Pufferfish: A Framework for Mathematical Privacy Definitions

DANIEL KIFER, Penn State University
ASHWIN MACHANAVAJJHALA, Duke University

In this paper we introduce a new and general privacy framework called Pufferfish. The Pufferfish framework can be used to create new privacy definitions that are customized to the needs of a given application. The goal of Pufferfish is to allow experts in an application domain, who frequently do not have expertise in privacy, to develop rigorous privacy definitions for their data sharing needs. In addition to this, the Pufferfish framework can also be used to study existing privacy definitions.

We illustrate the benefits with several applications of this privacy framework: we use it to analyze differential privacy and formalize a connection to attackers who believe that the data records are independent; we use it to create a privacy definition called hedging privacy, which can be used to rule out attackers whose prior beliefs are inconsistent with the data; we use the framework to define and study the notion of composition in a broader context than before; we show how to apply the framework to protect unbounded continuous attributes and aggregate information; and we show how to use the framework to rigorously account for prior data releases.

Categories and Subject Descriptors: H.2.8 [**Database Applications**]: Statistical Databases; K.4.1 [**Computers and Society**]: Privacy

General Terms: Theory

Additional Key Words and Phrases: privacy, differential privacy

1. INTRODUCTION

With improvements in data collection technologies, increased emphasis on data analytics, and the increasing need for different parties to share datasets, the field of *statistical privacy* is seeing an unprecedented growth in importance and diversity of applications. These applications include protecting privacy and confidentiality in computer network data collections [PREDICT 2005], protecting privacy and identifiability in genome-wide association studies (GWAS) [Homer et al. 2008], protecting confidentiality in Census data products [Machanavajjhala et al. 2008; Fienberg 1997], etc. In each case, the goal is to release useful information (i.e. privacy-preserving query answers or a sanitized version of the dataset) while ensuring that confidential information cannot be inferred from the data release. This is achieved by using algorithms, called *privacy mechanisms*, whose outputs allow researchers to learn statistical properties of the data. The behavior of these privacy mechanisms is governed by *privacy definitions*, which limit the amount of information disclosed about individual records.

A number of privacy definitions and mechanisms have been proposed in the literature (see surveys [Adam and Worthmann 1989; Chen et al. 2009; Dwork 2008]). How-

This material is based upon work supported by the National Science Foundation under Grants No. 1054389 and 1253327, and a gift from Google.
Author's addresses: D. Kifer, Department of Computer Science & Engineering, 360F IST Building, University Park, PA 16802; A. Machanavajjhala, Department of Computer Science, D329 Levine Research Science Building, Duke University, Durham, NC 27708.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

ever, adoption of those technologies is slow due to application-specific requirements that are not necessarily met by existing solutions. Such requirements are due to relationships between records in the data, different types of sensitive information, and utility considerations, each of which is elaborated below.

Cross-sectional surveys and censuses are tabular one-time snapshots where individual records are mostly independent of each other. On the other hand, user activity streams like GPS traces and social networks produce records that are correlated with each other. Recent work [Kifer and Machanavajjhala 2011; Gehrke et al. 2011] shows that attackers can use these correlations to improve their inferences about individuals (even when the data is protected by strong notions such as differential privacy [Dwork et al. 2006b; Dwork 2006]).

Variability in sensitive information comes in many forms. Some data sets (medical and military records) require more protection than others (movie ratings); some applications must protect aggregate secrets (database outsourcing) while others must protect secrets about individuals (Census data); in some cases, only certain attributes (or attribute values) need to be protected, and certain individuals may require more privacy protection than others.

Finally, in some applications, a certain level of utility must be achieved. As is the case with some Census data products, legal or contractual obligations can force the accurate release of some statistics about the data. Once these obligations have been met, how should one account for such information disclosure when planning future privacy-preserving data releases? A related concern is that an organization can only justify the expense of a privacy-preserving public data release if it provides a certain level of utility. There are fundamental limits on the utility permitted by current state-of-the-art privacy mechanisms [Dwork et al. 2007; Machanavajjhala et al. 2011]. It is also well-known [Dwork and Naor 2010; Kifer and Machanavajjhala 2011] that no privacy mechanism can provide both high utility and privacy against attackers with arbitrary background knowledge or beliefs about the data. Thus new advances will require a fine-grained ability to distinguish between realistic and worst-case attackers.

To address such problems, we propose a customizable framework called Pufferfish that makes it easier to generate new privacy definitions with rigorous statistical guarantees about the leakage of sensitive information. We also illustrate its use with many applications.

1.1. Contributions and Outline

The contributions of this paper are:

— A new Bayesian privacy framework which provides rigorous privacy guarantees against many types of attackers.

— A new privacy definition called hedging privacy and a histogram release algorithm for it. The goal of hedging privacy is protect against reasonable attackers while automatically ruling out those attackers whose beliefs about the data are implausible.

— An analysis of differential privacy within our framework. We present crisp Bayesian semantics for differential privacy in terms of data-generating distributions that model records independently.

— A principled extension of differential privacy that accounts for prior releases of non-differentially private information (without such an extension, a naive application can leak too much information [Kifer and Machanavajjhala 2011]).

— An application of the framework to study the notion of composition between privacy definitions and to provide conditions under which certain types of composition hold.

— A general application to datasets with unbounded continuous variables; in particular, we show how to prevent an attacker from accurately inferring the value of an unbounded continuous attribute (both in terms of relative and absolute error). We further show how this is related to the protection of aggregate secrets.

The rest of the paper is organized in the following way. In Section 2 we introduce basic concepts and notation. In Section 3 we present the Pufferfish framework. We discuss related work in Section 4. We show that Pufferfish satisfies fundamental privacy axioms in Section 5. Subsequent sections present a variety of applications of this framework. We use Pufferfish to analyze differential privacy and clarify its connections to data distributions that model records independently in Section 6. We present hedging privacy and a histogram publishing algorithm in Section 7. We show how to deal with continuous attributes and aggregate secrets in Section 8. We use Pufferfish to study composition in Section 9. We show how to provide privacy while accounting for prior data releases in Section 10. We explain how Pufferfish resists the de Finetti attack [Kifer 2009] in Section 11.

2. NOTATION AND TERMINOLOGY

Let $\mathcal{I}$ be the set of database instances that are possible for a given application. For datasets that are collections of records, we let $\mathcal{T}$ represent the domain of tuples, we use the symbol t to represent a value in the domain $\mathcal{T}$, and we use the variable r to represent a record. Note r is a random variable associated with an individual and $t \in \mathcal{T}$ is a value it can take. For ease of explanation and to simplify notation, we will assume each individual is associated with at most one record; we note that the Pufferfish framework does not need this assumption.

In the setting we consider, a *data curator* has a dataset $\mathfrak{Data}$. Let $\mathrm{records}(\mathfrak{Data})$ denote the set of records in $\mathfrak{Data}$ and let $\mathrm{tuples}(\mathfrak{Data})$ denote the record values (tuples). The data curator will choose a *privacy definition* and a *privacy mechanism* (algorithm) $\mathfrak{M}$ that satisfies that privacy definition. The data curator will then apply $\mathfrak{M}$ to the data to obtain a *sanitized output* $\omega \equiv \mathfrak{M}(\mathfrak{Data})$.

To an *attacker* (who does not know what the dataset is), $\mathfrak{Data}$ represents a *random variable*. We use the letter θ to represent a probability distribution and, for $D \in \mathcal{I}$, we will use the notation $P(\mathfrak{Data} = D \mid \theta)$ to represent the probability, under θ, that the true dataset is D. For convenience, we summarize the notation used in this paper in Table I.

3. THE PUFFERFISH FRAMEWORK

In this section we introduce the Pufferfish framework. This framework requires a domain expert to specify three crucial components: a set of *potential secrets* $\mathbb{S}$, a set of *discriminative pairs* $\mathbb{S}_{\text{pairs}} \subseteq \mathbb{S} \times \mathbb{S}$, and a collection of *data evolution scenarios* $\mathbb{D}$. Based on these three components the framework generates a rich class of privacy definitions.

The set of potential secrets $\mathbb{S}$, is an explicit specification of what we would like to protect.[1] Examples include statements such as *"the record for individual h_i is in the data"*, *"the record for individual h_i is <u>not</u> in the data"*, *"the query volume is* 1–5 *million queries"*, etc. Additional examples can be found in Sections 3.2, 6, 8, and 10. A statement $s \in \mathbb{S}$ need not be true for the actual dataset – attackers will form their own opinions about which statements are likely to be true or not. In general, a domain expert should add a statement s to the potential secrets $\mathbb{S}$ if either the claim that s is

[1] In general (not just for Pufferfish), this requirement is necessary to avoid confusion about whether or not a privacy definition is relevant to a particular application.

Table I. Table of Notation

$\mathcal{I}$	The set of possible database instances.
$\boldsymbol{\mathfrak{Data}}$	A random variable representing the true dataset (which is unknown to the attacker).
$\mathcal{T}$	The domain of tuples.
t	A tuple, a value in $\mathcal{T}$.
$\mathcal{H}$	The set of all individuals. $\mathcal{H} = \{h_1, h_2, \dots\}$
r_i	The record associated with individual h_i (i.e. a tuple with explicit identity information).
D	A dataset belonging to $\mathcal{I}$. We use the notation $\text{records}(\boldsymbol{\mathfrak{Data}})$ to refer to the identities and record values of individuals in the data. We use the notation $\text{tuples}(\boldsymbol{\mathfrak{Data}})$ to refer to the tuples in the database (record values without explicit reference to the identities of individuals).
$\mathbb{S}$	Set of potential secrets. Revealing s or $\neg s$ may be harmful if $s \in \mathbb{S}$.
σ_i	$r_i \in \text{records}(\boldsymbol{\mathfrak{Data}})$: The statement that the record r_i belonging to individual h_i is in the data.
$\sigma_{(i,t)}$	$r_i \in \text{records}(\boldsymbol{\mathfrak{Data}}) \wedge r_i = t$: The statement that the record r_i belonging to individual h_i has value $t \in \mathcal{T}$ and is in the data.
$\mathbb{S}_{\text{pairs}}$	Discriminative pairs. $\mathbb{S}_{\text{pairs}} \subset \mathbb{S} \times \mathbb{S}$.
$\mathbb{D}$	The set of evolution scenarios: a conservative collection of plausible data generating distributions.
θ	A probability distribution. The probability, under θ, that the data equals D_i is $P(\boldsymbol{\mathfrak{Data}} = D_i \mid \theta)$.
$\mathfrak{M}$	A privacy mechanism: a deterministic or randomized algorithm (often used in the context of a privacy definition)

true or the claim that s is false can be harmful. The role of $\mathbb{S}$ is to provide a domain for the *discriminative pairs*, a subset of $\mathbb{S} \times \mathbb{S}$, which we discuss next.

The set of discriminative pairs $\mathbb{S}_{\text{pairs}}$ is a subset of $\mathbb{S} \times \mathbb{S}$. The role of $\mathbb{S}_{\text{pairs}}$ is to tell us *how* to protect the potential secrets $\mathbb{S}$. For any discriminative pair (s_i, s_j), where $s_i \in \mathbb{S}$ and $s_j \in \mathbb{S}$, we would like to guarantee that attackers are unable to distinguish between the case where s_i is true of the actual data and the case where s_j is true of the actual data.[2] For this reason, s_i and s_j must be **mutually exclusive** but not necessarily exhaustive (it could be the case that neither is true). One example of a discriminative pair is ("Bob is in the table", "Bob is not in the table").

The discriminative pairs allow highly customizable privacy guarantees. For example, we can specify discriminative pairs such ("Bob has Cancer", "Bob has AIDS") to prevent inference about what disease Bob has (assuming only one disease is recorded). If, additionally we **avoid** specifying ("Bob is not healthy", "Bob is healthy"), then we are allowing disclosure of whether Bob is sick or healthy, but if he is sick we are not allowing inference about the specific disease. For continuous attributes we can specify discriminative pairs of the form ("Bob's salary is $x \pm 10,000$", "Bob's salary is $[x + 20,000] \pm 10,000$") for all x to say that we want to prevent inference about Bob's salary to within an absolute error of $10,000$. We can similarly use this trick for aggregates to specify that we should not allow inference about total number of sales to within $20,000$ units. Additional examples are given throughout the paper. We illustrate the particular importance of using discriminative pairs when we discuss continuous attributes in Section 8.

The evolution scenarios $\mathbb{D}$ can be viewed as a set of conservative assumptions about how the data evolved (or were generated) and about knowledge of potential at-

[2]In general, this cannot be simulated by specifying all pairs of databases (D_k, D_ℓ) where s_i is true of D_k and s_j is true of D_ℓ – the resulting privacy definition (instantiated by Definition 3.4) will often be too strong because such D_k, D_ℓ pairs can differ almost arbitrarily. For example, D_k could be a table that contains information about Bob and 0 cancer patients and D_ℓ could be a table that has no information about Bob but has $10,000$ cancer patients. Making such pairs of tables indistinguishable means that it is not possible to get accurate estimates of the number of cancer patients in a table.

tackers. Note that assumptions are absolutely necessary – privacy definitions that can provide privacy guarantees without making any assumptions provide little utility beyond the default approach of releasing nothing at all [Kifer and Machanavajjhala 2011; Dwork and Naor 2010]. Since the data curator wants to release useful information, the role of the domain expert will be to identify a reasonable set of assumptions; in many cases, they already do this informally [Sankararaman et al. 2009]. Formally, $\mathbb{D}$ is represented as a set of probability distributions over $\mathcal{I}$ (the possible database instances). Each probability distribution $\theta \in \mathbb{D}$ corresponds to an attacker that we want to protect against and represents that attacker's belief in how the data were generated (incorporating any background knowledge and side information). We illustrate the importance of specifying these distributions in Section 6 where we analyze differential privacy. Below we give some examples of possible choices of $\mathbb{D}$ and their interpretations.

Example 3.1 (*No assumptions*). $\mathbb{D}$ can consist of all possible probability distributions over database instances (i.e. including those with arbitrary correlations between records). This corresponds to making no assumptions. We explore this choice in Section 3.2.

Example 3.2 (*I.I.D. Data*). $\mathbb{D}$ can consist of all probability distributions over tables that generate records i.i.d. That is, for every f that is a distribution over $\mathcal{T}$ (domain of tuples), we have $\theta_f \in \mathbb{D}$ where the distribution θ_f is defined as: $P(\boldsymbol{\mathfrak{Data}} = \{r_1, \ldots, r_n\} \mid \theta_f) = f(r_1) \times \cdots \times f(r_n)$. Thus each attacker can have a widely different opinion about the probability a random individual has cancer, etc. These attackers, however, do not have knowledge about specific individuals (for this, see Example 3.3).

Example 3.3 (*Independent but not I.I.D.*). $\mathbb{D}$ can consist of all probability distributions over tables where records are independent but may have different distributions. That is, $\mathbb{D}$ consists of all θ for which $P(\boldsymbol{\mathfrak{Data}} = \{r_1, \ldots, r_n\} \mid \theta)$ equals $f_1(r_1) \times f_2(r_2) \times \cdots \times f_n(r_n)$ for arbitrary $f_1, f_2, \ldots, f_n$. That is, an attacker may know that the first individual is a smoker and so the corresponding record r_1 will have a different distribution than record r_2 which corresponds to an individual who is known to be a non-smoker. This is an extension of Example 3.2 where now attackers may have additional information about all individuals. Many additional variations are possible (i.e. attackers only have information about k individuals, etc.). We shall see in Section 6 the close connection between this example and differential privacy. Note that records here are still independent, so this choice of $\mathbb{D}$ may not be appropriate to social networks where correlations between records exist.

Role of the domain expert. The goal of our framework is to make assumptions explicit. Thus the domain expert needs to specify the potential secrets $\mathbb{S}$ and discriminative pairs $\mathbb{S}_{\text{pairs}}$ (i.e. what should be protected) and evolution scenarios (data assumptions) $\mathbb{D}$ – are data records independent, what correlation structures exist, are attributes independent, etc. Thus to use the Pufferfish framework, the domain expert simply does what he or she does best. Most importantly, the domain expert is no longer required to be a privacy expert.

Definition 3.4 (*Pufferfish Privacy*). Given set of potential secrets $\mathbb{S}$, a set of discriminative pairs $\mathbb{S}_{\text{pairs}}$, a set of data evolution scenarios $\mathbb{D}$, and a privacy parameter $\epsilon > 0$, a (potentially randomized) algorithm $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ privacy if

(i) for all possible outputs $\omega \in \text{range}(\mathfrak{M})$,
(ii) for all pairs $(s_i, s_j) \in \mathbb{S}_{\text{pairs}}$ of potential secrets,
(iii) for all distributions $\theta \in \mathbb{D}$ for which $P(s_i \mid \theta) \neq 0$ and $P(s_j \mid \theta) \neq 0$

the following holds[3]:

$$P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid s_i, \theta) \leq e^{\epsilon} P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid s_j, \theta) \quad (1)$$

$$P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid s_j, \theta) \leq e^{\epsilon} P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid s_i, \theta) \quad (2)$$

The probabilities in Equations 1 and 2 depend on possible randomness in $\mathfrak{M}$ and as well as the randomness in the data. Note that $P(s_i \mid \theta) \neq 0$ is a technical condition which ensures that the conditional probability $P(\cdot \mid s_i, \theta)$ is defined. Operationally, it means we should focus on attackers (and their associated θ) who still have uncertainty about s_i and s_j (i.e. $P(s_i \mid \theta) \neq 0$ and $P(s_j \mid \theta) \neq 0$)).

3.1. Semantic Guarantees

The guarantees of the Pufferfish framework are interpreted in terms of *odds* and *odds ratios*. If E_1 and E_2 are mutually exclusive events, then the *prior odds* of E_1 and E_2 is the fraction $\frac{P(E_1)}{P(E_2)}$. When the odds are equal to α, this simply means that E_1 is α times as likely as E_2. If we are given a piece of information A, it may alter the beliefs in the probabilities that E_1 or E_2 are true. For this reason we call $\frac{P(E_1 \mid A)}{P(E_2 \mid A)}$ the *posterior odds* of E_1 and E_2. If the prior odds and posterior odds are approximately equal, $\frac{P(E_1 \mid A)}{P(E_2 \mid A)} \Big/ \frac{P(E_1)}{P(E_2)} \approx 1$, then the event A did not provide information that was useful in discriminating between the case where E_1 was true or E_2 was true. The quantity $\frac{P(E_1 \mid A)}{P(E_2 \mid A)} \Big/ \frac{P(E_1)}{P(E_2)}$ is known as the *odds ratio* and reflects how much more likely event E_1 has become relative to E_2 after observing A.

