Efficient Decision Procedures for Differential Privacy via Probabilistic Couplings

Azadeh Farzan
University of Toronto

Sky Li
University of Toronto

Aleksandar Nikolov
University of Toronto

Vishnu Nittoor
University of Toronto

1 Introduction

Differential privacy (DP) has been seeing ever wider adoption in recent years, and has become a *de facto* standard for private statistical analysis. Both private companies such as Google and Apple and government bodies such as the United States Census Bureau have announced their adoption of DP algorithms for their data collection and release procedures [1, 3, 2, 4]. However, it can be tricky to analyze DP algorithms; in one notable example, many different iterations of an algorithm known as the Sparse Vector Technique (SVT) algorithm have been produced and supposedly proven correct, but were later shown to fail to achieve finite privacy loss [16].

The difficulty of ensuring that DP algorithms are truly private has motivated the development of automated tools to formally verify that DP algorithms meet their claimed privacy bounds. However, it is known that complete verification of differentially private algorithms is undecidable, even for a relatively limited class of programs [8]. One way to circumvent these negative results is to restrict the class of programs we analyze further, such as requiring that every program can take input and output only from a finite domain, or limiting programs to branching on real-valued comparisons [8, 11, 12]. Another approach has been to develop heuristic or incomplete techniques for automatically generating proofs of privacy for DP algorithms. An especially notable tool for heuristic approaches is a construct known as an **approximate lifting** [9, 10, 14, 7, 6]. Approximate liftings are a generalization of probabilistic couplings, themselves a well-known technique in probability theory for analyzing relationships between random variables. Approximate liftings allow for a more structured approach to the privacy analysis of many algorithms that may not be conducive to a simple analysis via standard DP tools like composition theorems. Examples of algorithms that can be more easily analyzed using approximate liftings are SVT, Report Noisy Max, and Between Thresholds.

There are direct connections between approximate liftings and privacy; for example, the existence of a certain type of "equality" coupling for an algorithm is exactly equivalent to the algorithm itself being differentially private [9]. However, it can be extremely non-trivial to show that an arbitrary approximate lifting is valid—in the case of "equality" couplings, it can require essentially doing a manual (human) proof of privacy for the algorithm. We analyze a specific class of **"shift" couplings**, based on coupling noise random variables sampled by the algorithm. These couplings are easy to **algorithmically construct**, and, while their existence is not guaranteed to be *equivalent* to the privacy of an algorithm, it still implies differential privacy.

Perhaps surprisingly, we show that shift couplings can *also* be used to construct *complete* decision procedures for privacy for suitably constrained models of computation. Inspired by algorithms like SVT, we construct a limited language-theoretic program model that takes in a stream of *query values*, that is, real-valued functions of some underlying dataset, and compares them to a persistent stored threshold variable, after adding noise to both the variable and the query value. As one application, such a program model can be used to implement algorithms that detect distribution shift, i.e., to detect that a parameter of the population has changed its value significantly. They are also used as subroutines in other algorithms, for example in private multiplicative weights [13].

We first demonstrate how to construct shift couplings for programs in our model, starting with individual program transitions, and then "straight line programs", and full programs, which we model, respectively, as individual characters in a finite alphabet, words constructed from those characters, and a regular language comprised of those words. Through our language-theoretic framework, we show that it is possible to decompose any, possibly infinitely sized, program into a finite set of smaller, "periodic", programs that can each be analyzed independently and efficiently. We then also show that, under a mild additional constraint on our model, if no shift coupling proof exists, then the program cannot be differentially private. Moreover, since we can also decide in linear time if a coupling proof of the required type exists for a given program, we also derive a linear-time *decision procedure* for the privacy of programs in our model. We further provide an approach to *optimize* coupling proofs, providing tighter bounds on the privacy parameter ε for a given program.

Our proof that approximate liftings are complete for our model, in the sense above, builds on an equivalence that we show between a subclass of programs in our model and the automata-theoretic model DiPA [11]. We argue that coupling proofs reframe and re-explain the results of [11] in a natural and intuitive manner.

Finally, we extend our program model to accommodate an arbitrary number of threshold variables. We also generalize the construction of coupling proofs of privacy to this more general model. This generalization is an illustration of the flexibility and power of approximate liftings as a method of giving privacy proofs. Interestingly, the generalization also introduces new subtleties, and we need to introduce a new coupling strategy, which we call cross couplings, that does not have an analogue in the one variable model.

We also generalize the optimization procedure for single variable coupling proofs to an incomplete optimization procedure for multiple variable programs. Finally, we show that, for two variable programs, shift coupling proofs *remain complete for deciding privacy*, conjecture that that this remains the case for three and more variables. If our conjecture holds, then coupling proofs would characterize the privacy of a natural and rich class of programs.

