3]). However to the best of our knowledge, there are currently no solvers that are specialized for mixing the polynomial hierarchy **PH** and counting problems #**P**, in particular $\mathbf{coNP}^{\#\mathbf{P}}$ and $\mathbf{coNP}^{\#\mathbf{P}^{\#\mathbf{P}}}$.

Approximating the differential privacy parameters

We show that distinguishing (ε, δ)-differential privacy from (ε', δ') differential privacy where $(\varepsilon, \delta) < (\varepsilon', \delta')$ is both **NP**- and **coNP**-hard. We leave refining the classification of this problem as an open problem.

References

1 John M. Abowd. The U.S. census bureau adopts differential privacy. In Yike Guo and Faisal Farooq, editors, *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, page 2867. ACM, 2018. doi:10.1145/3219819.3226070.

2 Aws Albarghouthi and Justin Hsu. Synthesizing coupling proofs of differential privacy. *Proc. ACM Program. Lang.*, 2(POPL):58:1–58:30, 2018. doi:10.1145/3158146.

[3] See, for example, http://beyondnp.org/pages/solvers/, for a range of solvers.

3 Mário S. Alvim, Miguel E. Andrés, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. On the relation between differential privacy and quantitative information flow. In Luca Aceto, Monika Henzinger, and Jirí Sgall, editors, *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part II*, volume 6756 of *Lecture Notes in Computer Science*, pages 60–76. Springer, 2011. doi:10.1007/978-3-642-22012-8_4.

4 Apple. Apple differential privacy technical overview. URL: https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf.

5 Gilles Barthe, Rohit Chadha, Vishal bibsource = self, Jagannath, A Prasad Sistla, and Mahesh Viswanathan. Deciding differential privacy for programs with finite inputs and outputs. In *LICS 2020 (to appear)*, 2020. arXiv preprint: arXiv:1910.04137.

6 Gilles Barthe, Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Pierre-Yves Strub. Higher-order approximate relational refinement types for mechanism design and differential privacy. In Sriram K. Rajamani and David Walker, editors, *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2015, Mumbai, India, January 15-17, 2015*, pages 55–68. ACM, 2015. doi:10.1145/2676726.2677000.

7 Gilles Barthe, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. Proving differential privacy via probabilistic couplings. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 749–758. ACM, 2016. doi:10.1145/2933575.2934554.

8 Gilles Barthe, Marco Gaboardi, Justin Hsu, and Benjamin C. Pierce. Programming language techniques for differential privacy. *SIGLOG News*, 3(1):34–53, 2016. URL: https://dl.acm.org/citation.cfm?id=2893591.

9 Gilles Barthe, Boris Köpf, Federico Olmedo, and Santiago Zanella Béguelin. Probabilistic relational reasoning for differential privacy. In John Field and Michael Hicks, editors, *Proceedings of the 39th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2012, Philadelphia, Pennsylvania, USA, January 22-28, 2012*, pages 97–110. ACM, 2012. doi:10.1145/2103656.2103670.

10 Benjamin Bichsel, Timon Gehr, Dana Drachsler-Cohen, Petar Tsankov, and Martin T. Vechev. Dp-finder: Finding differential privacy violations by sampling and optimization. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 508–524. ACM, 2018. doi:10.1145/3243734.3243863.

11 Rohit Chadha, Dileep Kini, and Mahesh Viswanathan. Quantitative information flow in boolean programs. In Martín Abadi and Steve Kremer, editors, *Principles of Security and Trust - Third International Conference, POST 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, volume 8414 of *Lecture Notes in Computer Science*, pages 103–119. Springer, 2014. doi:10.1007/978-3-642-54792-8_6.

12 Konstantinos Chatzikokolakis, Daniel Gebler, Catuscia Palamidessi, and Lili Xu. Generalized bisimulation metrics. In Paolo Baldan and Daniele Gorla, editors, *CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings*, volume 8704 of *Lecture Notes in Computer Science*, pages 32–46. Springer, 2014. doi:10.1007/978-3-662-44584-6_4.

13 Dmitry Chistikov, Rayna Dimitrova, and Rupak Majumdar. Approximate counting in SMT and value estimation for probabilistic programs. *Acta Inf.*, 54(8):729–764, 2017. doi:10.1007/s00236-017-0297-2.