In the Pufferfish framework, each probability distribution $\theta \in \mathbb{D}$ corresponds to an attacker and reflects the attacker's probabilistic beliefs and background knowledge. For all $(s_i, s_j) \in \mathbb{S}_{\text{pairs}}$, all $\theta \in \mathbb{D}$ for which $P(s_i \mid \theta) \neq 0$ and $P(s_j \mid \theta) \neq 0$, and all $\omega \in \text{range}(\mathfrak{M})$, a simple calculation shows that Equations 1 and 2 in Definition 3.4 are equivalent to the condition:

$$e^{-\epsilon} \leq \frac{P(s_i \mid \mathfrak{M}(\mathfrak{Data}) = \omega, \theta)}{P(s_j \mid \mathfrak{M}(\mathfrak{Data}) = \omega, \theta)} \Big/ \frac{P(s_i \mid \theta)}{P(s_j \mid \theta)} \leq e^{\epsilon}$$

This is the odds ratio of s_i to s_j and has the following interpretation: *if an attacker thinks s_i is α times as likely as s_j then after seeing the sanitized output the attacker will believe s_i is at most $e^{\epsilon}\alpha$ times and at least $e^{-\epsilon}\alpha$ times as likely as s_j.* In other words, for small values of ϵ, seeing the sanitized output ω provides nearly no information gain to attackers who are trying to distinguish between whether s_i or s_j is true.

Note that it is still possible to create "bad" privacy definitions using this framework. To get a bad definition in terms of privacy semantics, one should specify a single prior (in which case the privacy definition is sensitive to attackers who deviate from the prior) or omit reasonable priors (hence the need for a domain expert). Also, for some instantiations of the framework, algorithm design will be more challenging than for other instantiations due to the complex nature of Bayesian computations.

3.2. Example: Privacy with no Assumptions

As a warmup, we use Pufferfish to create a privacy definition with no assumptions (a re-interpretation of no-free-lunch privacy [Kifer and Machanavajjhala 2011]). Let $\mathcal{T}$ be the domain of tuples and let $\mathcal{H} = \{h_1, \ldots, h_N\}$ be the set of all individuals. Define σ_i to be the statement *"the record belonging to individual h_i is in the data"* and define $\sigma_{(i,t)}$

[3]In the case of continuous outputs, these conditions are interpreted in terms of the density function or the Radon-Nikodym derivative and are required to hold *almost everywhere* - the set of ω for which the conditions are violated must have probability 0.

to be the statement *"the record belonging to individual h_i is in the data and has value t"*. Define the set of potential secrets $\mathbb{S}$ and discriminative pairs $\mathbb{S}_{\text{pairs}}$ to be:

$$\mathbb{S} = \{\sigma_1, \ldots, \sigma_N\} \cup \{\sigma_{i,t} \ : \ i = 1, \ldots, N \wedge t \in \mathcal{T}\} \quad (3)$$

$$\mathbb{S}_{\text{pairs}} = \left\{(\sigma_{(i,t_a)}, \sigma_{(i,t_b)}) \ : \ i = 1, \ldots, N \wedge t_a, t_b \in \mathcal{T}\right\} \cup \{(\sigma_i, \neg\sigma_i) \ : \ i = 1, \ldots, N\} \quad (4)$$

with the interpretation that for every individual h_i we want to avoid leaking information about whether or not the record of h_i is in the data (this is specified by the discriminative pair $(\sigma_i, \neg\sigma_i)$), and if an attacker already believes h_i is in the data, we want to avoid leaking information about the value of the corresponding record (this is specified by the discriminative pairs $(\sigma_{(i,t_a)}, \sigma_{(i,t_b)})$ where t_a and t_b range over all possible tuple values).

To get privacy with no assumptions, we must make the set of evolution scenarios $\mathbb{D}$ as large as possible. That is, we set $\mathbb{D}$ to be the collection of all probability distributions over database instances (in particular, records in a database need not be independent), as in Example 3.1.

THEOREM 3.1. *Let $\epsilon > 0$, let $\mathbb{S}$ and $\mathbb{S}_{pairs}$ be specified as in Equations 3 and 4 and let $\mathbb{D}$ be the set of all possible distributions over database instances. Then an algorithm $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D})$ if and only if for **every** pair of databases D_1 and D_2 and every $\omega \in \text{range}(\mathfrak{M})$,*

$$e^{-\epsilon} P(\mathfrak{M}(D_2) = \omega) \leq P(\mathfrak{M}(D_1) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_2) = \omega)$$

For proof see Section A in the electronic appendix. Note that the randomness here only depends on $\mathfrak{M}$ because each of the D_1 and D_2 are given databases (and hence not random). From this warmup example, we see that if we make no assumptions then the goal is to prevent an attacker from distinguishing between any possible input datasets; the result is a total lack of utility. Note that this warmup shows a formal equivalence between Pufferfish with no assumptions and a strawman privacy definition called no-free-lunch privacy that was used in [Kifer and Machanavajjhala 2011] as an example of a privacy definition without utility.

4. RELATED WORK

This paper is an extension of our conference paper [Kifer and Machanavajjhala 2012]. This version of the paper includes several important additions: an analysis of the semantics of ϵ-indistinguishability [Dwork et al. 2006b] (a variant of differential privacy) in Section 6.2; a detailed critique of neighboring-table semantics for privacy definitions with respect to Bayesian semantics in Section 6.3; a new instantiation of Pufferfish that we call hedging privacy and corresponding histogram publishing algorithms in Section 7; and a discussion of conditions under which the Pufferfish framework resists the de Finetti attack [Kifer 2009] in Section 11. All of the proofs can be found in the electronic appendix.

4.1. Relation to Differential Privacy Variants

Differential privacy [Dwork et al. 2006b; Dwork 2006] represented a breakthrough in ideas and algorithms for protecting privacy. Informally, it states that the distribution of an algorithm's output is barely affected by the presence, absence, or modification of an individual's record. There are many ways of formalizing what "barely affected" means. In the case of differential privacy, it is: $P(\mathfrak{M}(D) \in S) \leq e^{\epsilon} P(\mathfrak{M}(D') \in S)$ for all S that are subsets of the range of $\mathfrak{M}$ and for all D and D' that differ in the presence/absence of a record [Dwork 2006] or that differ in the value of a record (this variation is known as ϵ-indistinguishability [Dwork et al. 2006b]). There are other ways of formalizing what

barely affected means. For example, (ϵ, δ)-indistinguishability requires $P(\mathfrak{M}(D) \in S) \leq e^{\epsilon} P(\mathfrak{M}(D') \in S) + \delta$ (e.g., [Nissim et al. 2007]). It allows $\mathfrak{M}$ to return 1 randomly chosen record when $\delta = 1/n$, where n is the number of records (thus always causing a privacy breach although the risk is spread out among individuals). It also allows $\mathfrak{M}$ to output the entire data with probability δ and run a differentially private algorithm with probability $1 - \delta$. Although δ is supposed to be cryptographically small [Nissim et al. 2007] or $O(1/n^2)$ [Ganta et al. 2008; Kasiviswanathan and Smith 2008], the possibility of an intentional privacy breach reduces the viability of this variation in real applications. Note that we are unaware of any proposed algorithms that use this power to create intentional breaches and so the algorithms that do work under this relaxed model actually satisfy a stronger privacy definition (it is not clear what this stronger definition is).

One can view the Pufferfish framework as a substantial generalization of differential privacy. Thus we explain the differences between Pufferfish and other variations of differential privacy [Duan 2009; Nissim et al. 2007; Chaudhuri and Mishra 2006; Blum et al. 2008; Kifer and Machanavajjhala 2011; Kifer and Lin 2010; Machanavajjhala et al. 2008; Mironov et al. 2009; Gehrke et al. 2011; Zhou et al. 2009; Rastogi et al. 2009].

A related framework, known as adversarial privacy [Rastogi et al. 2009] allows domain experts to plug in various data generating distributions. While there is no known equivalence between adversarial privacy and differential privacy, Rastogi et al. [2009] have proved an equivalence between a certain instantiation of adversarial privacy and ϵ-indistinguishability [Dwork et al. 2006b]. Adversarial privacy only seeks to protect the presence/absence of a tuple in the dataset, but we believe it can be extended to protecting arbitrary secrets in a manner analogous to Pufferfish. With such a generalization, the difference between our work and the work of Rastogi et al. [2009] would rest primarily in the variety of applications we discuss (e.g., Sections 6, 7, 8, 9, and 10).

Zhou et al. [2009] present two variants of differential privacy (δ-constrained and ZLW distributional) whose privacy semantics are unclear. We show approximate equivalences to instantiations of the Pufferfish framework in Section 8.3, thereby providing approximate semantics for those definitions as well.

BLR distributional privacy [Blum et al. 2008] is a definition whose goal is to protect everything in the data except for the distribution that generated the data (informally, if two databases are generated from the same distribution, the output distribution of a mechanism, with high probability, will be barely affected).

The concept of neighboring databases arose in differential privacy from the desire to protect individual records; thus two databases are neighbors if they differ on one record. In subsequent variations, the concept of indistinguishability between neighboring tables (or some generalization) plays a key role [Nissim et al. 2007; Chaudhuri and Mishra 2006; Kifer and Machanavajjhala 2011; Kifer and Lin 2010; Machanavajjhala et al. 2008; Mironov et al. 2009; Zhou et al. 2009; Gehrke et al. 2011]. One of the distinguishing approaches of the Pufferfish framework is that it does not search for "indistinguishability between the right notion of neighboring databases" (which, we feel, often obscures what it is we are trying to protect). Instead, it focuses on the actual secrets to be protected $\mathbb{S}$ and how they are to be protected via the discriminative pairs $\mathbb{S}_{\text{pairs}}$. It is important to note that pairs of secrets and pairs of databases are different entities within the Pufferfish framework – in general, pairs of secrets cannot be simulated with pairs of databases.

Ganta et al. [2008] use a Bayesian formulation of semantic security to analyze variants of differential privacy (with the proofs appearing in [Kasiviswanathan and Smith 2008]). Their definition considers whether an attacker can detect if information about

a tuple has been deleted. It follows a different philosophy than Pufferfish which considers the effect of an individual's participation in the data generating process. For example, in a social network, an individual can influence other people and such an influence would not necessarily be hidden by deleting the individual's data afterwards. Another case where the definition of Kasiviswanathan and Smith [2008] cannot be applied is when a record has been involved in a deterministic computation whose results have already been publicly released (we discuss this situation in Section 10).

Noiseless privacy [Duan 2009; Bhaskar et al. 2011] is a variation of differential privacy that is used to study which data-generating distributions provide enough entropy so that exact query answers do not leak too much information about individuals. Noiseless privacy cannot be used for histogram release for many choices of priors because a deterministic algorithm leaks significant information in cells with counts that are 0 or 1. Hedging privacy, which we present in Section 7, tries to only protect against "reasonable" priors and allows for a histogram release algorithm that can sometimes output true cell counts (and certify that they are exact).

Crowd-blending privacy [Gehrke et al. 2012] is a cryptographically inspired variant of differential privacy. It is another relaxation of differential privacy with the goal of explaining why moderately large deterministic counts can still preserve privacy (and thus has similar goals to our exploration of hedging privacy). The definition is rather complex but depends on the idea than an algorithm $\mathcal{A}$ *blends* individuals t and t' if, based on the output of $\mathcal{A}$, the attacker has difficulty in distinguishing between whether the database is $D^* \cup \{t\}$ or $D^* \cup \{t'\}$ no matter what D^* is. It treats datasets as *multisets*. An algorithm $\mathcal{A}$ with blending parameter k satisfies crowd-blending privacy if for every database D and individual $t \in D$, one of the following two conditions hold: (1) $\mathcal{A}$ blends t with $k-1$ other tuples in D (hence the requirement for multisets) or (2) an attacker cannot distinguish (based on the output of $\mathcal{A}$) between the databases D and $D \setminus \{t\}$. This privacy definition can be sensitive to background knowledge such as the size of the database. Consider the algorithm $\mathfrak{M}$ defined as

$$\mathfrak{M}(D) = \max(k, \text{number of tuples in } D \text{ not equal to "Bob"})$$

$\mathfrak{M}$ satisfies crowd-blending privacy because:

— $\mathfrak{M}$ blends any two tuples that are not equal to "Bob"
— If the number of tuples in D different from "Bob" is $\leq k$, all tuples in D satisfy the second condition of the definition
— If D contains more than k tuples and none of them are equal to "Bob" then each tuple in D blends with any choice of $k-1$ other tuples in D
— If D contains more than k tuples that differ from "Bob" but also contains some tuples that are equal to "Bob", then all tuples not equal to "Bob" satisfy the first condition (i.e. they blend with each other) and all tuples equal to "Bob" satisfy the second condition (in fact, the output of $\mathfrak{M}$ is invariant to removal of any number of such tuples).

The use of the second condition of crowd-blending privacy allows us to induce a correlation between the presence of the tuple "Bob" and the database size. If we expect such a tuple to be unique in the population and the attacker knows the database size (often this information is released because it appears harmless), then the attacker can deduce whether or not Bob's tuple is in the database. Removing the second condition, which was crucial to our attack, strengthens crowd-blending privacy to the point where it is equivalent to differential privacy [Gehrke et al. 2012]. We note that if this definition can be formulated in a Bayesian way, it could be easily strengthened by adding constraints that limit the inferences of additional attackers (such as those who know the database size).

Also related to the spirit of hedging privacy is the notion of random differential privacy [Hall et al. 2012]. In hedging privacy we try to rule out attackers whose priors are inconsistent with evidence about the data provided by a sanitizing algorithm $\mathfrak{M}$. In random differential privacy, the differential privacy constraints must hold except possibly for datasets that are unrepresentative of the unknown data-generating distribution (records that are unlikely can also be part of this exception). Thus, in the case of histogram publishing, random differential privacy can sometimes release (without noise) the cells whose counts are 0.

Zero knowledge privacy [Gehrke et al. 2011] is another cryptographically-inspired variant of differential privacy and is strictly stronger. It essentially ensures that an attacker does not learn much more from an algorithm $\mathfrak{M}$ than from a possibly randomized aggregate function of the data (from which an individual's tuple has been removed). It is designed to provide strong protections in datasets that have correlated records. Understanding how the choice of aggregate function should be guided by the type of correlation in the data is (to the best of our knowledge) an interesting open problem.

Although we consider traditional data releases here, there are other variations of differential privacy for which time plays a crucial role. This includes answering queries over streaming data [Chan et al. 2010; Dwork et al. 2010a] and providing privacy even if attackers have access to snapshots of a mechanism's internal state [Dwork et al. 2010b; Dwork et al. 2010a].

4.2. Relationship to Other Work

Other privacy frameworks also exist. A large class, of which k-anonymity [Samarati 2001] is perhaps the most well known, are known as *syntactic methods* because they are mainly concerned with the syntactic form of the outputs $\omega \in \mathrm{range}(\mathfrak{M})$ rather than the probabilities that govern the relationship between inputs to outputs: $P(\mathfrak{M}(D_i) = \omega)$. Because of this, the resulting privacy definitions tend to be less secure and are often subject to new kinds of attacks (see the surveys [Chen et al. 2009; Fung et al. 2010; Adam and Worthmann 1989] for more details). It is also often not clear what data assumptions those definitions are making.

The protection of aggregate information gets less attention than the problem of protecting individual records. The protection of aggregates is most relevant to business data where aggregates reflect different kinds of business secrets. Much of the work in this area is called *knowledge hiding* and focuses on hiding association rules, frequent itemsets, and classification rules (e.g., [Clifton and Marks 1996; Clifton 2000; Oliveira and Zaiane 2003; Wang and Lee 2008; Verykios et al. 2004b; Moustakides and Verykios 2008; Clifton et al. 2002; Aggarwal et al. 2006; Wang et al. 2005; Delis et al. 2010; Verykios et al. 2004a; Natwichai et al. 2006]) that are deemed to be sensitive. Work in this area generally follows the syntactic paradigm and it is often unclear what rigorous privacy guarantees can be provided or what data assumptions are needed for ensuring privacy.

5. PUFFERFISH AND PRIVACY AXIOMS

Research in statistical privacy has been moving away from ad-hoc privacy definitions and towards formal and rigorous privacy definitions. The reason is that rigorous privacy definitions offer the promise of ending the endless cycle of discovering a vulnerability in a privacy definition, proposing a fix, finding a vulnerability in the "fixed" version, etc. (see [Chen et al. 2009] for some examples).

To this end, recent research has started examining the properties that privacy definitions need to have [Kifer and Lin 2010; Kifer and Lin 2012]. Modern design guidelines for privacy definitions include 2 fundamental axioms known as *transformation invari-*

ance and *convexity* [Kifer and Lin 2012]. While the ideas contained in the axioms have been accepted by the privacy community for a long time, only recently has there been an insistence that privacy definitions actually satisfy them.

In this section we show that every privacy definition in the Pufferfish framework satisfies both fundamental axioms, thus ensuring that it satisfies modern design guidelines. The axioms are:

AXIOM 5.1. (Transformation Invariance [Kifer and Lin 2012]). *If an algorithm $\mathfrak{M}$ satisfies a privacy definition and $\mathcal{A}$ is any algorithm such that (1) its domain contains the range of $\mathfrak{M}$ and (2) its random bits (if any) are statistically independent from the random bits (if any) of $\mathfrak{M}$, then the algorithm $\mathcal{A} \circ \mathfrak{M}$, which first runs $\mathfrak{M}$ on the data and then runs $\mathcal{A}$ on the output should also satisfy the same privacy definition.*

The justification for the transformation invariance axiom is that $\mathcal{A}$ is an algorithm whose only input is the output of $\mathfrak{M}$ and so it simulates a data analyst who is performing a statistical analysis using the output of $\mathfrak{M}$; thus it would be strange if a privacy definition implied the output ω of $\mathfrak{M}$ was safe to release, but the results of the statistical analysis on this output ω were not (many existing privacy definitions fail to satisfy this property [Kifer and Lin 2012]).

AXIOM 5.2. (Convexity [Kifer and Lin 2012]). *If $\mathfrak{M}_1$ and $\mathfrak{M}_2$ satisfy a privacy definition, and $p \in [0, 1]$, then the algorithm $\mathfrak{M}^p$ which runs $\mathfrak{M}_1$ with probability p and $\mathfrak{M}_2$ with probability $1 - p$ should also satisfy the privacy definition.*

The convexity axiom says that a data curator is allowed to choose any algorithm $\mathfrak{M}$ that satisfies the curator's chosen privacy definition and that this choice can be randomized (thus adding further uncertainty into the creation of sanitized data). Again, many existing syntactic privacy definitions fail to satisfy this property [Kifer and Lin 2010].

The following theorem confirms that the Pufferfish framework satisfies modern privacy design guidelines.

THEOREM 5.1. *For every $\mathbb{S}$, $\mathbb{S}_{pairs}$, $\mathbb{D}$, and $\epsilon > 0$, the privacy definition ϵ-$\mathcal{P}\mathrm{uffer}\mathcal{F}\mathrm{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D})$ satisfies the axioms of convexity and transformation invariance.*

For proof see Section B in the electronic appendix.