2 Our Model and Results

Our program model accepts a *stream* of real-valued inputs, taken to be query functions of an underlying dataset. Each program is broken down into individual program steps, or **transitions** that operate as follows: at each transition, the program reads in an input value, adds Laplace noise to the input value, and compares the input value to a stored real-valued variable. Depending on the result of the comparison, the program **takes** the transition, **outputs** a value, and optionally **assigns** the noisy input into the real-valued variable. As mentioned, we take each transition to be a single character in our language-theoretic approach.

We first show that approximate liftings can be constructed for program transitions in isolation. Specifically, we define a family of "**coupling strategies**" parameterized by three real-valued "shifts" $\gamma_x, \gamma_t, \gamma'_t \in [-1, 1]$, where the noise variables used by the transition on one input are coupled with shifted copies of the noise variables used on a neighboring input. In particular, if the shifts satisfy certain **validity** conditions, then we can immediately construct a coupling-based proof of privacy for the individual transition. Of note, every coupling proof is associated with a **privacy cost** that directly corresponds to the ε in ε-differential privacy.

By sequentially concatenating transitions, we construct "straight line programs" (SLPs), which model a full execution of a program. Equivalently, each SLP is a word comprised of characters from a finite alphabet of transitions. We show that we can also concatenate coupling strategies associated with each transition to create a family of coupling strategies for an entire SLP. In particular, the validity conditions of individual coupling strategies for transitions also combine cleanly to define a constraint system that defines **validity** for SLP coupling strategies. Just as with individual transitions, we show that if a coupling strategy for an SLP is **valid** with cost ε, then the SLP itself is ε-differentially private.

Full programs in our model are modeled as a structured subset of regular languages over some finite alphabet of transitions. In particular, programs must be "generated" by an underlying finite-state control flow graph that supports standard looping and branching operations. Additionally, the underlying graph of a program must satisfy certain properties, such as determinism.

Naively, each constituent SLP of a program must be assigned its own coupling strategy, which could be computationally intractable since programs can be infinitely sized due to loops. Fortunately, it is known (see, for example, [17, 5]) that every regular language can be **decomposed** into a finite union of **union-free**

regular languages. We thus decompose every program into such a finite set of union-free regular languages, which we individually call **periodic programs**. We show that each periodic program can be assigned a *single* coupling strategy, thus making the overall problem of finding coupling strategies for full programs tractable.

We also show that a single additional condition for coupling strategies determines whether or not a coupling proof for a periodic program has finite cost or not.

Lemma 2.1. *For a periodic program L a valid coupling strategy $C = (\gamma, \gamma')$ has finite cost if and only if C assigns 0 privacy cost to every transition contained within a cycle in the underlying control flow graph of L.*

We can bundle this **finite cost constraint** with the **validity constraints** on coupling strategies to create a **privacy constraint system** for any periodic program L.

Lemma 2.2. *If the privacy constraint system for a periodic program L is satisfiable, then L is ε-differentially private for some $\varepsilon > 0$.*

We further define the class of **output-distinct** programs, which, intuitively, captures programs for which the transitions taken can be recovered from the output sequence of the program. We show that shift couplings are complete for deciding the privacy of output distinct programs.

Theorem 2.3. *An output-distinct program P is $\varepsilon-$differentially private for some finite $\varepsilon > 0$ if and only if, for every periodic program L of a decomposition of P, the privacy constraint system for L is satisfiable.*

Beyond showing that such coupling proof systems are complete, we also provide algorithms for finding and optimizing the cost of coupling proofs. We consider this problem through the framework of **constraint optimization**: what is the minimal cost coupling strategy that satisfies the **privacy constraint system**?

We state the exact problem for a single periodic program, which we call the **coupling cost optimization problem**, and conjecture that the cost is *privacy-optimal*; that is, the "true" privacy cost of a program **exactly matches** the optimal solution to the coupling cost optimization problem for the program.

Unfortunately, it is unknown whether or not an efficient (i.e. polynomial time) algorithm for solving the exact coupling cost optimization problem exists. In particular, the exact formulation requires solving a nested optimization problem. In lieu of such an algorithm, we introduce an **approximate** coupling cost optimization problem, which makes assumptions about an optimization parameter such that the originally nested optimization problem becomes a single layer optimization problem. This simplification allows the approximate cost problem for a periodic program to be solved in **polynomial time**[1] by an application of the ellipsoid method. We show that the approximation produces valid coupling proofs and that the approximation factor is bounded by a term linear in the number of non-cyclic transitions in L.