14 Dmitry Chistikov, Andrzej S. Murawski, and David Purser. Bisimilarity distances for approximate differential privacy. In Shuvendu K. Lahiri and Chao Wang, editors, *Automated Technology for Verification and Analysis - 16th International Symposium, ATVA 2018, Los Angeles, CA, USA, October 7-10, 2018, Proceedings*, volume 11138 of *Lecture Notes in Computer Science*, pages 194–210. Springer, 2018. `doi:10.1007/978-3-030-01090-4_12`.

15 Dmitry Chistikov, Andrzej S. Murawski, and David Purser. Asymmetric distances for approximate differential privacy. In Wan Fokkink and Rob van Glabbeek, editors, *30th International Conference on Concurrency Theory, CONCUR 2019, August 27-30, 2019, Amsterdam, the Netherlands*, volume 140 of *LIPIcs*, pages 10:1–10:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. `doi:10.4230/LIPIcs.CONCUR.2019.10`.

16 Loris D'Antoni, Marco Gaboardi, Emilio Jesús Gallego Arias, Andreas Haeberlen, and Benjamin C. Pierce. Sensitivity analysis using type-based constraints. In Richard Lazarus, Assaf J. Kfoury, and Jacob Beal, editors, *Proceedings of the 1st annual workshop on Functional programming concepts in domain-specific languages, FPCDSL@ICFP 2013, Boston, Massachusetts, USA, September 22, 2013*, pages 43–50. ACM, 2013. `doi:10.1145/2505351.2505353`.

17 Dorothy E. Denning. *Cryptography and Data Security.* Addison-Wesley, 1982.

18 Differential Privacy Team, Apple. Learning with privacy at scale, 2017. URL: `https://machinelearning.apple.com/docs/learning-with-privacy-at-scale/appledifferentialprivacysystem.pdf`.

19 Zeyu Ding, Yuxin Wang, Guanhong Wang, Danfeng Zhang, and Daniel Kifer. Detecting violations of differential privacy. In David Lie, Mohammad Mannan, Michael Backes, and XiaoFeng Wang, editors, *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, CCS 2018, Toronto, ON, Canada, October 15-19, 2018*, pages 475–489. ACM, 2018. `doi:10.1145/3243734.3243818`.

20 Kashyap Dixit, Madhav Jha, Sofya Raskhodnikova, and Abhradeep Thakurta. Testing the lipschitz property over product distributions with applications to data privacy. In Amit Sahai, editor, *Theory of Cryptography - 10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 418–436. Springer, 2013. `doi:10.1007/978-3-642-36594-2_24`.

21 Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503. Springer, 2006. `doi:10.1007/11761679_29`.

22 Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006. `doi:10.1007/11681878_14`.

23 Cynthia Dwork, Guy N. Rothblum, and Salil P. Vadhan. Boosting and differential privacy. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 51–60. IEEE Computer Society, 2010. `doi:10.1109/FOCS.2010.12`.

24 Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. RAPPOR: randomized aggregatable privacy-preserving ordinal response. In Gail-Joon Ahn, Moti Yung, and Ninghui Li, editors, *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, Scottsdale, AZ, USA, November 3-7, 2014*, pages 1054–1067. ACM, 2014. `doi:10.1145/2660267.2660348`.

25 Matthew Fredrikson and Somesh Jha. Satisfiability modulo counting: a new approach for analyzing privacy properties. In Thomas A. Henzinger and Dale Miller, editors, *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS '14, Vienna, Austria, July 14 - 18, 2014*, pages 42:1–42:10. ACM, 2014. doi:10.1145/2603088.2603097.

26 Anna C. Gilbert and Audra McMillan. Property testing for differential privacy. In *56th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2018, Monticello, IL, USA, October 2-5, 2018*, pages 249–258. IEEE, 2018. doi:10.1109/ALLERTON.2018.8636068.

27 J. W. Gray. Probabilistic interference. In *Proceedings. 1990 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 170–179, May 1990. doi:10.1109/RISP.1990.63848.

28 Charlie Jacomme, Steve Kremer, and Gilles Barthe. Universal equivalence and majority on probabilistic programs over finite fields. In *LICS 2020 (to appear)*, 2020.

29 Madhav Jha and Sofya Raskhodnikova. Testing and reconstruction of lipschitz functions with applications to data privacy. *SIAM J. Comput.*, 42(2):700–731, 2013. doi:10.1137/110840741.