6. PUFFERFISH ANALYSIS OF DIFFERENTIAL PRIVACY

Differential privacy [Dwork 2006] is a state of the art privacy definition which has been very influential in modern privacy research. It is formally defined as:

Definition 6.1 (*Differential Privacy [Dwork 2006]*). Given a privacy parameter $\epsilon > 0$, an algorithm $\mathfrak{M}$ satisfies ϵ-differential privacy if for all $\omega \in \mathrm{range}(\mathfrak{M})$ and all pairs of datasets D_i and D_j that differ on the presence of one tuple (i.e. D_i can be derived from D_j by either adding or deleting exactly one tuple), the following holds:

$$P(\mathfrak{M}(D_i) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_j) = \omega) \tag{5}$$

where the probability only depends on the randomness in $\mathfrak{M}$.

The main interpretation of this definition is that adding or removing Bob's tuple to the database has a small effect on the distribution of outputs of the randomized algorithm $\mathfrak{M}$. When ϵ is not too large, $\mathfrak{M}$ is likely to produce the same output regardless of whether Bob's tuple was used in the computation.

Differential privacy does not mention any prior probabilities although it is recognized that it may leak information when data records are correlated [Gehrke et al. 2011; Kifer and Machanavajjhala 2011]. The goal of this section is to state a precise relationship between differential privacy, data generating distributions, and bounds on

an attacker's inference. In particular, we show that changes in an attacker's odds (see Section 3.1) are bounded by a factor of e^ϵ precisely when data records as well as the presence/absence of individuals are independent,[4] while correlations cause additional information leakage.

We present our analysis of differential privacy in Section 6.1 and we discuss ϵ-indistinguishability (a variation of differential privacy) [Dwork et al. 2006b] in Section 6.2. In Section 6.3 we discuss why we believe that privacy semantics in terms of Bayesian inference can shed more light on a privacy definition than privacy semantics in terms of indistinguishable pairs of neighboring databases.

6.1. Analysis of Differential Privacy

Let $\mathcal{T}$ be the domain of tuples. Let $\mathcal{H} = \{h_1, h_2, \ldots, h_N\}$ be the set of all individuals in a population of size N. Define σ_i to be the statement $r_i \in \text{records}(\boldsymbol{\mathfrak{Data}})$ (i.e. *"record r_i belonging to individual h_i is in the data"*) and let $\sigma_{(i,t)}$ be the statement $r_i \in \text{records}(\boldsymbol{\mathfrak{Data}}) \wedge r_i = t$ (i.e. *"record r_i belonging to individual h_i has value t and is in the data"*). Let

$$\mathbb{S} = \left\{\sigma_{(i,t)} \ : \ h_i \in \mathcal{H},\ t \in \mathcal{T}\right\} \cup \left\{\neg\sigma_i \ : \ h_i \in \mathcal{H}\right\} \tag{6}$$

$$\mathbb{S}_{\text{pairs}} = \left\{(\sigma_{(i,t)}, \neg\sigma_i) \ : \ h_i \in \mathcal{H}, t \in \mathcal{T}\right\} \tag{7}$$

Thus for any individual h_i in the population $\mathcal{H}$ and any possible tuple value $t \in \mathcal{T}$, the goal is to prevent an attacker from distinguishing whether the record r_i belonging to h_i is in the data and has value t vs. the data has no record about individual h_i (this is our mathematical translation of the goals in [Dwork 2006]).

For the probabilistic model, suppose each individual h_i is associated with distributions π_i and f_i in the following roles:

— The probability that record r_i belonging to individual h_i is in the data is $P(r_i \in \text{records}(\boldsymbol{\mathfrak{Data}})) \equiv P(\sigma_i) = \pi_i$.
— $P(r_i = t \mid r_i \in \text{records}(\boldsymbol{\mathfrak{Data}})) \equiv P(\sigma_{(i,t)} \mid \sigma_i) = f_i(t)$

With this notation, the model is:

$$\theta \equiv \{\pi_1, \ldots, \pi_N, f_1, \ldots, f_N\} \tag{8}$$

$$P(\boldsymbol{\mathfrak{Data}} \mid \theta) = \prod_{r_i \in \text{records}(\boldsymbol{\mathfrak{Data}})} f_i(r_i)\pi_i \prod_{r_j \notin \text{records}(\boldsymbol{\mathfrak{Data}})} (1 - \pi_j)$$

In other words, the presence/absence/record-value of each individual is independent of the presence/absence/record-values of other individuals. We set $\mathbb{D}$ to be the set of all possible probability distributions of the form given in Equation 8 (i.e. for all possible choices of the π_i and f_i).

The following theorem says that under this probabilistic model, ϵ-differential privacy becomes an instantiation of the Pufferfish framework.

THEOREM 6.1. *Let $\mathbb{S}$ and $\mathbb{S}_{pairs}$ be defined as in Equations 6 and 7. Let $\mathbb{D}^*$ be the set of all distributions of the form specified in Equation 8. With these choices, ϵ-differential privacy is equivalent to ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*)$.*

See Section C of the electronic appendix for the proof.

The following theorem says that if we have any correlations between records, then some differentially private algorithms leak more information than is allowable (under

[4]Depending on the application, this would mean independence between medical records, edges in a social network, queries in a search log, etc.

the odds ratio semantics in Section 3.1), in which case an attacker's posterior beliefs may differ significantly from the prior beliefs depending on the strength of the correlation.

THEOREM 6.2. *Let $\mathbb{S}$ and $\mathbb{S}_{pairs}$ be defined as in Equations 6 and 7. Let $\mathbb{D}^*$ be the set of all distributions of the form specified in Equation 8. If we choose the data evolution scenarios $\mathbb{D}_{other}$ such that $\mathbb{D}_{other} \not\subseteq \mathbb{D}^*$ then ϵ-differential privacy is not equivalent to ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}_{other})$ (i.e. with the same ϵ-parameter) and hence does not bound the odds-ratio to the interval $[e^{-\epsilon}, e^{\epsilon}]$.*

See Section D of the electronic appendix for the proof.

6.2. Analysis of ϵ-indistinguishability

Similar results can be shown for ϵ-indistinguishability:

Definition 6.2 (ϵ-*indistinguishability [Dwork et al. 2006b]*). Given a privacy parameter $\epsilon > 0$, an algorithm $\mathfrak{M}$ satisfies ϵ-indistinguishability if for all $\omega \in \text{range}(\mathfrak{M})$ and all pairs of datasets D_i and D_j that differ on the value of one tuple (i.e. D_i can be derived from D_j by modifying one tuple), the following holds:

$$P(\mathfrak{M}(D_i) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_j) = \omega)$$

where the probability only depends on the randomness in $\mathfrak{M}$.

We note that Rastogi et al. [2009] proved an equivalence between ϵ-indistinguishability and ϵ-adversarial privacy when using a set of probability distributions called PTLM [Rastogi et al. 2009] but there is no known equivalence between ϵ-differential privacy and ϵ-adversarial privacy. However, there are connections between instantiations of the Pufferfish framework and both ϵ-indistinguishability and ϵ-differential privacy (the latter was shown in Section 6). The reason for this is the extra flexibility offered by Pufferfish in specifying what to protect. As we shall see, ϵ-indistinguishability protects something slightly different from ϵ-differential privacy.

As before, we specify the potential secrets $\mathbb{S}$ and discriminative pairs $\mathbb{S}_{\text{pairs}}$. Let $\mathcal{T}$ be the domain of tuples. Let $\mathcal{H} = \{h_1, h_2, \ldots, h_N\}$ be the set of all individuals in a population of size N. Define σ_i to be the statement $r_i \in \text{records}(\mathfrak{Data})$ (i.e. *"record r_i belonging to individual h_i is in the data"*) and let $\sigma_{(i,t)}$ be the statement $r_i \in \text{records}(\mathfrak{Data}) \wedge r_i = t$ (i.e. *"record r_i belonging to individual h_i has value t and is in the data"*). Let

$$\mathbb{S} = \left\{\sigma_{(i,t)} \ : \ h_i \in \mathcal{H},\ t \in \mathcal{T}\right\} \bigcup \left\{\neg\sigma_i \ : \ h_i \in \mathcal{H}\right\} \tag{9}$$

$$\mathbb{S}_{\text{pairs}} = \left\{(\sigma_{(i,t_a)}, \sigma_{(i,t_b)}) \ : \ {}^{i=1,\ldots,N}_{t_a \in \mathcal{T} \wedge t_b \in \mathcal{T}}\right\} \cup \left\{(\sigma_{(i,t)}, \neg\sigma_j) \ : \ {}^{i=1,\ldots,N \wedge j=1,\ldots,N}_{i \neq j \wedge t \in \mathcal{T}}\right\} \tag{10}$$

Notice that here the goal, as specified by the choice of $\mathbb{S}$ is different from ϵ-differential privacy. This follows the reasoning in [Dwork et al. 2006b] where the goal is to prevent an attacker from guessing whether an individual's record has been changed or even swapped with the record of another individual not in the data.

The probabilistic model is almost the same as with ϵ-differential privacy. The only difference is that ϵ-indistinguishability only considers datasets of the same size n. Thus we simply condition on n being the size of the data.

More concretely,

$$\theta \equiv \{\pi_1, \ldots, \pi_N, f_1, \ldots, f_N\} \tag{11}$$

$$P(\boldsymbol{\mathfrak{Data}} \mid \theta) = \begin{cases} 0 & \text{if } |\boldsymbol{\mathfrak{Data}}| \neq n \\ \left(\prod_{i \in \text{records}(\boldsymbol{\mathfrak{Data}})} f_i(r_i)\pi_i \prod_{j \notin \text{records}(\boldsymbol{\mathfrak{Data}})} (1-\pi_j) \right) / Z_n & \text{otherwise} \end{cases}$$

where Z_n is the proportionality constant that guarantees this is a probability distribution. We set $\mathbb{D}$ to be all probability distributions of the form given in Equation 11.

The proof that ϵ-indistinguishability is equivalent to this instantiation ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ is straightforward and tedious yet almost identical to the proof about differential privacy (Theorem 6.1, with proof in Section C of the electronic appendix) except with the obvious modifications caused by replacing the definition of $\mathbb{S}_{\text{pairs}}$ and conditioning on n, the size of the dataset.

6.3. Advantages of Bayesian Semantics

One of the most compelling arguments in favor of ϵ-differential privacy is that for small ϵ it guarantees that any output ω is almost as likely to occur whether or not any particular individual's data was recorded in the database [Dwork 2006]. This statement is true no matter what distribution generated the data. However, Theorems 6.1 and 6.2 imply that such a property may not always help us prevent leakage of sensitive information. In this section we provide a few more reasons why we believe new privacy definitions based on neighboring database semantics should be accompanied by Bayesian analyses.

6.3.1. Parameter settings. Kasiviswanathan et al. [Ganta et al. 2008; Kasiviswanathan and Smith 2008] studied a variant of differential privacy known as δ-approximate ϵ-indistinguishability [Dwork et al. 2006a]. This privacy definition also requires that changes in one tuple cause small changes in an algorithm's output probabilities, and the parameters ϵ and δ together control how much the output probabilities are allowed to change.[5] A Bayesian analysis [Ganta et al. 2008; Kasiviswanathan and Smith 2008] of this privacy definition showed that in order to prevent an attacker from learning too much about the value of a tuple, the parameter δ needs to be $O(1/N^2)$ (where N is the number of data records). We note that this setting of δ was not obvious from the original definition.

6.3.2. Pre-released statistics. When deterministic information about a data set is released, Kifer and Machanavajjhala [2011] noted that subsequent differentially private data releases may leak more information than is intended. They illustrated the point with an example of a histogram containing k cells $x_1, \ldots, x_k$. Suppose the following sums had already been released: $x_1 + x_2$, $x_2 + x_3$, $\ldots$, $x_{k-1} + x_k$. Note that knowledge of any cell x_i now gives us knowledge of the entire table. If we use the standard differentially private histogram release algorithm [Dwork 2006] of adding independent Laplace (with variance $2/\epsilon^2$) noise to each cell, we get a k noisy cell counts $\tilde{x}_1, \ldots, \tilde{x}_k$. However, each noisy cell count $\tilde{x}_i$ combined with the prereleased queries, gives us another noisy estimate of x_1. Averaging all k of these independent noisy estimates of x_1 provides us with a new estimate whose variance is $2/k\epsilon^2$. Neighboring table privacy semantics do not let us reason about this situation easily. However, Bayesian semantics are useful here because they give an easy way of adapting a privacy definition to account for such side information. We illustrate this point in detail in Section 10.

[5] i.e. $P(\mathfrak{M}(D_i) \in S) \leq e^{\epsilon} P(\mathfrak{M}(D_j) \in S) + \delta$ for all subsets S of the range of $\mathfrak{M}$

In particular, Kifer and Machanavajjhala [2011] proposed a privacy definition with neighboring table semantics to address the issue of pre-released deterministic query answers. On the surface, it seems like a reasonable privacy definition, but in Section 10 we show that it too can leak more information than is intended. However, the instantiation of Pufferfish we propose can fix this problem because it explicitly specifies the set of attackers who are going to have difficulty inferring sensitive information.

6.3.3. Context of an individual's choices. One important justification of neighboring table semantics resembles a non-cooperative game where an individual must choose an action without consulting other individuals. For example, suppose a data curator must release sanitized data and an individual only has the option of opting-out (that is, requesting that his or her information be deleted prior to data release). In this case differential privacy is often an appropriate privacy definition to use to ensure attackers cannot detect whether the opting out occurred or not [Dwork 2006; Kasiviswanathan and Smith 2008]. When is it more appropriate to consider a larger class of strategies such as an individual trying to convince a large group of people to opt-out simultaneously, or choosing not to participate in the data generating process at all (rather than asking for tuple deletion after the fact)? We believe that Bayesian semantics can give one such answer since they are directly concerned with inferable information (e.g., when data are correlated, deletion of a tuple does not limit inference about it since information about that tuple is carried by other records, so users may want to cooperate to limit inferences).

7. HEDGING PRIVACY: A CASE STUDY IN PUBLISHING HISTOGRAMS

In this section we consider algorithms for publishing histograms under privacy definitions that are weaker than differential privacy. For illustrative purposes, we first begin with a very weak privacy definition, which we call *single-prior privacy*, and then we strengthen it to a privacy definition we call *hedging privacy*. In single-prior privacy, the data curator specifies only one data-generating distribution. We present a histogram-publishing algorithm under this model and show that in some cases it allows partially deterministic outputs (i.e. sometimes query answers can be released with the guarantee that the answers are exact). Hedging privacy is a stronger definition. Although the data curator still specifies one distribution, the set $\mathbb{D}$ of evolution scenarios contains many other distributions. An interesting feature of this privacy definition is that it can provide utility (and even partially deterministic outputs) when the distribution specified by the data curator is an accurate model of the true data. However, it also provides privacy when the data curator's distribution is incorrect (a poor model of the data).

7.1. Single Prior Privacy

In this section, we consider datasets with n records. Define σ_i to be the statement $r_i \in \text{records}(\mathfrak{Data})$ (i.e. *"record r_i belonging to individual h_i is in the data"*) and let $\sigma_{(i,t)}$ be the statement $r_i \in \text{records}(\mathfrak{Data}) \wedge r_i = t$ (i.e. *"record r_i belonging to individual h_i has value t and is in the data"*). We can then define the potential secrets $\mathbb{S}$ and discriminative pairs $\mathbb{S}_{\text{pairs}}$ as follows:

$$\mathbb{S} = \left\{\sigma_{(i,t)} \ : \ h_i \in \mathcal{H},\ t \in \mathcal{T}\right\} \cup \left\{\neg\sigma_i \ : \ h_i \in \mathcal{H}\right\} \tag{12}$$

$$\mathbb{S}_{\text{pairs}} = \left\{(\sigma_{(i,t)}, \neg\sigma_i) \ : \ h_i \in \mathcal{H}, t \in \mathcal{T}\right\} \tag{13}$$

Let f be a fixed probability distribution over records. Assume there are N individuals $\{h_1, \ldots, h_N\}$ in the population and it is known that the data contain records from exactly n of them ($n < N$). Then the set of evolution scenarios $\mathbb{D}$ consists of the proba-

bility distributions θ having the following form (for a fixed f and arbitrary $\pi_1, \ldots, \pi_N$):

$$\theta = \{f, \pi_1, \ldots, \pi_N\}$$

$$P(\mathfrak{Data} \mid \theta) = \begin{cases} 0 & \text{if } |\mathfrak{Data}| \neq n \\ \left(\prod_{i \in \text{records}(\mathfrak{Data})} f(r_i)\pi_i \prod_{j \notin \text{records}(\mathfrak{Data})} (1-\pi_j) \right) / Z_n & \text{otherwise} \end{cases}$$

where the π_i correspond to probabilities that the corresponding individual h_i is included in the data. Z_n is a normalizing constant (i.e. the probability that the data contains information about n individuals) and depends on the π_i but not on f. Since our algorithms do not use the explicit identity information (they only operate on tuples), then inference is based on the conditional probabilities:

$$P(\text{tuples}(\mathfrak{Data}) = \{t_1, \ldots, t_n\} \mid \neg\sigma_i\ ,\ \theta) = \prod_{j=1}^{n} f(t_j)$$

$$P(\text{tuples}(\mathfrak{Data}) = \{t_1, \ldots, t_{n-1}, t^*\} \mid \sigma_{(i,t^*)},\ \theta) = \prod_{j=1}^{n-1} f(t_j)$$

With these settings, the corresponding version of Pufferfish, which we call single-prior privacy, has the following goal: make it difficult for an attacker to distinguish between the case where individual h_i is in the data and the case where the record for h_i was replaced with a random draw from the data generating distribution.

We will first look at how to answer one counting query under this privacy definition (Section 7.1.1), extend it to publishing a histogram (Section 7.1.2), and then modify the algorithm to satisfy a much stronger notion of privacy (Section 7.2).

7.1.1. Answering one counting query. In this section we consider how to answer one counting query under single-prior privacy. Note that this is equivalent to possessing a database of n bits (one bit for each individual) and asking how many bits are set to 1. Thus without loss of generality, the record values can be 0 or 1 and the data generating distribution specified by the data curator is a binomial with parameter $q \equiv P(r_j = 1)$.

Pseudo-code for the mechanism is shown in Algorithm 1. The mechanism first forms an interval $[k_{\text{lo}}, k_{\text{hi}}]$ around the expected value nq of the true answer in Lines 1 and 2. If the true answer k is unexpected (i.e. outside of this interval), then it adds to k a random variable r from the 2-sided geometric distribution [Ghosh et al. 2009] (Lines 3-5). On the other hand, if the true answer k is expected, it adds less noise than the 2-sided geometric distribution.[6] The leftover probability mass allows the mechanism to sometimes return the true answer along with the assertion that it is indeed the true answer (note that the output "true count k" can only be produced when the answer actually is k).