Proposition 2.4. *There exists a valid coupling strategy C_L for any differentially private periodic program L such that the privacy cost of C_L matches the optimal solution to the approximate coupling cost problem for L. Further, $opt(L) \leq approx(L) \leq opt(L) + \sum_{i \in I} d_i + \sum_{\sigma_i = \mathtt{insample'}} d'_i$, where I is the set of transitions in L that do not appear in a cycle and d_i, d'_i are program specific constants.*

The constraints derived from the approximate cost problem additionally allow us to define a linear-time algorithm for the problem of *deciding* whether or not *any* finite $\varepsilon > 0$ exists such that a program is ε-differentially private. Importantly, we show that the approximate cost constraints are satisfiable if and only if the privacy constraint system of a periodic program is satisfiable. Our algorithm constructs a **privacy constraint graph**, which directly maps from the approximate cost constraints into edges of the graph; we can check for privacy by looking for certain "contradictory" paths in the graph. Notably, we show how to construct a privacy constraint graph for an *full* program, not just individual periodic programs, and show that the overall algorithm has linear time complexity in the size of the underlying control flow graph of a program.

Theorem 2.5. *Given an output-distinct program P generated by a control flow graph G, we can decide if P is differentially private in linear time in the size of G.*

[1]Note that a single program may possibly contain exponentially many periodic programs.

As mentioned, part of our completeness proof flows from an equivalence between our program model and a previously introduced program model, known as DiPA [11]. DiPA is an automata-theoretic model that also models a class of algorithms based on comparing real-valued *queries* to a stored threshold variable. Indeed, we show that this class of algorithms is equivalent to the class of output-distinct program in our model.

Notably, the privacy of a DiPA program can be decided through a decision procedure that checks for the existence of four particular problematic graph structures in the automata graph of the program; a DiPA program is private if and only if none of the problematic graph structures appear in the graph (i.e. it is "**well-formed**"). Perhaps surprisingly, we show a direct connection between the existence of these graph structures and the satisfiability of the privacy constraint system for a program.

Theorem 2.6. *An output-distinct program P contains a periodic program whose privacy constraint system is unsatisfiable if and only if the corresponding DiPA A_P of P is well-formed.*

Interestingly, coupling-based proofs also allow for straightforward *generalizations* to different program models in a similar paradigm. We specifically apply coupling proofs to prove the privacy of an extended program model that allows for an *arbitrary* number of threshold variables to compare inputs against. A similar multivariable extension of DiPA was considered before, but it only allowed for conjunctions between variable guards, and required input noise variables to be correlated [12]

Analogously to the single variable case, we define a family of "shift" couplings for programs; however, this time each shift coupling family is defined independently for *each* program variable. Indeed, we show that we can create coupling strategies "**in parallel**" by analyzing each program variable in isolation and compose them together to create a coupling strategy for the entire program.

We also introduce a new type of coupling, which we call **cross couplings**. Cross couplings are a type of coupling, compatible with existing "parallel" shift couplings, that allow for certain transitions whose transition conditions correspond to tautologies or contradictions to be coupled "for free".

We show that if any coupling proof strategy (using *either* parallel or cross couplings) exists for a multivariable program, then the program is private. Again as in the single variable case, we can characterize the existence of coupling proofs through a **privacy constraint system**.

Lemma 2.7. *If, for every constituent periodic program L of a multivariable program P, the (multivariable) privacy constraint system for L is satisfiable, then P is ε-differentially private for some finite $\varepsilon > 0$.*

Unfortunately, the conditions that allow for a cross coupling to be created are much more complex than the inequality-based constraints of parallel couplings (which themselves are derived directly from single variable couplings). Thus, it is unknown whether there exists a tractable algorithm for even approximately optimizing multivariable coupling cost. However, by ignoring cross couplings entirely, we can define an **incomplete** optimization problem consisting of parallel coupling constraints and a similarly incomplete decision algorithm for multivariable programs derived directly from the single variable analogue.

We also show that, for output-distinct programs with exactly two variables, coupling proofs are **still complete**. We again take a graph-theoretic approach: accounting for an exception raised by the introduction of cross-couplings, we show that the unsatisfiability of the privacy constraint system for a periodic program implies that the underlying program graph must contain one of four problematic graph structures. Through detailed analysis, we show that the existence of these four graph structures directly implies that the program cannot be ε-differentially private for any $\varepsilon > 0$. While this is similar to the analysis of DiPA for the single variable case, we need new input instance constructions to certify the lack of privacy for the multivariate case, and their analysis is more complicated than the single variables case.

Theorem 2.8. *An output-distinct two variable program P is ε-differentially private for some $\varepsilon > 0$ if and only if the privacy constraint system for every constituent periodic program of P is satisfiable.*

We conjecture that our completeness result extends to programs with three or more variables as well, and discuss possible proof approaches in the full version of our paper.