30 Dexter Kozen. Semantics of probabilistic programs. *J. Comput. Syst. Sci.*, 22(3):328–350, 1981. doi:10.1016/0022-0000(81)90036-2.

31 Michael L. Littman, Judy Goldsmith, and Martin Mundhenk. The computational complexity of probabilistic planning. *J. Artif. Intell. Res.*, 9:1–36, 1998. doi:10.1613/jair.505.

32 Min Lyu, Dong Su, and Ninghui Li. Understanding the sparse vector technique for differential privacy. *Proc. VLDB Endow.*, 10(6):637–648, 2017. doi:10.14778/3055330.3055331.

33 Frank McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In Ugur Çetintemel, Stanley B. Zdonik, Donald Kossmann, and Nesime Tatbul, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2009, Providence, Rhode Island, USA, June 29 - July 2, 2009*, pages 19–30. ACM, 2009. doi:10.1145/1559845.1559850.

34 Ilya Mironov. On significance of the least significant bits for differential privacy. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, pages 650–661. ACM, 2012. doi:10.1145/2382196.2382264.

35 Jack Murtagh and Salil P. Vadhan. The complexity of computing the optimal composition of differential privacy. In Eyal Kushilevitz and Tal Malkin, editors, *Theory of Cryptography - 13th International Conference, TCC 2016-A, Tel Aviv, Israel, January 10-13, 2016, Proceedings, Part I*, volume 9562 of *Lecture Notes in Computer Science*, pages 157–175. Springer, 2016. doi:10.1007/978-3-662-49096-9_7.

36 Nicolas Papernot, Martín Abadi, Úlfar Erlingsson, Ian J. Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL: https://openreview.net/forum?id=HkwoSDPgg.

37 Jason Reed and Benjamin C. Pierce. Distance makes the types grow stronger: a calculus for differential privacy. In Paul Hudak and Stephanie Weirich, editors, *Proceeding of the 15th ACM SIGPLAN international conference on Functional programming, ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010*, pages 157–168. ACM, 2010. doi:10.1145/1863543.1863568.

38 Indrajit Roy, Srinath T. V. Setty, Ann Kilzer, Vitaly Shmatikov, and Emmett Witchel. Airavat: Security and privacy for mapreduce. In *Proceedings of the 7th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2010, April 28-30, 2010, San Jose, CA, USA*, pages 297–312. USENIX Association, 2010. URL: http://www.usenix.org/events/nsdi10/tech/full_papers/roy.pdf.

39 Seinosuke Toda. PP is as hard as the polynomial-time hierarchy. *SIAM J. Comput.*, 20(5):865–877, 1991. doi:10.1137/0220053.

40 Michael Carl Tschantz, Dilsun Kirli Kaynar, and Anupam Datta. Formal verification of differential privacy for interactive systems (extended abstract). In Michael W. Mislove and Joël Ouaknine, editors, *Twenty-seventh Conference on the Mathematical Foundations of Programming Semantics, MFPS 2011, Pittsburgh, PA, USA, May 25-28, 2011*, volume 276 of *Electronic Notes in Theoretical Computer Science*, pages 61–79. Elsevier, 2011. doi:10.1016/j.entcs.2011.09.015.

41 Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

42 Hirotoshi Yasuoka and Tachio Terauchi. On bounding problems of quantitative information flow. In Dimitris Gritzalis, Bart Preneel, and Marianthi Theoharidou, editors, *Computer Security - ESORICS 2010, 15th European Symposium on Research in Computer Security, Athens, Greece, September 20-22, 2010. Proceedings*, volume 6345 of *Lecture Notes in Computer Science*, pages 357–372. Springer, 2010. doi:10.1007/978-3-642-15497-3_22.

43 Hirotoshi Yasuoka and Tachio Terauchi. Quantitative information flow - verification hardness and possibilities. In *Proceedings of the 23rd IEEE Computer Security Foundations Symposium, CSF 2010, Edinburgh, United Kingdom, July 17-19, 2010*, pages 15–27. IEEE Computer Society, 2010. doi:10.1109/CSF.2010.9.

44 Danfeng Zhang and Daniel Kifer. Lightdp: towards automating differential privacy proofs. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 888–901. ACM, 2017. URL: http://dl.acm.org/citation.cfm?id=3009884.