When k is 0 or n, if one publishes the true answer, this would result in the adversary knowing the values of all the records in the database. However, in these situations, our algorithm always outputs a noisy answer (note that the interval of expected values $[k_{\text{lo}}, k_{\text{hi}}]$ always excludes $k = 0$ and $k = n$).

[6] I.e., the probability of any noisy count is at most, and sometimes strictly less than, the probability of that noisy count under the geometric mechanism [Ghosh et al. 2009]. This fact is due to the triangle inequality, $|r-k| \leq \min\{|r-k_{\text{lo}}| + |k_{\text{lo}}-k|,\ |r-k_{\text{hi}}| + |k_{\text{hi}}-k|\}$, which can be applied to Line 8 to show that when $k \in [k_{\text{lo}}, k_{\text{hi}}]$ then $P(\text{"noisy count r"}) < \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}} e^{-|r-k|}$ (i.e. the probability under the geometric mechanism) for $r \in [k_{\text{lo}}, k_{\text{hi}}]$.

Algorithm 1 SinglePriorCountingQuery

Require: q, n: the binomial parameters, $q \in [0, 1]$
Require: k: number of bits equal to 1

1: $k_{\text{lo}} \leftarrow \max\left(1, \left\lceil n\frac{e^{-\epsilon}q}{e^{-\epsilon}q+1-q}\right\rceil\right)$
2: $k_{\text{hi}} \leftarrow \min\left(n-1, \left\lfloor n\frac{e^{\epsilon}q}{e^{\epsilon}q+1-q}\right\rfloor\right)$
3: **if** $k \notin [k_{lo}, k_{hi}]$ **then**
4: Define the distribution: $P(\text{output=“noisy count r”}) = \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}}\ \exp\left(-\epsilon|r-k|\right)$ for $r \in \mathbb{Z}$
5: **return** one sampled value from this distribution
6: **else**
7: /* i.e, when $k \in [k_{lo}, k_{hi}]$ */
8: Define the following distribution:

$$P(\text{“noisy count r”}) = \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}}\ \exp\left(-2\epsilon - \epsilon \min\left\{\begin{matrix}|r-k_{\text{lo}}|+|k_{\text{lo}}-k| \\ |r-k_{\text{hi}}|+|k_{\text{hi}}-k|\end{matrix}\right\}\right), \quad r \in [k_{\text{lo}}, k_{\text{hi}}]$$

$$P(\text{“noisy count r”}) = \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}}\ \exp\left(-\epsilon|r-k|\right), \quad r \notin [k_{\text{lo}}, k_{\text{hi}}]$$

/* the leftover probability mass: */

$$P(\text{“true count k”}) = \frac{1+e^{-\epsilon}-e^{-\epsilon(k-k_{\text{lo}}+1)}-e^{-\epsilon(k_{\text{hi}}-k+1)}}{1+e^{-\epsilon}} - \frac{e^{-\epsilon(k-k_{\text{lo}}+2)}+e^{-\epsilon(k_{\text{hi}}-k+2)}-e^{-\epsilon(k_{\text{hi}}-k_{\text{low}}+3)}-e^{-\epsilon(k_{\text{hi}}-k_{\text{low}}+2)}}{1+e^{-\epsilon}}$$

9: **return** one sampled value from this distribution
10: **end if**

THEOREM 7.1. *Define the set of potential secrets as in Equation 12 and discriminative pairs as in Equation 13. If the set of evolution scenarios $\mathbb{D}$ consists only of the binomial distribution with parameters q and n then the mechanism in Algorithm 1 satisfies ϵ-$\mathcal{P}\text{ufferF}\mathcal{I}\text{sh}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*)$.*

See Section E of the electronic appendix for the proof.

7.1.2. Histogram Publishing. In order to publish a histogram, we can re-use the mechanism from Algorithm 1 to publish noisy counts for each entry. For notational convenience, we let $\vec{q} = (q_1, \ldots, q_m)$ denote the multinomial parameter and n denote the number of records in the database and let $\mathbb{D}$ contain the single multinomial distribution with parameters n and $\vec{q}$. Let H be the true input histogram, where $H[i]$ denotes the count in cell i. Algorithm 2 shows a mechanism for producing a sanitized histogram $\tilde{H}$ while satisfying single-prior privacy.

Algorithm 2 SinglePriorSanitizedHistogram

Require: $\vec{q}, n$: the multinomial parameters. No component of $\vec{q}$ can be 0.
Require: H: the true histogram.

1: **for** $i = 1, \ldots, m$ **do**
2: $\tilde{H}[i] = SinglePriorCountQuery(q_i, n, H[i])$
3: **end for**
4: **return** $\tilde{H}$

THEOREM 7.2. *Define the set of potential secrets as in Equation 12 and discriminative pairs as in Equation 13. If the set of evolution scenarios $\mathbb{D}$ consists only of the multinomial distribution with parameters $\vec{q}$ and n then the mechanism in Algorithm 2 satisfies* 2ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*)$.

See Section F of the electronic appendix for the proof.

Note that adding 2-sided geometric noise (with probability mass function $P(r) = \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}} e^{-|r|}$ [Ghosh et al. 2009]) to every cell of a histogram when n is known achieves 2ϵ-differential privacy (since it is an output with sensitivity 2 [Dwork et al. 2006b; Dwork 2006]). As discussed in Section 7.1.1, Algorithm 2 adds less noise than this. Thus with this weaker privacy definition, we can sanitize a histogram with less noise than under differential privacy.

However, single prior privacy is a very weak privacy definition. First, because there is only one probability distribution in $\mathbb{D}$, there are few guarantees about what happens when this distribution is not an accurate model of reality or when there are other plausible data-generating distributions. Second, since the data publisher specifies no uncertainty about this distribution, it makes more sense for the data publisher to just release this distribution instead of the data. We address these concerns with a more powerful privacy definition in Section 7.2.

7.2. Hedging Privacy

Hedging privacy is a strengthening of single-prior privacy that protects data publishers whose prior beliefs about the data are poor models of the data, yet allows partially deterministic outputs (i.e. the algorithm can sometimes output the true answer and certify that it is the true answer) when the prior beliefs are a good model of the data.

We use the same potential secrets and discriminative pairs as before.

$$\mathbb{S} = \left\{\sigma_{(i,t)} \ : \ h_i \in \mathcal{H},\ t \in \mathcal{T}\right\} \cup \left\{\neg\sigma_i \ : \ h_i \in \mathcal{H}\right\} \tag{14}$$

$$\mathbb{S}_{\text{pairs}} = \left\{(\sigma_{(i,t)}, \neg\sigma_i) \ : \ h_i \in \mathcal{H}, t \in \mathcal{T}\right\} \tag{15}$$

However, the specification of evolution scenarios is different in two ways. First, notice that Definition 3.4 only uses the conditional probabilities (conditioned on potential secrets). Thus, we will use a slight generalization where the evolution scenarios are directly given in terms of conditional probabilities. In this case, the evolution scenarios describe how the data would evolve if $\sigma_{(i,t)}$ or $\neg\sigma_i$ were true.

The second change is that the data publisher specifies a prior f^* and a small positive number α (say $\alpha = 0.001$). The set of evolution scenarios is then all conditional distributions θ that are mixtures between f^* and arbitrary distributions f. That is, for fixed f^*, fixed α and arbitrary f define $\mathbb{D}$ to contain all conditional distributions θ of the form:

$$\begin{aligned}
\theta &= \{f, f^*, \alpha\} \\
P(\text{tuples}(\boldsymbol{\mathfrak{Data}}) = \{t_1, \ldots, t_n\} \mid \neg\sigma_i, \theta) &= \left[(1-\alpha)\prod_{j=1}^{n} f(t_j) + \alpha \prod_{j=1}^{n} f^*(t_j)\right] \\
P(\text{tuples}(\boldsymbol{\mathfrak{Data}}) = \{t_1, \ldots, t_{n-1}, t^*\} \mid \sigma_{(i,t^*)}, \theta) &= \left[(1-\alpha)\prod_{j=1}^{n-1} f(t_j) + \alpha \prod_{j=1}^{n-1} f^*(t_j)\right]
\end{aligned}$$

The motivations for these changes are the following. First, when an attacker reasons about a specific individual h_i, the attacker may have background knowledge about h_i so that the record belonging to h_i can come from some distribution can be unrelated

to f and f^* (or the record may have even been altered arbitrarily). By specifying conditional probabilities (conditioned on the potential secret), we don't have to explicitly enumerate all of those possible distributions.[7]

The motivation for the mixture distribution is the following. The attackers are allowed to believe that the records are generated i.i.d. from some arbitrary distribution f (unknown to the data publisher). However, the evolution scenarios we use are a mixture (with parameter α) of the probability of the dataset under f and the probability of the dataset under f^* (specified by the data publisher). The rationale behind this change is the following. When f is at least as good as f^* in modeling the data, the probabilities under f and under the mixture are almost the same (since α is small). In such a case, providing privacy protections against this attacker and against this mixture distribution are concordant goals.

On the other hand, when f is worse than f^* in modeling the data, then f is not a reasonable model and inferences made by an attacker who believes in f are not plausible. In such cases probabilities under f and the probabilities under the mixture distribution may give different results but this is acceptable because there is no need to defend against this attacker. [8] The more implausible the attacker's beliefs relative to the data publisher's beliefs, the more the mixture distribution behaves like the probability under f^*. Although we provide privacy protections against this mixture distribution, we only need the protections that arise when the mixture closely models the attacker's belief probabilities.

Another way to think about this is that a rational attacker cannot completely discount f^* and so considers it possible (with a small probability α).

The mechanisms for answering a single count query and publishing a histogram under hedging privacy are shown in Algorithm 3 and 4, respectively. Note that the only difference with single prior privacy is the computation of k_{hi} and k_{lo} but now we have stronger privacy protections. Thus sometimes the algorithm is allowed to output exact counts and to certify that those counts are exact.

THEOREM 7.3. *Algorithm 3 satisfies ϵ-hedging privacy and Algorithm 4 satisfies 2ϵ-hedging privacy.*

For proof see Section G of the electronic appendix.

It is important to note two properties of Algorithms 3 and 4. First, the interval $[k_{\text{lo}}, k_{\text{hi}}]$ can be empty for small n (so that only noisy counts are produced). The reason is that n needs to be large enough to provide evidence that f^* is really superior to distributions f that are far enough away from it. For example, in the case of Algorithm 3, n needs to be large enough to identify the counts k whose probability is modeled by q^* much better than distributions outside the interval $[q_{\text{lo}}, q_{\text{hi}}]$. The proof of Theorem 7.3 shows that $[k_{\text{lo}}, k_{\text{hi}}]$ is nonempty when n is large enough (essentially n has to be large compared with the additive term containing $\log \lambda$, in Line 4 in Algorithm 3).

The second important property of Algorithms 3 and 4 is that an algorithm that uses the mixture parameter α also works for hedging privacy definitions that use larger mixture parameters $\alpha' \in (\alpha, 1/2)$ meaning that if the data publisher's pre-specified distribution f^* is deemed even more likely a priori, the same algorithm will still work. That condition that $\alpha < 1/2$ ensures that the attacker's distribution gets more weight than f^* (i.e. the whole point of hedging privacy is to get the data, not prior knowledge, to overrule an attacker's distribution f).

---

[7] As a consequence, this means that each θ is specific to a discriminative pair $(\sigma_{(i,t)}, \neg\sigma_i)$ so that we can make statements about an attacker with background knowledge about h_i making inferences about h_i.

[8] We still protect against attackers with more plausible beliefs because we consider all possible distributions f

Algorithm 3 HedgingPrivacyCountingQuery

Require: q^*, n: the binomial parameters, $q^* \in (0,1)$
Require: α: a small number $< 1/2$, the hedging parameter
Require: k: number of bits equal to 1

$q_{\text{lo}} \leftarrow \frac{e^{-\epsilon/3}q^*}{e^{-\epsilon/3}q^*+1-q^*}$
$q_{\text{hi}} \leftarrow \frac{e^{\epsilon/3}q^*}{e^{\epsilon/3}q^*+1-q^*}$
$\lambda \leftarrow \frac{1-\alpha}{\alpha(e^{\epsilon/3}-1)}$
$k_{\text{lo}} = 1 + \left\lceil (n-1)\frac{\log\frac{1-q_{\text{lo}}}{1-q^*}}{\log\left(\frac{q^*}{1-q^*}/\frac{q_{\text{lo}}}{1-q_{\text{lo}}}\right)} + \frac{\log\lambda}{\log\left(\frac{q^*}{1-q^*}/\frac{q_{\text{lo}}}{1-q_{\text{lo}}}\right)} \right\rceil$
$k_{\text{hi}} = \min\left(n-1, \left\lfloor (n-1)\frac{\log\frac{1-q^*}{1-q_{\text{hi}}}}{\log\left(\frac{q_{\text{hi}}}{1-q_{\text{hi}}}/\frac{q^*}{1-q^*}\right)} - \frac{\log\lambda}{\log\left(\frac{q_{\text{hi}}}{1-q_{\text{hi}}}/\frac{q^*}{1-q^*}\right)} \right\rfloor\right)$
if $k \notin [k_{lo}, k_{hi}]$ **then**
  Define the distribution: $P(\text{output=“noisy count r”}) = \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}}\ \exp\left(-\epsilon|r-k|\right)$ for $r \in \mathbb{Z}$
  return one sampled value from this distribution
else
  /* i.e, when $k \in [k_{lo}, k_{hi}]$ */
  Define the following distribution:

$$P(\text{“noisy count r”}) = \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}}\ \exp\left(-2\epsilon - \epsilon\min\left\{\begin{matrix}|r-k_{\text{lo}}|+|k_{\text{lo}}-k| \\ |r-k_{\text{hi}}|+|k_{\text{hi}}-k|\end{matrix}\right\}\right), \quad r \in [k_{\text{lo}}, k_{\text{hi}}]$$

$$P(\text{“noisy count r”}) = \frac{1-e^{-\epsilon}}{1+e^{-\epsilon}}\ \exp\left(-\epsilon|r-k|\right), \quad r \notin [k_{\text{lo}}, k_{\text{hi}}]$$

  /* the leftover probability mass: */

$$P(\text{“true count k”}) = \frac{1+e^{-\epsilon}-e^{-\epsilon(k-k_{\text{lo}}+1)}-e^{-\epsilon(k_{\text{hi}}-k+1)}}{1+e^{-\epsilon}} - \frac{e^{-\epsilon(k-k_{\text{lo}}+2)}+e^{-\epsilon(k_{\text{hi}}-k+2)}-e^{-\epsilon(k_{\text{hi}}-k_{\text{low}}+3)}-e^{-\epsilon(k_{\text{hi}}-k_{\text{low}}+2)}}{1+e^{-\epsilon}}$$

  return one sampled value from this distribution
end if

Algorithm 4 HedgingPrivacySanitizedHistogram

Require: $\vec{q}, n$: the multinomial parameters. No component of $\vec{q}$ can be 0.
Require: α: a small number $< 1/2$, the hedging parameter
Require: H: the true histogram.

for $i = 1, \ldots, m$ **do**
  $\tilde{H}[i] = HedgingPrivacyCountQuery(q_i, n, H[i])$
end for
return $\tilde{H}$

We emphasize that Algorithms 3 and 4 should be seen as a proof of concept – a demonstration that sometimes no noise may be added while still protecting privacy against a reasonable range of attackers. We believe both algorithms can be improved in practical applications. This is an interesting direction for future work.

8. CONTINUOUS ATTRIBUTES AND AGGREGATE SECRETS

One of the difficult problems in privacy-preserving data publishing is protecting the values of continuous variables that are not a priori bounded or which are bounded but can take very large values (such as income). For example, many algorithms for differential privacy do not work in the first case (i.e. no a priori bound) [Dwork et al. 2006b] and provide poor utility in the second case.

In this section we use the Pufferfish framework to provide a solution to this problem (Section 8.1). This application shows the importance of specifying discriminative pairs $\mathbb{S}_{\text{pairs}}$. We then show how this solution can be used to protect aggregate secrets (Section 8.2). Finally, we use Pufferfish to provide approximate privacy semantics for δ-constrained α-differential privacy [Zhou et al. 2009] and ZLW distributional privacy [Zhou et al. 2009] (Section 8.3); those two definitions were also designed for continuous attributes but their semantic guarantees and conditions under which those guarantees hold were not clear.

8.1. Protecting Continuous Attributes

As we saw in Section 6, differential privacy is designed to make it difficult to distinguish between the case when an individual's record was included in the data with value t or whether it was not in the data at all. This has to be true whether $t = 1$ or $t = 1,000,000$ (e.g., in the case where the tuple domain $\mathcal{T} = [0, 10^6]$). To account for the possibility that the record of one individual could dominate an aggregate statistic such as $SUM(t)$, an algorithm such as the Laplace mechanism [Dwork 2006] needs to add noise with standard deviation proportional to 10^6 in order to satisfy differential privacy (thus potentially masking out the signal in the data).

If this loss of utility is unacceptable, the data curator may want to relax privacy by stating requirements such as (1) an attacker should not be able to infer any individual's salary to within an absolute error of less than $1,000$, or (2) an attacker should not be able to infer any individual's salary to within a relative error of 10%. Both of these requirements can be handled in the Pufferfish framework.

8.1.1. Privacy via Absolute Error. For ease of explanation, suppose that records belonging to individuals $h_1, \ldots, h_n$ are known to be in the data and suppose the data curator only collects an individual's income so that the domain of tuples is $\mathbb{R}^+$, the set of non-negative real numbers (hence the domain is an unbounded set). If we are interested in preventing inference about income that is within absolute error k, then we can proceed as follows. Let $\sigma_{i,[x-k,x+k)}$ be the statement "the income of individual h_i is in the range $[x-k, x+k)$". Define

$$\mathbb{S} = \left\{\sigma_{i,[x-k,x+k)} \ : \ i = 1, \ldots, n, \ x \in [0, \infty)\right\} \tag{16}$$

We set discriminative pairs to be neighboring intervals (note that the intervals are half-open so that each discriminative pair consists of mutually exclusive statements). With this setting, we are requiring that the attackers should have difficulty in distinguishing between whether someone's income is between $y-k$ and $y+k$ or a neighboring interval $[y+k, y+3k)$ or $[y-3k, y-k)$ thus ensuring that accurate inference to within $\pm k$ is not possible. Formally,

$$\mathbb{S}_{\text{pairs}} = \left\{\left(\sigma_{i,[x-k,x+k)}, \sigma_{i,[x+k,x+3k)}\right) \ : \ \begin{smallmatrix} i=1,\ldots,n \\ x\in[0,\infty) \end{smallmatrix}\right\} \tag{17}$$

We can set the evolution scenarios $\mathbb{D}$ to be the set of all probability distributions that assign incomes to individuals $h_1, \ldots, h_n$ independently. Thus the model is:

$$\begin{aligned} \theta &\equiv [f_1, \ldots, f_n] \\ P(\mathfrak{Data} \mid \theta) &= \prod_{r_i \in \text{records}(\mathfrak{Data})} f_i(r_i) \end{aligned} \tag{18}$$

where the interpretation of the probability (as a probability mass function or density function) depends on whether the f_i are continuous or not. We set $\mathbb{D}$ to be all probability distributions of the form given in Equation 18 (i.e. for all choices of $f_1, \ldots, f_n$).