We propose multiple areas for future work. Most notably, the limited program model is ripe for extensions, both trivial and non-trivial. For example, in addition to extending our completeness results to programs with three or more variables, we suggest that it is possible to extend the definition of transition guards to encompass more complicated predicates (beyond real-valued comparisons) in both the single and multivariable cases.

The completeness of coupling proofs for both single variable and two variable threshold programs (and possibly, as we conjecture, general multivariable threshold programs) also suggests the possibility of using coupling proof techniques to *directly* discover and prove the validity of privacy violating input sequence pairs, in the spirit of current efforts at "auditing" DP machine learning algorithms [15, 18]. If such constructions exist, they could make the problem of finding counterexamples and lower bounds for privacy cost a much simpler task.

References

[1] How we're helping developers with differential privacy. https://developers.googleblog.com/2021/01/how-were-helping-developers-with-differential-privacy.html.

[2] Learning with Privacy at Scale. https://machinelearning.apple.com/research/learning-with-privacy-at-scale.

[3] Privacy - Features. https://www.apple.com/privacy/features/.

[4] John M. Abowd. The U.S. Census Bureau Adopts Differential Privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, page 2867, New York, NY, USA, July 2018. Association for Computing Machinery.

[5] Sergey Afonin and Denis Golomazov. Minimal Union-Free Decompositions of Regular Languages. In Adrian Horia Dediu, Armand Mihai Ionescu, and Carlos Martín-Vide, editors, *Language and Automata Theory and Applications*, Lecture Notes in Computer Science, pages 83–92, Berlin, Heidelberg, 2009. Springer.

[6] Aws Albarghouthi and Justin Hsu. Synthesizing coupling proofs of differential privacy. *Proceedings of the ACM on Programming Languages*, 2(POPL):58:1–58:30, December 2017.

[7] Aws Albarghouthi and Justin Hsu. Constraint-Based Synthesis of Coupling Proofs. In Hana Chockler and Georg Weissenbacher, editors, *Computer Aided Verification*, pages 327–346, Cham, 2018. Springer International Publishing.

[8] Gilles Barthe, Rohit Chadha, Vishal Jagannath, A. Prasad Sistla, and Mahesh Viswanathan. Deciding Differential Privacy for Programs with Finite Inputs and Outputs. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '20, pages 141–154, New York, NY, USA, July 2020. Association for Computing Machinery.

[9] Gilles Barthe, Marco Gaboardi, Benjamin Grégoire, Justin Hsu, and Pierre-Yves Strub. Proving Differential Privacy via Probabilistic Couplings. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '16, pages 749–758, New York, NY, USA, July 2016. Association for Computing Machinery.

[10] Gilles Barthe and Federico Olmedo. Beyond Differential Privacy: Composition Theorems and Relational Logic for f-divergences between Probabilistic Programs. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Fedor V. Fomin, Rūsiņš Freivalds, Marta Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming*, volume 7966, pages 49–60. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[11] Rohit Chadha, A. Prasad Sistla, and Mahesh Viswanathan. On linear time decidability of differential privacy for programs with unbounded inputs. In *Proceedings of the 36th Annual ACM/IEEE Symposium on Logic in Computer Science*, LICS '21, pages 1–13, New York, NY, USA, November 2021. Association for Computing Machinery.

[12] Rohit Chadha, A. Prasad Sistla, Mahesh Viswanathan, and Bishnu Bhusal. Deciding Differential Privacy of Online Algorithms with Multiple Variables. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, CCS '23, pages 1761–1775, New York, NY, USA, November 2023. Association for Computing Machinery.

[13] Moritz Hardt and Guy N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 61–70. IEEE Computer Society, 2010.

[14] Justin Hsu. *Probabilistic Couplings For Probabilistic Reasoning.* PhD thesis, University of Pennsylvania, January 2017.

[15] Fred Lu, Joseph Munoz, Maya Fuchs, Tyler LeBlond, Elliott Zaresky-Williams, Edward Raff, Francis Ferraro, and Brian Testa. A General Framework for Auditing Differentially Private Machine Learning, October 2022.

[16] Min Lyu, Dong Su, and Ninghui Li. Understanding the Sparse Vector Technique for Differential Privacy, September 2016.

[17] Benedek Nagy. Union-free regular languages and 1-cycle-free-path-automata. *Publicationes Mathematicae Debrecen*, 68(1-2):183–197, January 2006.

[18] Thomas Steinke, Milad Nasr, and Matthew Jagielski. Privacy Auditing with One (1) Training Run, May 2023.