With the resulting instantiation ϵ-$\mathcal{P}\text{ufferF}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$, the following lemma shows that we can answer the query "what is the sum of the salaries" as follows. Let $t_1, \ldots, t_n$ be the tuple values. Compute $X + \sum_{i=1}^n t_i$ where X is a random variable drawn from the Laplace$(4k/\epsilon)$ distribution with density function $\frac{\epsilon}{8k} e^{-\epsilon|x|/4k}$. Note that when the data set size n is large, the true average salary $\frac{\sum_{i=1}^n t_i}{n}$ and the noisy average (i.e. this noisy sum divided by n) are very close to each other with high probability. In contrast, satisfying differential privacy by adding noise to the sum would require a distribution with infinite variance (i.e. not possible) since there is no upper bound on tuple values; thus the relaxation created using Pufferfish allows more utility while clearly describing the privacy lost (i.e. income is inferable to an absolute error of $\geq k$ but to no smaller range).

LEMMA 8.1. *With $\mathbb{S}$ and $\mathbb{S}_{pairs}$ defined in Equations 16 and 17 let $\mathbb{D}$ be the set of all probability distributions having the form specified in Equation 18. The algorithm $\mathfrak{M}$ which returns $X + \sum_{i=1}^n t_i$ where X has density $\frac{\epsilon}{8k} e^{-\epsilon|x|/4k}$ satisfies ϵ-$\mathcal{P}\text{ufferF}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D})$.*

For proof see Section H of the electronic appendix.

8.1.2. Privacy via Relative Error. We can extend these ideas to protect against inference to within a prespecified *relative error* as well. One way to approach this problem is to choose $c \in (0, 1)$ and define $\sigma_{i,[cy,y/c)}$ to be the statement that "the income of individual h_i is in the range $[cy, y/c)$". Thus, for example, to express inference to within 10% we set $c = 0.1$. We define the set of potential secrets to be:

$$\mathbb{S} = \left\{\sigma_{i,[cy,y/c)} \ : \ i = 1, \ldots, n, \ y > 0\right\}$$

The discriminative pairs are again neighboring intervals. With this setting, we are requiring that the attackers should have difficulty in distinguishing between whether someone's income is in the interval $[cy, y/c)$ – whose center in terms of geometric mean is y – or a neighboring interval $[y/c, y/c^3)$ or $[cy, c^3 y)$, thus limiting the attacker's ability to infer the income to within a factor of c. Formally,

$$\mathbb{S}_{\text{pairs}} = \left\{(\sigma_{i,[cy,y/c)}, \sigma_{i,[y/c,y/c^3)}) \ : \ i = 1, \ldots, n, \ y > 0\right\}$$

As in the case of absolute error, we can set the evolution scenarios $\mathbb{D}$ to be the set of all data-generating distributions that generate records independently (but not necessarily i.i.d.):

$$\begin{aligned} \theta &\equiv [f_1, \ldots, f_n] \\ P(\mathfrak{Data} \mid \theta) &= \prod_{r_i \in \text{records}(\mathfrak{Data})} f_i(r_i) \end{aligned} \tag{19}$$

Now note that $t \in [cy, y/c)$ if and only if $\log t \in [y + \log c, y - \log c)$ and so protecting y for relative error becomes the same as protecting $\log y$ for absolute error $\pm \log c$. Thus

this version of relative error is reduced to the case of absolute error, and so we can protect tuples by applying additive noise to the logarithm, or, equivalently by using multiplicative noise.

8.2. Aggregate Secrets

In this section we discuss how Pufferfish can be used to protect secrets that are aggregate properties over the data. A typical use case for protecting aggregates is sharing of business data. For instance, a shipping company may want to publish data collected from their vehicles as it might reveal useful information about traffic patterns and energy usage. However, the company may not want business secrets like number of vehicles owned, or the geographical distribution of vehicle routes to be disclosed. We present a brief discussion of a specific scenario of protecting an aggregate secret using a simple extension of the ideas in Sections 8.1.1 and 8.1.2. Though brief, the following discussion is necessary because there is little focus in the literature on rigorous and formal privacy guarantees for business data.[9] We defer solving the challenging problem of developing Pufferfish instantiations and algorithms for general aggregate secrets to future work.

In some cases a business may have a large dataset $\mathfrak{Data} = \{r_1, \ldots, r_n\}$ that can be considered to be an i.i.d. sample from some distribution with sufficient statistics Θ. The business may decide that letting the public learn about Θ too precisely is unacceptable. The business may reason as follows: if an attacker had a large dataset generated independently from the same distribution, the attacker may use it to make inferences about the business's own data. For instance, if r_i's are real values and the adversary learns that they are drawn from a distribution with mean θ, then the adversary can use the gained knowledge about the true distribution to make inferences about the realized sample sum (in the business data) $\sum_{i=1}^{n} r_i$ up to an additive error of $O(\sqrt{n})$ with high probability (i.e. sampling error). Thus the business may consider it acceptable to create a data release as long as one cannot infer certain sums to within $\pm\sqrt{n}$.

We instantiate a Pufferfish instantiation for protecting aggregate secrets as follows. We would like to ensure that an adversary can't determine whether the value of an aggregate Φ is close to some value $x = \Phi(D)$, for some dataset $D \in \mathcal{I}$. We capture this using a metric $\mu(\mathfrak{Data}, D)$ which quantifies the distance between the value of the aggregate on the unknown database $\mathfrak{Data}$ and an arbitrarily chosen database D. For instance, suppose each $r_i = (x_i, y_i) \in D$ denotes the foreign and domestic sales of an item, and $\Phi(D) = (x(D), y(D)) = (\sum_{i \in D} x_i, \sum_{i \in D} y_i)$ denotes the total foreign and domestic sales. Then, $\mu(\mathfrak{Data}, D)$ could denote the L_p norm $(|x(\mathfrak{Data}) - x(D)|^p + |y(\mathfrak{Data}) - y(D)|^p)^{\frac{1}{p}}$.

We define $\sigma_{[\mu,D,\delta]}$ to be the statement that $\mu(\mathfrak{Data}, D) \leq \delta$ and $\sigma^*_{[\mu,D,\delta]}$ to be the statement that $2\delta \geq \mu(\mathfrak{Data}, D) > \delta$. The set of potential secrets and discriminative pairs could then be defined as:

$$\mathbb{S}_\delta = \left\{\sigma_{[\mu,D,\delta]} \ : \ D \in \mathcal{I}\right\} \cup \left\{\sigma^*_{[\mu,D,\delta]} \ : \ D \in \mathcal{I}\right\} \tag{20}$$

$$\mathbb{S}_{\text{pairs}_\delta} = \left\{(\sigma_{[\mu,D,\delta]}, \sigma^*_{[\mu,D,\delta]}) \mid D \in \mathcal{I}\right\} \tag{21}$$

where δ could be set to $O(\sqrt{n})$ and the data evolution scenarios can be the set of all distributions over $\mathcal{I}$ in which records are generated i.i.d. or just independently. Thus, by analogy to the case of continuous attributes, the attacker would not be able to dis-

[9] Even nonrigorous approaches are rare for business data.

tinguish between the cases where the value of the true aggregates are within a δ-ball centered around an arbitrary value or a within δ-band outside of that ball.

Note that one can union the potential secrets in Equations 20 and 16 and union the discriminative pairs in Equations 21 and 17 to protect both aggregate information and secrets about individuals.

8.3. δ-Constrained and Distributional Privacy

Zhou et al. [2009] also proposed two privacy definitions that can be used with continuous variables. Those definitions were introduced solely for the study of utility and their precise privacy semantics (i.e. what inferences do they protect against) were not explained nor explored. They are phrased in terms of databases that should be indistinguishable. We show an approximate equivalence between those privacy definitions and instantiations of Pufferfish, so that the Pufferfish framework (approximately) subsumes those definitions and gives them clear Bayesian privacy semantics. We start with those definitions:

Definition 8.2. *(Constrained and ZLW-Distributional Privacy [Zhou et al. 2009])* Let μ be a metric on databases and let $\delta > 0$ and $\epsilon > 0$ be constants. Let $\mathfrak{M}$ be an algorithm. For all $\omega \in \text{range}(\mathfrak{M})$, if the following constraints hold

$$e^{-\epsilon} P(\mathfrak{M}(D_2) = \omega) \leq P(\mathfrak{M}(D_1) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_2) = \omega)$$

— whenever D_1 and D_2 differ on the value of 1 tuple and $\mu(D_1, D_2) \leq \delta$ then algorithm $\mathfrak{M}$ satisfies δ-constrained ϵ-differential privacy.

— If those conditions hold when (1) D_1 and D_2 belong to a prespecified countable subset $S \subset \mathcal{I}$ and (2) $\mu(D_1, D_2) < \delta$ and (3) $D_1 \cap D_2 \neq 0$ then algorithm $\mathfrak{M}$ satisfies ZLW (ϵ, δ)-distributional privacy[10].

First, note that δ-constrained ϵ-differential privacy with metric μ is equal to ZLW (ϵ, δ^*)-distributional privacy with a properly chosen metric μ^* that combines Hamming distance with the metric μ, an appropriate choice of δ^*, and setting $S = \mathcal{I}$. Thus, we can focus only on ZLW distributional privacy. Second, the condition $D_1 \cap D_2 = \emptyset$ is also not necessary since it can also be achieved by proper choice of metric (we therefore drop this condition to increase generality). Third, it is not clear what (if any) privacy semantics are produced by the condition $D_1, D_2 \in S$. Thus we remove this condition (i.e. set $S = \mathcal{I}$) and use Pufferfish to provide approximate semantics to the resulting definition:

Definition 8.3. *(Modified ZLW-Privacy).* Let μ be a metric and let $\delta > 0$ and $\epsilon > 0$ be constants. An algorithm $\mathfrak{M}$ satisfies (ϵ, δ)-modified ZLW privacy if for every $\omega \in \text{range}(\mathfrak{M})$ and every pair of databases D_1, D_2 such that $\mu(D_1, D_2) \leq \delta$, the following conditions hold:

$$e^{-\epsilon} P(\mathfrak{M}(D_2) = \omega) \leq P(\mathfrak{M}(D_1) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_2) = \omega)$$

The approximate equivalence (subject to a mild condition on μ) to a Pufferfish instantiation is the following:

THEOREM 8.1. *Let μ be a metric over database instances such that whenever $\mu(D_1, D_2) \leq \delta$ there exists*[11] *a $D^* \in \mathcal{I}$ with $\mu(D_1, D^*) \leq \delta$ and $\mu(D_2, D^*) > \delta$. Let $\epsilon > 0$ and $\delta > 0$. Set $\mathbb{S}_\delta$ as in Equation 20 and $\mathbb{S}_{pairs_\delta}$ as in Equation 21. Define $\mathbb{D}$ to*

[10] We use the prefix ZLW to distinguish it from the distributional privacy definition introduced in [Blum et al. 2008].

[11] This condition is achieved, for example, by the L_1 norm, L_2 norm, etc. as long as δ is less than the radius of $\mathcal{I}$.

be the set of distributions over dataset instances with n records where record values are independent (e.g., all distributions of the form given in Equation 19). If $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}_\delta, \mathbb{S}_{pairs_\delta}, \mathbb{D})$ then it also satisfies (ϵ, δ)-modified ZLW privacy; conversely, if $\mathfrak{M}$ satisfies (ϵ, δ)-modified ZLW privacy then it satisfies the definition 4ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}_\delta, \mathbb{S}_{pairs_\delta}, \mathbb{D})$ (i.e. up to a four-fold degradation of semantic guarantees in terms of odds-ratio).

For proof, see Section I of the electronic appendix. Thus although the precise privacy semantics of the ZLW privacy definitions [Zhou et al. 2009] are unknown, the ideas discussed in Sections 8.1.1, 8.1.2, and 8.2 combined with Theorem 8.1 show that the Pufferfish framework can give the ZLW privacy definitions some approximate semantics in terms of an odds-ratio bound of $4e^\epsilon$ when protecting secrets about a database to within absolute error (as defined by the metric μ) of $\pm\delta$.

9. COMPOSITION

Given a privacy definition, the notion of *composition* [Ganta et al. 2008] refers to the degradation of privacy due to two independent data releases. For example, Alice may choose two algorithms $\mathfrak{M}_1$ and $\mathfrak{M}_2$ (whose random bits are independent of each other), run both on the same data and publish both results. Alternatively, Alice and Bob may have overlapping datasets; they can each run an algorithm $\mathfrak{M}_{Alice}$ and $\mathfrak{M}_{Bob}$ (possibly satisfying different privacy definitions) on their own dataset and output the result. It is known that privacy can degrade in those instances. For example, two independent releases of k-anonymous tables can lead to a privacy breach [Ganta et al. 2008]. On the other hand, differential privacy composes well with itself: if Alice uses an ϵ_1-differentially private algorithm and Bob uses an ϵ_2-differentially private algorithm, the result (from an attacker's perspective) is the same as if Alice and Bob pooled their data and used an $(\epsilon_1 + \epsilon_2)$-differentially private algorithm. More formally,

THEOREM 9.1. (Composition of Differential Privacy [McSherry 2009; Ganta et al. 2008]). *Let $\mathfrak{Data}$ denote the unknown input database, and let $\mathfrak{M}_i(\cdot)$, $\mathfrak{M}_i(\cdot, \cdot)$ be randomized algorithms which satisfy ϵ_i-differential privacy. Then we have:*

(*1*) Serial Composition: *An algorithm $\mathfrak{M}(\mathfrak{Data})$ that outputs both $\omega_1 = \mathfrak{M}_1(\mathfrak{Data})$ and $\omega_2 = \mathfrak{M}_2(\mathfrak{Data}, \omega_1)$ satisfies $(\epsilon_1 + \epsilon_2)$-differential privacy.*
(*2*) Parallel Composition: *For datasets $\mathfrak{Data}_1 \cap \mathfrak{Data}_2 = \emptyset$, an algorithm $\mathfrak{M}(\mathfrak{Data}_1, \mathfrak{Data}_2)$ that outputs $\mathfrak{M}_1(\mathfrak{Data}_1)$ and $\mathfrak{M}_2(\mathfrak{Data}_2)$ satisfies $(\max\{\epsilon_1, \epsilon_2\})$-differential privacy.*
(*3*) Post Processing*: For any algorithm $\mathfrak{M}$, outputting $\mathfrak{M}(\mathfrak{M}_1(\mathfrak{Data}))$ satisfies ϵ-differential privacy.*

However, a differentially private release may not compose nicely with other data release mechanisms [Kifer and Machanavajjhala 2011] and so it is important to study composition between algorithms within the same privacy definition and also to study composition between algorithms from different privacy definitions.

Properly handling multiple data releases requires a mixture of policy and technological solutions. If Alice is planning multiple data releases, Alice needs to account for information leaked by her prior data releases. If Alice and Bob are planning to release sanitized data, they need to co-ordinate their efforts if their raw data are correlated. That is, they need to agree on a technological solution that guarantees that the combination of their data releases will not breach privacy.

9.1. Pufferfish View of Composition

Since the Pufferfish framework provides a wide variety of privacy definitions, it allows us to study composition more generally. The key point is that, as before, we need to specify probabilistic assumptions about how datasets are related.

Suppose Alice has a dataset $\mathfrak{Data}_{Alice}$ and Bob has a dataset $\mathfrak{Data}_{Bob}$. Alice announces that she will use a privacy definition $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ to publish a sanitized version of her data and Bob announces that he will use a privacy definition $\mathfrak{Priv}$ (which need not belong to the Pufferfish framework). Alice would like to examine the consequences to privacy that can occur when they both release sanitized data using their chosen privacy definitions.

In order for Alice to study how her privacy definition composes with possible data releases from Bob, she needs to consider *all* the different plausible relationships between her dataset and Bob's dataset. Thus she also needs to specify a set $\mathcal{C}\text{ond}$ of conditional probability distributions of Bob's data given her own.[12] Each $\phi \in \mathcal{C}\text{ond}$ specifies a conditional probability distribution $P(\mathfrak{Data}_{\text{Bob}} = D' \mid \mathfrak{Data}_{\text{Alice}} = D, \phi)$. The distributions $\phi \in \mathcal{C}\text{ond}$ and $\theta \in \mathbb{D}$ combine to form a joint distribution:

$$P(\mathfrak{Data}_{\text{Bob}} \wedge \mathfrak{Data}_{\text{Alice}} \mid \phi, \theta) \;=\; P(\mathfrak{Data}_{\text{Bob}} \mid \mathfrak{Data}_{\text{Alice}}, \phi) P(\mathfrak{Data}_{\text{Alice}} \mid \theta)$$

Alice can then reason as follows. For the moment, suppose Bob has already chosen to apply an algorithm $\mathcal{A}$ to his data. Alice can study the effect of releasing the output of $\mathfrak{M}(\mathfrak{Data}_{\text{Alice}})$ by considering the distributions in $\theta \in \mathbb{D}$ and $\phi \in \mathcal{C}\text{ond}$. Simulating an adversary's reasoning, for each $(s_i, s_j) \in \mathbb{S}_{\text{pairs}}$, $\omega \in \text{range}(\mathfrak{M})$, $\omega^* \in \text{range}(\mathcal{A})$, she derives:

$$P\Big(\mathcal{A}(\mathfrak{Data}_{\text{Bob}}) = \omega^* \wedge \mathfrak{M}(\mathfrak{Data}_{\text{Alice}}) = \omega \mid s_i, \phi, \theta\Big) = \int \left(\begin{smallmatrix} P(\mathfrak{M}(D)=\omega) P(\mathfrak{Data}_{Alice}=D) \mid s_i, \theta) \\ \times E[P(\mathcal{A}(\mathfrak{Data}_{\text{Bob}})=\omega^*) \mid \mathfrak{Data}_{Alice}=D, \phi] \end{smallmatrix}\right) \, dD \tag{22}$$

where $E\left[P(\mathcal{A}(\mathfrak{Data}_{Bob}) = \omega^*) \mid \mathfrak{Data}_{\text{Alice}} = D, \phi\right]$

$$= \int P\left(\mathcal{A}(D') = \omega^*\right) P\left(\mathfrak{Data}_{\text{Bob}} = D' \middle| \, \mathfrak{Data}_{\text{Alice}} = D, \phi\right) dD'$$

is the averaged conditional probability (using distribution $\phi \in \mathcal{C}\text{ond}$) of seeing Bob's sanitized output ω^* given that Alice's dataset is D.

The significance of Equation 22 is that an attacker who uses the distributions $\theta \in \mathbb{D}$ and $\phi \in \mathcal{C}\text{ond}$ to reason about the joint data release would reason in the exact same way in an alternate universe in which Bob releases nothing and Alice releases information about her dataset using an algorithm $\mathfrak{M}^{\star}_{\phi,\mathcal{A},\mathfrak{M}}$ with $\text{range}(\mathfrak{M}) \times \text{range}(\mathcal{A})$ and which outputs the pair (ω, ω^*) with probability $P[\mathfrak{M}^{\star}_{\phi,\mathcal{A},\mathfrak{M}}(D) = (\omega, \omega^*)] = P(\mathfrak{M}(D) = \omega) E\big[P(\mathcal{A}(\mathfrak{Data}_{Bob}) = \omega^*) \big| \, \mathfrak{Data}_{\text{Alice}} = D, \phi\big]$. Thus to study the privacy properties of this joint data release, Alice only needs to study the algorithms of the form $\mathfrak{M}^{\star}_{\phi,\mathcal{A},\mathfrak{M}}$ (for all choices of $\phi \in \mathcal{C}\text{ond}$, all $\mathfrak{M}$ satisfying her privacy definition, and all $\mathcal{A}$ satisfying Bob's privacy definition). In particular, she can ask when $\mathfrak{M}^{\star}_{\phi,\mathcal{A},\mathfrak{M}}$ (for all choices of $\phi, \mathfrak{M}, \mathcal{A}$) satisfies ϵ'-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ (i.e. her privacy definition with a different privacy parameter ϵ').

The above discussion generalizes to the case when Bob's algorithm $\mathcal{A}$ takes as input both $\mathfrak{Data}_{Bob}$ as well as the output of Alice's mechanism $\mathfrak{M}(\mathfrak{Data}_{Alice})$. The only

[12] Thus we decompose the joint distribution of $\mathfrak{Data}_{Alice}$ and $\mathfrak{Data}_{Bob}$ into the marginal distribution of $\mathfrak{Data}_{Alice}$ and the conditional distribution of $\mathfrak{Data}_{Bob}$ given $\mathfrak{Data}_{Alice}$.

change would be that in Equation 22, where we would consider

$$E\left[P(\mathcal{A}(\mathfrak{Data}_{Bob}, \mathfrak{M}(D) = \omega^* \mid \mathfrak{Data}_{\text{Alice}} = D, \phi\right] = \int P\left(\mathcal{A}(D', \mathfrak{M}(D)) = \omega^*\right) P\left(\mathfrak{Data}_{\text{Bob}} = D' \mid \mathfrak{Data}_{\text{Alice}} = D, \phi\right) dD'$$

9.2. Self-composition

In this section, we study a special case of the discussion in the previous section. We call this special case *self-composition*. This is a helpful property for privacy definitions to have since it is useful in the design of algorithms. In self-composition, Alice plans multiple independent releases of her own data (i.e. the Alice and Bob from the previous section are the same person, and $\mathcal{C}\text{ond}$ consists of the trivial conditional probability where $P(\mathfrak{Data}_{Bob} = D \mid \mathfrak{Data}_{Alice} = D) = 1$).

Thus, Alice has a dataset $\mathfrak{Data}$, announces a privacy definition ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ and chooses two algorithms $\mathfrak{M}_1$ and $\mathfrak{M}_2$ (with independent sources of randomness) that satisfy this definition. Alice computes $\omega_1 = \mathfrak{M}_1(\mathfrak{Data})$ and $\omega_2 = \mathfrak{M}_2(\mathfrak{Data})$ and releases the sanitized output (ω_1, ω_2) to the public.

From the previous section, we see that this is the same as if Alice had used an algorithm $\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}$ whose range equals $\text{range}(\mathfrak{M}_1) \times \text{range}(\mathfrak{M}_2)$ and with the probabilistic behavior $P\left[\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}(D) = (\omega_1, \omega_2)\right] = P(\mathfrak{M}_1(D) = \omega_1)P(\mathfrak{M}_2(D) = \omega_2)$. Ideally, $\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}$ still satisfies Alice's chosen instantiation of the Pufferfish framework with some privacy parameter ϵ'. This brings up the notion of *linear self-composition*:

Definition 9.2. *(Linear Self-composition).* We say that $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ self-composes linearly if $\forall \epsilon_1, \epsilon_2 > 0$, all $\mathfrak{M}_1$ satisfying ϵ_1-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ and all $\mathfrak{M}_2$ satisfying ϵ_2-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$, the algorithm $\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}$ satisfies $(\epsilon_1 + \epsilon_2)$-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$, where $\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}$ is the algorithm with $\text{range}(\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}) = \text{range}(\mathfrak{M}_1) \times \text{range}(\mathfrak{M}_2)$ such that for all $D \in \mathcal{I}$, $P\left[\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}(D) = (\omega_1, \omega_2)\right] = P[\mathfrak{M}_1(D) = \omega_1]P[\mathfrak{M}_2(D) = \omega_2]$

Linear self-composition is useful for algorithm design because it allows Alice to split a complicated algorithm $\mathfrak{M}$ into a collection of simpler algorithms $\mathfrak{M}_1, \ldots, \mathfrak{M}_k$ and allocate her overall privacy budget ϵ among them [McSherry 2009]. As in the previous section, all our discussion on linear self composition generalizes to the case when $\mathfrak{M}_2$ takes as input both $\mathfrak{Data}$ as well as the output $\omega_1 = \mathfrak{M}_1(\mathfrak{Data})$. This is because like differential privacy, all Pufferfish instantiations satisfy transformation invariance (Axiom 5.1), which ensures that postprocessing the output of a mechanism does not degrade the privacy guarantee.

9.2.1. Sufficient conditions for self-composition. In general, not all instantiations of the Pufferfish framework will self-compose linearly. Furthermore, it is not always easy to tell if a particular instantiation will self-compose linearly. However, we provide an important class of sufficient conditions called *universally composable evolution scenarios*.

When domain experts create instantiations of the Pufferfish framework, they add more and more probability distributions θ into the evolution scenarios $\mathbb{D}$ to create a reasonable (yet conservative) set of data-generating distributions. Each evolution scenario θ adds an additional constraint that an algorithm $\mathfrak{M}$ must satisfy.

If we have a privacy definition $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ that self-composes linearly, it can happen that adding more evolution scenarios (i.e. replacing $\mathbb{D}$ with a strict superset $\mathbb{D}'$) will break the composition property. However, there are some θ that we can always add without worrying about breaking composition. We refer to these special θ as *universally composable evolution scenarios*.

Definition 9.3. *(Universally composable evolution scenarios).* Given $\mathbb{S}$ and $\mathbb{S}_{\text{pairs}}$, we say that θ is a *universally composable evolution scenario for* $\mathbb{S}_{pairs}$ if the privacy definition $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D} \cup \{\theta\})$ self-composes linearly whenever $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ self-composes linearly.

Universally composable evolution scenarios have a very special form.

THEOREM 9.1. *Given $\mathbb{S}$ and $\mathbb{S}_{pairs}$, the probability distribution θ is a universally composable evolution scenario for $\mathbb{S}_{pairs}$ if and only if for all $(s_i, s_j) \in \mathbb{S}_{pairs}$ having $P(s_i \mid \theta) \neq 0$ and $P(s_j \mid \theta) \neq 0$ there exist datasets $D_i, D_j \in \mathcal{I}$ such that $P(\mathfrak{Data} = D_i \mid s_i, \theta) = 1$ and $P(\mathfrak{Data} = D_j \mid s_j, \theta) = 1$*

See Section J of the electronic appendix for the proof. The form of any universally composable evolution scenario θ, as identified by Theorem 9.1, corresponds to an extremely confident attacker. Such an attacker believes that there is only one possible dataset for which s_i is true and only one possible dataset for which s_j is true. This can be due to the attacker possessing enough background knowledge to rule out all other datasets; hence it is a generalization of existing models where an attacker may know all but one bits (or records) in a database [Dinur and Nissim 2003]. The next result follows directly from Theorem 9.1, and provides a sufficient condition for when privacy definitions satisfy self-composition.

COROLLARY 9.4. *Given $\mathbb{S}$ and $\mathbb{S}_{pairs}$, if $\mathbb{D}$ is constructed only using universally composable evolution scenarios, then $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D})$ self-composes linearly.*

The interesting aspect of Theorem 9.1 and Corollary 9.4 is that when the set of evolution scenarios $\mathbb{D}$ consists solely of universally composable evolution scenarios, then the resulting instantiation of the Pufferfish framework is a neighbor-based definition similar to differential privacy.

That is, let $\theta \in \mathbb{D}$ be a universally composable evolution scenario and let $(s_i, s_j) \in \mathbb{S}$ be a discriminative pair with nonzero probability under θ and $D_i, D_j \in \mathcal{I}$ be the datasets associated with θ by Theorem 9.1. Then D_i and D_j can be considered "neighbors" and the Pufferfish constraints (Equations 1 and 2 in Definition 3.4) become:

$$P(\mathfrak{M}(D_i) = \omega) \;\leq\; e^{\epsilon} P(\mathfrak{M}(D_j) = \omega)$$
$$P(\mathfrak{M}(D_j) = \omega) \;\leq\; e^{\epsilon} P(\mathfrak{M}(D_i) = \omega)$$

with randomness only depending on $\mathfrak{M}$.

In the case of differential privacy, those universally composable evolution scenarios θ are those for which there exist databases D_1 and D_2 that differ only on one tuple and have $P(\mathfrak{Data} = D_1 \mid \theta) + P(\mathfrak{Data} = D_2 \mid \theta) = 1$. Note that Theorem 6.1 says that we can further increase $\mathbb{D}$ to include all of the other distributions that generate records independently without change to the privacy guarantees, and Theorem 6.2 says that differentially private algorithms may leak too much information if we include any other distributions (i.e. those with correlated records).

10. DIFFERENTIAL PRIVACY WITH DETERMINISTIC CONSTRAINTS

It was shown in [Kifer and Machanavajjhala 2011] that even state-of-the-art privacy definitions may not compose well with deterministic data constraints such as those caused by previous deterministic releases of information. As we saw in Section 6.3, when a data curator provides exact query answers about the data and subsequently publishes additional information using ϵ-differential privacy, the combined data releases can leak much more information than each of the 2 releases in isolation. Constraints caused by deterministic releases of information are often the result of legal or contractual obligations (e.g., the U.S. Decennial Census).

Kifer and Machanavajjhala [2011] proposed a modification of differential privacy, called *induced neighbors privacy* [Kifer and Machanavajjhala 2011] to account for prior deterministic data releases. As with many variants of differential privacy, it was a "neighbors-based" based definition that tried to make certain pairs of databases indistinguishable from each other. We analyze this definition using Pufferfish to show that it does not always properly bound the attacker's odds ratio.

We then extend the discussion of composition from Section 9 to show how to use Pufferfish to modify differential privacy in a way that takes into account arbitrary deterministic constraints (not just those caused by prior deterministic releases of data). The result is a privacy definition with precise semantic guarantees and clearly specified assumptions under which they hold. We also show some conditions under which induced neighbors privacy [Kifer and Machanavajjhala 2011] is actually equivalent to an instantiation of the Pufferfish framework, thus providing induced neighbors privacy with precise semantic guarantees in those situations.

10.1. Preliminaries

Several types of constraints are common (some of them result from deterministic releases of data):

— ***Counts***: The number of tuples in the dataset or the number of AIDS patients with age less than 25 are pieces of knowledge that impose count constraints. These constraints are often called *marginal constraints*, and can be represented as a constraint like $\sum_{r \in \text{records}(\mathfrak{Data})} g(r) = C$, where g is a function from the tuple domain $\mathcal{T}$ to $\{0, 1\}$, and C is an integer. For instance, to encode a constraint about the number of AIDS patients with age less than 25, one can choose g such that $g(t) = 1$ only when t is a tuple with AIDS and age less than 25, and $g(t) = 0$ otherwise.

— ***Univariate Histograms***: These are a special kind of count constraints $\sum_{r \in \text{records}(\mathfrak{Data})} g_i(r) = C_i$ (for $i = 1, \ldots, k$) where the g_i have disjoint supports (i.e. if for some $t \in \mathcal{T}$ we have $g_i(t) = 1$ then $g_j(t) = 0$ for all other j). Such constraints capture a variety of statistics that might be known about a database, including the total number of rows, number of tuples satisfying a set of mutually exclusive properties, as well as the results of bucketization algorithms that are common in the k-anonymization and ℓ-diversity literature [Chen et al. 2009; Xiao and Tao 2006].

— ***General Deterministic Constraints***: In general, deterministic constraints eliminate some of the databases from the domain of database instances $\mathcal{I}$. Such a general constraint $\mathcal{Q}$ can be formally described as a function $\mathcal{Q} : \mathcal{I} \to \{0, 1\}$ such that $\mathcal{Q}(D) = 0$ means D is not possible (i.e. does not satisfy the constraint) and $\mathcal{Q}(D) = 1$ means D is a possible.

Example 10.1. Let us give an example of a general constraint that will help illustrate the benefits of Pufferfish and distinguish it from neighbor-based privacy definitions. Suppose there are n students with ID numbers ranging from 1 to n. They are scheduled take an oral exam in the order determined by their id numbers. The dataset $\mathfrak{Data}$ tracks whether or not a student has taken the exam. Initially $\mathfrak{Data}$ consists of n records with value 0. After student i takes the exam, the i^{th} record is set to 1. At any point in time, the database $\mathfrak{Data}$ can be in only one of the $n + 1$ states defined by the constraint $\mathcal{Q}$: $\exists k \forall i \leq k$, $r_i = 1 \bigwedge \forall i > k$, $r_i = 0$, as shown below.

D_0	D_1	D_2	...	D_{n-2}	D_{n-1}	D_n
0	1	1	...	1	1	1
0	0	1	...	1	1	1
...	...	...	...	...	...	...
0	0	0	...	0	1	1
0	0	0	...	0	0	1

Suppose at some time point, the data curator wants to release the current number of students who have taken the exam without revealing who has or has not taken the test. The Laplace mechanism, which satisfies ϵ-differential privacy, would add noise with density $f(x) = \frac{1}{2\epsilon}e^{-|x|/\epsilon}$ to the current number of students who have taken the exam. When n, the number of total students, is large, this strategy leaks too much information. For example, if the noisy answer is close to 0, the true value was probably not n and so the n^{th} student probably has not yet taken the exam. As we shall see, it is not possible to release meaningful information in this situation, but differential privacy does not warn us about it (neither will induced-neighbors privacy, Definition 10.3).

Induced neighbors privacy [Kifer and Machanavajjhala 2011] uses the following definitions:

Definition 10.2. *(Move [Kifer and Machanavajjhala 2011]).* Given a database D, a *move* m is a process that adds or deletes a tuple from D, resulting in a database $m(D)$.

Definition 10.3. *(Induced Neighbors $\mathcal{N}_Q$ [Kifer and Machanavajjhala 2011]).* Given a general constraint $\mathcal{Q}$, let $\mathcal{I}_\mathcal{Q}$ be the set of databases satisfying those constraints. Let D_a and D_b be two databases. Let n_{ab} be the smallest number of moves necessary to transform D_a into D_b and let $\{m_1, \ldots, m_{n_{ab}}\}$ be the set of those moves. The databases D_a and D_b are called *neighbors induced by* $\mathcal{Q}$, denoted as $(D_a, D_b) \in \mathcal{N}_\mathcal{Q}$, if the following holds.

— $D_a \in \mathcal{I}_\mathcal{Q}$ and $D_b \in \mathcal{I}_\mathcal{Q}$.
— No subset of $\{m_1, \ldots, m_{n_{ab}}\}$ can transform D_a into some $D_c \in \mathcal{I}_\mathcal{Q}$.

Definition 10.4. *(Induced Neighbors Privacy [Kifer and Machanavajjhala 2011]).* An algorithm $\mathfrak{M}$ satisfies induced neighbor privacy with constraint $\mathcal{Q}$, if for each output $\omega \in \text{range}(\mathfrak{M})$ and for every pair D_1, D_2 of neighbors induced by Q, the following holds:

$$P(\mathfrak{M}(D_1) = \omega) \leq e^\epsilon P(\mathfrak{M}(D_2) = \omega)$$

It is easy to see that the Laplace mechanism from Example 10.1 also satisfies induced neighbor privacy for this particular scenario since all induced neighbors are pairs (D_i, D_{i+1}). We compute the amount of information leakage (Example 10.5) after considering the Pufferfish view.

10.2. Pufferfish with Deterministic Constraints

Following the notation from Section 6 (also see Table I containing our notation), define:

$$\mathbb{S} = \left\{\sigma_{(i,t))} \ : \ h_i \in \mathcal{H},\ t \in \mathcal{T}\right\} \cup \{\neg\sigma_i \ : \ h_i \in \mathcal{H}\} \tag{23}$$

$$\mathbb{S}_{\text{pairs}} = \left\{(\sigma_{(i,t)}, \neg\sigma_i) \ : \ h_i \in \mathcal{H}, t \in \mathcal{T}\right\} \cup \left\{(\sigma_{(i,t)}, (\sigma_{i,t'})) \ : \ h_i \in \mathcal{H}, t, t' \in \mathcal{T}\right\} \tag{24}$$

Thus, the goal is to prevent an attacker from (a) learning whether the record of individual h_i is in the data, and (b) distinguishing between two possible values of r_i (in case h_i is known to be in the data). The data evolution scenarios $\mathbb{D}^*_\mathcal{Q}$ is the set of all probability distributions with the following form (i.e., they generate records independently

conditioned on $\mathcal{Q}$):

$$\theta \equiv \{\pi_1, \ldots, \pi_N, f_1, \ldots, f_N, \mathcal{Q}\} \tag{25}$$

$$P(\mathfrak{Data} \mid \theta) = \begin{cases} 0 & \text{if } \mathcal{Q}(\mathfrak{Data}) = 0 \\ \frac{1}{Z_{\mathcal{Q}}} \prod_{r_i \in \text{records}(\mathfrak{Data})} f_i(r_i)\pi_i \prod_{r_j \notin \text{records}(\mathfrak{Data})} (1-\pi_j) & \text{otherwise} \end{cases}$$

where the normalization constant $Z_{\mathcal{Q}} = P(\mathcal{Q}(\mathfrak{Data}) = 1 \mid \theta)$.

We show that ϵ-induced neighbors privacy is a necessary condition for guaranteeing ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}})$, for any general constraint $\mathcal{Q}$.

THEOREM 10.1. (Necessary Condition). *Given a set of general constraint $\mathcal{Q}$, if $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*_{\mathcal{Q}})$ (with $\mathbb{S}$, $\mathbb{S}_{pairs}$, and $\mathbb{D}^*_{\mathcal{Q}}$ given by Equations 23, 24, and 25) then $\mathfrak{M}$ satisfies ϵ-induced neighbors privacy with respect to $\mathcal{Q}$.*

For proof see Section K of the electronic appendix. However, the next example shows ϵ-induced neighbors privacy is not sufficient, hence it does not guarantee an attacker's odds ratio is bounded within $[e^{-\epsilon}, e^{\epsilon}]$.

Example 10.5. Continuing Example 10.1, we show that the Laplace mechanism, which satisfies both ϵ-differential privacy and ϵ-induced neighbors privacy, does not satisfy ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}})$.

Consider a θ of the form given in Equation 25 such that for all i, $f_i(0) = f_i(1) = 0.5$, $\pi_i = 1$. Thus all the allowable datasets $D_0, \ldots, D_n$ from Example 10.1 are equally likely under θ. Consider the discriminative pair $(\sigma_{(1,0)}, \sigma_{(1,1)})$ and note that if record 1 has value 0 then, according to our constraints, so do all records r_ℓ for $\ell > 1$, so D_0 is the only dataset for which $\sigma_{(1,0)}$ can be true.

$$\begin{aligned} P(\mathfrak{M}(\mathfrak{Data}) = n \mid \sigma_{(1,1)}, \theta) &= \sum_{j=1}^{n} \frac{P(\mathfrak{M}(D_j) = n)}{n} = \sum_{j=1}^{n} \frac{\epsilon}{2n} e^{-\epsilon(n-j)} \\ &> e^{\epsilon} \cdot \frac{\epsilon}{2} e^{-\epsilon n} = e^{\epsilon} \cdot P(\mathfrak{M}(\mathfrak{Data}) = n \mid \sigma_{(1,0)}, \theta) \end{aligned}$$

Therefore satisfying ϵ-differential privacy or induced neighbors privacy in this situation does not bound an attacker's odds-ratio to the range $[e^{-\epsilon}, e^{\epsilon}]$.

In this situation, ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}})$ requires

$$\forall D_i, D_j,\ P(\mathfrak{M}(D_i) = \omega) \leq e^{\epsilon} \cdot P(\mathfrak{M}(D_j) = \omega) \tag{26}$$

The condition is clearly sufficient. Necessity can be shown by considering the output $\omega = n$, and different priors θ_i that assign $f_j(1) = 1 - \delta$ for $j \leq i$, and $f_j(1) = \delta$ otherwise, where δ tends to zero. These priors capture different adversaries who believe strongly (with high prior probability) that i students have taken the exam.

Pufferfish tells us (via Equation 26) that we cannot release meaningful data in this situation because it reduces to the condition that attackers should not be able to distinguish between any pair of valid datasets. However, we next show that it is possible to release meaningful data for a broad class of typical constraints.

10.3. Pufferfish with Univariate Histograms

Univariate histograms, as defined in Section 10.1, form an important subclass of constraints. For instance, the Census Bureau is legally obliged to publish the exact number of people living in each state [Cantwell et al. 2004]. A search engine is contractually bound to report to an advertiser the number of users who have clicked on ads using mutually exclusive predefined ranges (e.g., $100 - 200$ clicks). Another interesting use

case occurs when there has been a prior release of data using a mechanism $\mathfrak{M}_{uni}$ based on statistical disclosure limitation techniques like partitioning and microaggregation [Adam and Worthmann 1989], or bucketization algorithms based on syntactic privacy notions like k-anonymity (with say $k = 10,000$), ℓ-diversity, etc. [Chen et al. 2009; LeFevre et al. 2006; Xiao and Tao 2006]. The output of $\mathfrak{M}_{uni}$ in all the above cases is a univariate histogram. In all these cases, we can provide additional releases of information by using Pufferfish to limit any further inference an attacker could make.

In fact, for those cases, ϵ-induced neighbor privacy becomes an instantiation of the Pufferfish framework (Theorems 10.1 and 10.2).

THEOREM 10.2. (SUFFICIENT CONDITION FOR UNIVARIATE HISTOGRAMS). *Given a univariate histogram constraint $\mathcal{Q}_{uni} : \{\sum_{t \in \boldsymbol{Data}} g_i(t) = C\}$, define $\mathbb{S}$ and $\mathbb{S}_{pairs}$ as in Equations 23 and 24 and let $\mathbb{D}^*_Q$ be the set of all distributions with form specified in Equation 25. Then $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*_{\mathcal{Q}_{uni}})$ if $\mathfrak{M}$ satisfies ϵ-induced neighbors privacy with respect to $\mathcal{Q}_{uni}$.*

For proof see Section L of the electronic appendix. Thus the algorithms proposed in [Kifer and Machanavajjhala 2011] can be used in this case to bound an attacker's inference.

An important question that was left open in [Kifer and Machanavajjhala 2011] is whether induced neighbor privacy is linear self composable. Theorems 10.1 and 10.2 allow us to answer this question. Since ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}_{uni}})$ and induced neighbor privacy (for univariate histograms) are equivalent definitions, it is easy to see that the former can be written solely in terms of universally composable evolution scenarios, proving that ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}_{uni}})$ composes with itself linearly. This means that, for a database with a prior univariate histogram release $\mathcal{Q}_{uni}$, and further releases using $\mathfrak{M}_1$ and $\mathfrak{M}_2$ that satisfy $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}_{uni}})$ with parameters ϵ_1 and ϵ_2, respectively, the combined mechanism $\mathfrak{M}_{\mathfrak{M}_1,\mathfrak{M}_2}$ guarantees $(\epsilon_1 + \epsilon_2)$-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}_{uni}})$.

11. PUFFERFISH AND THE DE FINETTI ATTACK

In this section, we discuss the relationship between the Pufferfish framework and an attack that is successful against certain types of data anonymization algorithms, called the de Finetti attack [Kifer 2009]. We first provide a brief overview of the de Finetti attack [Kifer 2009] and then we discuss when an instantiation of the Pufferfish framework is immune to the de Finetti attack.

11.1. Review of the de Finetti Attack

The de Finetti attack [Kifer 2009] is the application of Bayesian reasoning to sanitized data to simultaneously estimate the data generating distribution while using this knowledge to undo the anonymization. Note that learning about the true distribution is not considered a privacy breach [Dwork 2006]. However, if the true distribution can be used to undo the anonymization in a dataset, then additional information will be leaked and this can be considered a privacy breach.

As an illustration of these ideas, consider a dataset such as the one shown in Figure 1 (based on examples from [Kifer and Machanavajjhala 2011]). Such a dataset may be output by an anonymization algorithm that performs a variation of bucketization (e.g., [Xiao and Tao 2006; Zhang et al. 2007]).

The dataset in Figure 1 consists of two tables. The first table contains tuple ids (TID), demographics (gender, age, zip code, smoking status), and a group id (GID) that is generated by an anonymization algorithm. The second table contains the cancer status of individuals in a group. Thus for each group, we know the demographics of the

TID	Gender	Age	Zip Code	Smoker?	GID
1	M	25	90210	Y	1
3	M	29	90212	Y	1
7	F	40	07620	N	2
8	F	24	33109	N	2
2	F	43	90211	N	3
5	F	41	07620	Y	3
9	M	48	07620	Y	4
10	M	48	07620	N	4
4	M	41	90213	Y	5
6	F	40	33109	Y	5
11	M	48	33109	N	6
12	M	49	33109	N	6
⋮	⋮	⋮	⋮	⋮	⋮

GID	Cancer?
1	Y
1	N
2	N
2	N
3	Y
3	N
4	Y
4	N
5	N
5	Y
6	N
6	N
⋮	⋮

Fig. 1. An anonymized dataset used to illustrate the de Finetti attack.

group and the number of cancer patients, but we do not have an explicit link between which tuple in a group corresponds to a cancer patient (in this table the number of cancer patients in a group is always less than 2 so it is not possible that all members of a group have cancer).

Now, an analyst looking at the tables in Figure 1 would see that groups that have smokers are also more likely to have cancer patients. For concreteness, let us suppose that by statistical means the analyst has learned from the dataset that $P(\text{cancer} \mid \text{smoker}) = 0.05$ and $P(\text{cancer} \mid \text{non-smoker}) = 0.01$. Thus knowing that some random individual Bob is a smoker would mean there is a 5% chance that Bob has cancer. This would not be considered a privacy breach since that inference is based on estimates of the underlying distribution.

However, if Bob is a 48-year-old male smoker who lives in zip code 07620 and is known to be in the table, then estimates about the underlying distribution can be used to partially undo the anonymization and make better inferences about Bob. To see why, our background knowledge implies that Bob corresponds to tuple 9 in the table in Figure 1. The corresponding group contains Bob (a smoker) and a non-smoker. The second table in Figure 1 shows that Bob's group contains exactly one cancer patient. Who is more likely to be the cancer patient, Bob or the non-smoker? According to the analyst's estimates, $P(\text{cancer} \mid \text{smoker}) = 0.05$ and $P(\text{cancer} \mid \text{non-smoker}) = 0.01$. The probability of seeing a dataset in which exactly one of them has cancer is therefore $0.05(1 - 0.01) + 0.01(1 - 0.05) = 0.059$ and so the probability that Bob has cancer becomes $0.05(1 - 0.01) \;/\; 0.059 \approx 0.84$. Thus undoing the anonymization has sharpened the inference about Bob (the probability of cancer went from 0.05 to 0.84) and this has caused a privacy breach.

We next address whether a Pufferfish privacy definition can prevent such an attack.

11.2. Resistance to the Attack

The attacker in the de Finetti attack initially considers a set $\mathbb{D}_{\text{attacker}}$ of probability distributions to be possible. Then, based on the sanitized data, the attacker decides which of these probability distributions are better (in terms of explaining the data) than others.

The Pufferfish framework will resist the de Finetti attack if the set of evolution scenarios $\mathbb{D}$ contains $\mathbb{D}_{\text{attacker}}$. The semantic guarantees described in Section 3.1 imply that for any probability distribution $\theta \in \mathbb{D}_{\text{attacker}}$, using θ to try to de-anonymize

sanitized data (i.e. using θ to make probabilistic inference about sensitive information) will only change an attacker's odds ratio by at most e^{ϵ}. In contrast, the attacker's odds ratio from the discussion in Section 11.1 changed from $0.05/(1-0.05) \approx 0.053$ to $0.84/(1-0.84) \approx 5.25$, an increase of two orders of magnitude.

Thus if the set of evolution scenarios $\mathbb{D}$ contains $\mathbb{D}_{\text{attacker}}$ then the attacker's beliefs can change, but most of the change is the result of learning about which distribution $\theta \in \mathbb{D}_{\text{attacker}}$ best approximates the true distribution; very little of the change in beliefs could be attributed to undoing the sanitization because Pufferfish provides the guarantee that none of the $\theta \in \mathbb{D}_{\text{attacker}}$ would be useful in learning about sensitive information.

Consideration of the de Finetti attack shows why the evolution scenarios should be a conservative set of data generating distributions, (such as the set of all distributions in which records are i.i.d.); to avoid attacks, the data evolution scenarios $\mathbb{D}$ need to contain all of the probability distributions that would be considered possible by a reasonable attacker (hence a domain expert is needed).

12. CONCLUSIONS

We presented the Pufferfish framework, a new and general framework that allows application domain experts to develop rigorous privacy definitions for their data sharing needs. The framework provides crisp Bayesian semantics and allows the domain experts to customize privacy to the specific set of secrets and data evolution scenarios that are typical in their domain.

ELECTRONIC APPENDIX

The electronic appendix for this article can be accessed in the ACM Digital Library.

Acknowledgments

The authors would like to thank the anonymous reviewers for their helpful comments and suggestions.

REFERENCES

ADAM, N. AND WORTHMANN, J. 1989. Security-control methods for statistical databases. *ACM Computing Surveys 21,* 4, 515–556.

AGGARWAL, C. C., PEI, J., AND ZHANG, B. 2006. On privacy preservation against adversarial data mining. In *KDD*. ACM, New York, NY, USA.

BHASKAR, R., BHOWMICK, A., GOYAL, V., LAXMAN, S., AND THAKURTA, A. 2011. Noiseless database privacy. In *Proceedings of the 17th international conference on The Theory and Application of Cryptology and Information Security (ASIACRYPT)*.

BLUM, A., LIGETT, K., AND ROTH, A. 2008. A learning theory approach to non-interactive database privacy. In *STOC*. ACM, New York, NY, USA, 609–618.

CANTWELL, P. J., HOGAN, H., AND STYLES, K. M. 2004. The use of statistical methods in the u.s. census: Utah v. evans. *The American Statistician 58,* 3, 203–212.

CHAN, H., SHI, E., AND SONG, D. 2010. Private and continual release of statistics. In *ICALP*.

CHAUDHURI, K. AND MISHRA, N. 2006. When random sampling preserves privacy. In *CRYPTO*.

CHEN, B.-C., KIFER, D., LEFEVRE, K., AND MACHANAVAJJHALA, A. 2009. Privacy-preserving data publishing. *Foundations and Trends in Databases 2,* 1-2, 1–167.

CLIFTON, C. 2000. Using sample size to limit exposure to data mining. *Journal of Computer Security 8,* 4, 281–307.

CLIFTON, C., KANTARCIOGLU, M., AND VAIDYA, J. 2002. Defining privacy for data mining. In *Proc. of the NSF Workshop on Next Generation Data Mining*.

CLIFTON, C. AND MARKS, D. 1996. Security and privacy implications of data mining. In *Proceedings of the ACM SIGMOD Workshop on Data Mining and Knowledge Discovery*. ACM, New York, NY, USA.

DELIS, A., VERYKIOS, V. S., AND TSITSONIS, A. A. 2010. A data perturbation approach to sensitive classification rule hiding. In *SAC*.

DINUR, I. AND NISSIM, K. 2003. Revealing information while preserving privacy. In *PODS*.

DUAN, Y. 2009. Privacy without noise. In *CIKM*.

DWORK, C. 2006. Differential privacy. In *ICALP*.

DWORK, C. 2008. Differential privacy: A survey of results. In *TAMC*.

DWORK, C., KENTHAPADI, K., MCSHERRY, F., MIRONOV, I., AND NAOR, M. 2006a. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*. 486–503.

DWORK, C., MCSHERRY, F., NISSIM, K., AND SMITH, A. 2006b. Calibrating noise to sensitivity in private data analysis. In *TCC*.

DWORK, C., MCSHERRY, F., AND TALWAR, K. 2007. The price of privacy and the limits of lp decoding. In *STOC*.

DWORK, C. AND NAOR, M. 2010. On the difficulties of disclosure prevention in statistical databases or the case for differential privacy. *JPC 2,* 1.

DWORK, C., NAOR, M., PITASSI, T., AND ROTHBLUM, G. N. 2010a. Differential privacy under continual observation. In *Proceedings of the 42nd ACM symposium on Theory of computing (STOC)*.

DWORK, C., NAOR, M., PITASSI, T., ROTHBLUM, G. N., AND YEKHANIN, S. 2010b. Pan-private streaming algorithms. In *Proceedings of ICS*.

FIENBERG, S. E. 1997. Confidentiality and disclosure limitation methodology: Challenges for national statistics and statistical research. Tech. Rep. 668, CMU.

FUNG, B. C. M., WANG, K., CHEN, R., AND YU, P. S. 2010. Privacy-preserving data publishing: A survey on recent developments. *ACM Computing Surveys 42,* 4.

GANTA, S. R., KASIVISWANATHAN, S. P., AND SMITH, A. 2008. Composition attacks and auxiliary information in data privacy. In *KDD*. ACM, New York, NY, USA.

GEHRKE, J., HAY, M., LUI, E., AND PASS, R. 2012. Crowd-blending privacy. In *CRYPTO*.

GEHRKE, J., LUI, E., AND PASS, R. 2011. Towards privacy for social networks: A zero-knowledge based definition of privacy. In *TCC*.

GHOSH, A., ROUGHGARDEN, T., AND SUNDARARAJAN, M. 2009. Universally utility-maximizing privacy mechanisms. In *STOC*. 351–360.

HALL, R., WASSERMAN, L., AND RINALDO, A. 2012. Random differential privacy. *Journal of Privacy and Confidentiality 4,* 2.

HOMER, N., SZELINGER, S., REDMAN, M., DUGGAN, D., TEMBE, W., MUEHLING, J., PEARSON, J. V., STEPHAN, D. A., NELSON, S. F., AND CRAIG, D. W. 2008. Resolving individuals contributing trace amounts of dna to highly complex mixtures using high-density snp genotyping microarrays. *PLoS Genet 4,* 8.

KASIVISWANATHAN, S. P. AND SMITH, A. 2008. A note on differential privacy: Defining resistance to arbitrary side information. http://arxiv.org/abs/0803.3946.

KIFER, D. 2009. Attacks on privacy and de finetti's theorem. In *SIGMOD*.

KIFER, D. AND LIN, B.-R. 2010. Towards an axiomatization of statistical privacy and utility. In *PODS*. ACM, New York, NY, USA.

KIFER, D. AND LIN, B.-R. 2012. An axiomatic view of statistical privacy and utility. *Journal of Privacy and Confidentiality 4,* 1.

KIFER, D. AND MACHANAVAJJHALA, A. 2011. No free lunch in data privacy. In *SIGMOD*. ACM, New York, NY, USA.

KIFER, D. AND MACHANAVAJJHALA, A. 2012. A rigorous and customizable framework for privacy. In *PODS*.

LEFEVRE, K., DEWITT, D., AND RAMAKRISHNAN, R. 2006. Mondrian multidimensional k-anonymity. In *ICDE*.

MACHANAVAJJHALA, A., KIFER, D., ABOWD, J., GEHRKE, J., AND VILHUBER, L. 2008. Privacy: From theory to practice on the map. In *ICDE*.

MACHANAVAJJHALA, A., KOROLOVA, A., AND SARMA, A. D. 2011. Personalized social recommendations - accurate or private? In *Proceedings of Very Large Data Bases (PVLDB)*. Vol. 4. 440–450.

MCSHERRY, F. D. 2009. Privacy integrated queries: An extensible platform for privacy-preserving data analysis. In *SIGMOD*. ACM, New York, NY, USA, 19–30.

MIRONOV, I., PANDEY, O., REINGOLD, O., AND VADHAN, S. 2009. Computational differential privacy. In *CRYPTO*.

MOUSTAKIDES, G. V. AND VERYKIOS, V. S. 2008. A maxmin approach for hiding frequent itemsets. *Data Knowl. Eng. 65,* 1, 75–89.

NATWICHAI, J., LI, X., AND ORLOWSKA, M. E. 2006. A reconstruction-based algorithm for classification rules hiding. In *ADC*.

NISSIM, K., RASKHODNIKOVA, S., AND SMITH, A. 2007. Smooth sensitivity and sampling in private data analysis. In *STOC*. ACM, New York, NY, USA, 75–84.

OLIVEIRA, S. R. M. AND ZAIANE, O. R. 2003. Algorithms for balancing privacy and knowledge discovery in association rule mining. In *International Database Engineering and Applications Symposium*.

PREDICT 2005. PREDICT (protected repository for the defense of infrastructure against cyber threats). https://www.predict.org.

RASTOGI, V., HAY, M., MIKLAU, G., AND SUCIU, D. 2009. Relationship privacy: Output perturbation for queries with joins. In *PODS*. ACM, New York, NY, USA, 107–116.

SAMARATI, P. 2001. Protecting respondents' identities in microdata release. *TKDE 13,* 6, 1010 – 1027.

SANKARARAMAN, S., OBOZINSKI, G., JORDAN, M. I., AND HALPERIN, E. 2009. Genomic privacy and limits of individual detection in a pool. *Nature genetics 41,* 9, 965–967.

VERYKIOS, V. S., BERTINO, E., FOVINO, I. N., PROVENZA, L. P., SAYGIN, Y., AND THEODORIDIS, Y. 2004a. State-of-the-art in privacy preserving data mining. *SIGMOD Rec. 33,* 1, 50–57.

VERYKIOS, V. S., ELMAGARMID, A. K., BERTINO, E., SAYGIN, Y., AND DASSENI, E. 2004b. Association rule hiding. *IEEE Trans. Knowl. Data Eng. 16,* 4, 434 – 447.

WANG, E. T. AND LEE, G. 2008. An efficient sanitization algorithm for balancing information privacy and knowledge discovery in association patterns mining. *Data & Knowledge Engineering 65,* 3, 463–484.

WANG, K., FUNG, B., AND YU, P. 2005. Template-based privacy preservation in classification problems. In *ICDM*.

XIAO, X. AND TAO, Y. 2006. Anatomy: Simple and effective privacy preservation. In *VLDB*.

ZHANG, Q., KOUDAS, N., SRIVASTAVA, D., AND YU, T. 2007. Aggregate query answering on anonymized tables. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE)*.

ZHOU, S., LIGETT, K., AND WASSERMAN, L. 2009. Differential privacy with compression. In *ISIT*.

Received Month Year; revised Month Year; accepted Month Year

Online Appendix to: Pufferfish: A Framework for Mathematical Privacy Definitions

DANIEL KIFER, Penn State University
ASHWIN MACHANAVAJJHALA, Duke University

A. PROOF OF THEOREM 3.1

THEOREM 3.1. *Let $\epsilon > 0$, let $\mathbb{S}$ and $\mathbb{S}_{pairs}$ be specified as in Equations 3 and 4 and let $\mathbb{D}$ be the set of all possible distributions over database instances. Then an algorithm $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D})$ if and only if for **every** pair of databases D_1 and D_2 and every $\omega \in \text{range}(\mathfrak{M})$,*

$$e^{-\epsilon} P(\mathfrak{M}(D_2) = \omega) \leq P(\mathfrak{M}(D_1) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_2) = \omega)$$

PROOF. **Case 1:** Let $\mathfrak{M}$ be an algorithm that satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$.

Let D_1 and D_2 be any two distinct tables. Since they are not equal, they either differ on (1) the individuals whose records they contain, and/or (2) the value of some common individual's record. In any case, there exists a discriminative pair $(s, s^*) \in \mathbb{S}_{\text{pairs}}$ such that the potential secret s is true for D_1 and s^* is true for D_2. Define a probability distribution θ so that $P(D_1 \mid \theta) = 1/2$ and $P(D_2 \mid \theta) = 1/2$. Clearly $\theta \in \mathbb{D}$ because $\mathbb{D}$ contains all probability distributions. Recalling the notation that $\mathfrak{Data}$ is a random variable governed by θ, we have $P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid s, \theta) = P(\mathfrak{M}(D_1) = \omega)$ because D_1 is the only possible dataset (according to the distribution θ) for which s is true. Note also that $P(\mathfrak{M}(D_1) = \omega)$ is a statement about the randomness in $\mathfrak{M}_1$ only (for the θ that we chose, conditioning on s made everything deterministic). Similarly $P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid s^*, \theta) = P(\mathfrak{M}(D_2) = \omega)$. Thus since $\mathfrak{M}$ satisfies the instantiation of Pufferfish,

$$P(\mathfrak{M}(D_1) = \omega) \;=\; P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid s, \theta) \leq e^{\epsilon} P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid s^*, \theta) = e^{\epsilon} P(\mathfrak{M}(D_2) = \omega)$$

Similarly, $P(\mathfrak{M}(D_2) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_1) = \omega)$. Since D_1 and D_2 are arbitrary, this finishes the "only if" part of the proof.

Case 2: Now we show the "if" direction (i.e. if $\mathfrak{M}$ satisfies the conditions listed in Theorem 3.1 it also satisfies this instantiation of Pufferfish).

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

Pick any discriminative pair $(s, s^*) \in \mathbb{S}_{\text{pairs}}$.

$$
\begin{aligned}
& P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = r \mid s, \theta) \\
&= \int P(\mathfrak{M}(D) = \omega) P(\boldsymbol{\mathfrak{Data}} = D \mid s, \theta)\, dD \\
&= \int P(\mathfrak{M}(D) = \omega) \left(\int P(\boldsymbol{\mathfrak{Data}} = D' \mid s^*, \theta)\, dD' \right) \times P(\boldsymbol{\mathfrak{Data}} = D \mid s, \theta)\, dD \\
&= \int \left(\int P(\mathfrak{M}(D) = \omega) P(\boldsymbol{\mathfrak{Data}} = D' \mid s^*, \theta)\, dD' \right) \times P(\boldsymbol{\mathfrak{Data}} = D \mid s, \theta)\, dD \\
&\leq e^{\epsilon} \int \left(\int P(\mathfrak{M}(D') = \omega) P(\boldsymbol{\mathfrak{Data}} = D' \mid s^*, \theta)\, dD' \right) \times P(\boldsymbol{\mathfrak{Data}} = D \mid s, \theta)\, dD \\
&\quad \text{(because } P(\mathfrak{M}(D) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D') = \omega)\text{)} \\
&= e^{\epsilon} \left(\int P(\mathfrak{M}(D') = \omega) P(\boldsymbol{\mathfrak{Data}} = D' \mid s^*, \theta)\, dD' \right) \times \int P(\boldsymbol{\mathfrak{Data}} = D \mid s, \theta)\, dD \\
&= e^{\epsilon} \int P(\mathfrak{M}(D') = \omega) P(\boldsymbol{\mathfrak{Data}} = D' \mid s^*, \theta)\, dD' \\
&= e^{\epsilon} P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = \omega \mid s^*, \theta)
\end{aligned}
$$

A similar calculation shows that $P(\mathfrak{M}(data) = \omega \mid s^*, \theta) \leq e^{\epsilon} P(\mathfrak{M}(data) = \omega \mid s, \theta)$ and so $\mathfrak{M}$ also satisfies this instantiation of Pufferfish. □

B. PROOF OF THEOREM 5.1

THEOREM 5.1. *For every $\mathbb{S}$, $\mathbb{S}_{pairs}$, $\mathbb{D}$, and $\epsilon > 0$, the privacy definition ϵ-$\mathcal{P}\text{ufferF}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D})$ satisfies the axioms of convexity and transformation invariance.*

PROOF. Note that we are using measure-theoretic notation that unifies sums and integrals.

Step 1: First we prove it satisfies the axiom of transformation invariance. Let $\mathfrak{M}$ be an algorithm that satisfies ϵ-$\mathcal{P}\text{ufferF}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ and let $\mathcal{A}$ be any algorithm whose domain contains the range of $\mathfrak{M}$ and whose random bits are independent from those of $\mathfrak{M}$. For any $(s_i, s_j) \in \mathbb{S}$, $\omega^* \in \text{range}(\mathcal{A})$, and $\theta \in \mathbb{D}$ such that $P(s_i \mid \theta) \neq 0$ and $P(s_j \mid \theta) \neq 0$,

$$
\begin{aligned}
& P(\mathcal{A}(\mathfrak{M}(\boldsymbol{\mathfrak{Data}})) = \omega^* \mid s_i, \theta) \\
&= \int P(\mathcal{A}(\mathfrak{M}(D)) = \omega^*) P(\boldsymbol{\mathfrak{Data}} = D \mid s_i, \theta)\, dD \\
&= \int \int P(\mathcal{A}(\omega) = \omega^*) P(\mathfrak{M}(D) = \omega) P(\boldsymbol{\mathfrak{Data}} = D | s_i, \theta)\, d\omega\, dD \\
&= \int P(\mathcal{A}(\omega) = \omega^*) \int P(\mathfrak{M}(D) = \omega) P(\boldsymbol{\mathfrak{Data}} = D | s_i, \theta)\, dD\, d\omega \\
&= \int P(\mathcal{A}(\omega) = \omega^*) P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = \omega \mid s_i, \theta)\, d\omega \\
&\leq e^{\epsilon} \int P(\mathcal{A}(\omega) = \omega^*) P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = \omega \mid s_j, \theta)\, d\omega \\
&\quad \text{(Switching from } s_i \text{ to } s_j \text{ with a multiplier of } e^{\epsilon} \text{ since } \mathfrak{M} \\
&\quad \text{satisfies this instantiation of Pufferfish)} \\
&= e^{\epsilon} P(\mathcal{A}(\mathfrak{M}(\boldsymbol{\mathfrak{Data}})) = \omega^* \mid s_j, \theta)
\end{aligned}
$$

Similarly, $P(\mathcal{A}(\mathfrak{M}(\mathfrak{Data})) = \omega^* \mid s_j, \theta) \leq e^\epsilon P(\mathcal{A}(\mathfrak{M}(\mathfrak{Data})) = \omega^* \mid s_i, \theta)$. Thus $\mathcal{A} \circ \mathfrak{M}$ also satisfies the privacy definition.

Step 2: Now we prove that it satisfies the axiom of convexity. Let $\mathfrak{M}_1$ and $\mathfrak{M}_2$ be algorithms that satisfy the privacy definition ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$, let $p \in [0, 1]$ and let $\mathfrak{M}^p$ be the algorithm that runs $\mathfrak{M}_1$ with probability p and $\mathfrak{M}_2$ with probability $1 - p$. For any $\omega \in \text{range}(\mathfrak{M}^p)$,

$$\begin{aligned}
&P(\mathfrak{M}^p(\mathfrak{Data}) = \omega \mid s_i, \theta) \\
&= pP(\mathfrak{M}_1(\mathfrak{Data}) = \omega \mid s_i, \theta) + (1-p)P(\mathfrak{M}_2(\mathfrak{Data}) = \omega \mid s_i, \theta) \\
&\leq e^\epsilon pP(\mathfrak{M}_1(\mathfrak{Data}) = \omega \mid s_j, \theta) + e^\epsilon(1-p)P(\mathfrak{M}_2(\mathfrak{Data}) = \omega \mid s_j, \theta) \\
&= e^\epsilon P(\mathfrak{M}^p(\mathfrak{Data}) = \omega \mid s_j, \theta)
\end{aligned}$$

and similarly $P(\mathfrak{M}^p(\mathfrak{Data}) = \omega \mid s_j, \theta) \leq e^\epsilon P(\mathfrak{M}^p(\mathfrak{Data}) = \omega \mid s_i, \theta)$. Thus $\mathfrak{M}^p$ also satisfies the privacy definition. □

C. PROOF OF THEOREM 6.1

THEOREM 6.1. *Let $\mathbb{S}$ and $\mathbb{S}_{pairs}$ be defined as in Equations 6 and 7. Let $\mathbb{D}^*$ be the set of all distributions of the form specified in Equation 8. With these choices, ϵ-differential privacy is equivalent to ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*)$.*

PROOF.

Case 1: We first show that a mechanism satisfying ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*)$ also satisfies ϵ differential privacy.

Let D_1 and D_2 be two databases such that D_2 is obtained from D_1 by deleting one tuple. Let $\{r_{j_1}, \ldots, r_{j_n}\}$ be the records in D_1 (corresponding to individuals $\{h_{j_i}, \ldots, h_{j_n}\}$) with values $r_{j_1} = t_{j_1}, \ldots, r_{j_n} = t_{j_n}$. Let j_1 be used to denote the index of the individual whose record was dropped, so that D_2 is obtained from D_1 by dropping record r_{j_1}. Define $\pi_{j_1} = 1/2$. Then define $\pi_{j_2} = \ldots \pi_{j_n} = 1$ and $\pi_\ell = 0$ for all individuals h_ℓ whose records do not appear in D_1. Define $f_{j_1}(t_{j_1}) = \cdots = f_{j_n}(t_{j_n}) = 1$ (and 0 for all other tuple values). Define f_ℓ arbitrarily for all individuals h_ℓ who do not appear in D_1. Set $\theta = \{\pi_1, \ldots, \pi_N, f_1, \ldots, f_N\}$. With this setting, D_1 and D_2 are the only two datasets with nonzero probability, D_1 is the only dataset with nonzero probability for which $\sigma_{(j_1, t_{j_1})}$ is true, and D_2 is the only dataset with nonzero probability for which $\neg\sigma_{j_1}$ is true. Now, since $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*)$, we have:

$$\begin{aligned}
&P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid \sigma_{(j_1, t_{j_1})}, \theta) \leq e^\epsilon P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid \neg\sigma_{j_1}, \theta) \\
&\Leftrightarrow\ P(\mathfrak{M}(D_1) = \omega \mid \sigma_{(j_1, t_{j_1})}, \theta) \leq e^\epsilon P(\mathfrak{M}(D_2) = \omega \mid \neg\sigma_{j_1}, \theta) \\
&\Leftrightarrow\ P(\mathfrak{M}(D_1) = \omega) \leq e^\epsilon P(\mathfrak{M}(D_2) = \omega)
\end{aligned}$$

Which is one of the differential privacy constraints. A similar calculation shows that $\mathfrak{M}$ must satisfy $P(\mathfrak{M}(D_2) = \omega) \leq e^\epsilon P(\mathfrak{M}(D_1) = \omega)$. Repeating this calculation for all choices of D_1 and D_2 that differ in the presence of one tuple shows that $\mathfrak{M}$ satisfies ϵ-differential privacy.

Case 2: We now show that a mechanism satisfying ϵ differential privacy also satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*)$. Choose an individual h_{j_1}, an arbitrary $t_{j_1} \in \mathcal{T}$ and an arbitrary $\theta = \{\pi_1, \ldots, \pi_N, f_1, \ldots, f_N\}$ of record-generating distributions. We use the notation $D \cup \{r_{j_1} = t_{j_1}\}$ to denote a dataset formed from D by adding the record r_{j_1} (belonging to individual h_{j_1}) with value t_{j_1} and $D \setminus \{r_{j_1} = t_{j_1}\}$ to denote the dataset formed by the removal from D of this record (this notation is only defined if D contains the record for h_{j_1} and the value of the record is t_{j_1}). Since $\mathfrak{M}$ satisfies ϵ-differential

privacy, we have:

$$
\begin{aligned}
& P(\mathfrak{M}(\mathfrak{Data}) = \omega) \mid \sigma_{(j_1,t_{j_1})}, \theta) \\
&= \int P(\mathfrak{Data} = D, \mathfrak{M}(D) = \omega \mid \sigma_{(j_1,t_{j_1})}, \theta)\, dD \\
&\leq e^{\epsilon} \int P\left(\mathfrak{Data} = D \wedge \mathfrak{M}(D \setminus \{r_{j_1} = t_{j_1}\}) = \omega \;\middle|\; \sigma_{(j_1,t_{j_1})}, \theta\right)\, dD \\
&\quad \text{(by definition of differential privacy)} \\
&= e^{\epsilon} \int P\left(\mathfrak{Data} = D \setminus \{r_{j_1} = t_{j_1}\} \wedge \mathfrak{M}(D \setminus \{r_{j_1} = t_{j_1}\}) = \omega \;\middle|\; \neg\sigma_{j_1}, \theta\right)\, dD \\
&\quad \Big(\text{because under the probabilistic model (Equation 8), we have equality} \\
&\qquad \text{between } P(\mathfrak{Data} = D \mid \sigma_{(j_1,t_{j_1})}, \theta) \text{ and } P(\mathfrak{Data} = D \setminus \{r_{j_1} = t_{j_1}\} \mid \neg\sigma_j, \theta)\Big) \\
&= e^{\epsilon} \int P(\mathfrak{Data} = D, \mathfrak{M}(D) = \omega \mid \neg\sigma_{j_1}, \theta)\, dD \\
&= e^{\epsilon} P(\mathfrak{M}(\mathfrak{Data}) = \omega) \mid \neg\sigma_{j_1}, \theta)
\end{aligned}
$$

Similarly, $P(\mathfrak{M}(\mathfrak{Data}) = \omega) \mid \neg\sigma_{j_1}, \theta) \leq e^{\epsilon} P(\mathfrak{M}(\mathfrak{Data}) = \omega) \mid \sigma_{(j_1,t_{j_1})}, \theta)$ also holds. Since j_1, t_{j_1}, and the distribution $\theta = \{\pi_1, \ldots, \pi_N, f_1, \ldots, f_N\}$ were arbitrary the same must hold for all $(\sigma_{i,t},\ \neg\sigma_i) \in \mathbb{S}_{\text{pairs}}$ and so $\mathfrak{M}$ must also satisfy ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*)$. □

D. PROOF OF THEOREM 6.2

We first need a technical lemma. We use the notation $D \cup \{r_j = t_j\}$ to denote a dataset formed from D by adding the record r_j (belonging to individual h_j) with value t_j

LEMMA D.1. *Let $\mathbb{D}^*$ be the the set of all probability distributions having the form given in Equation 8. Then $\mathbb{D}^*$ is precisely the set of distributions for which the conditional probabilities $P(\mathfrak{Data} = D \mid \neg\sigma_i)$ and $P(\mathfrak{Data} = D \cup \{r_j = t_j\} \mid \sigma_{(i,t)})$ are equal for all $D \in \mathcal{I}$, $i = 1, \ldots, N$ and $t \in \mathcal{T}$.*

PROOF. Clearly every $\theta \in \mathbb{D}^*$ has this property. All that is left is to prove the other direction.

Step 1:

Suppose there exists a θ with this property. We must prove that $\theta \in \mathbb{D}^*$. First, a preliminary calculation.

$$
\begin{aligned}
& P(\mathfrak{Data} = D \cup \{r_{j_i} = t_{j_i}\} \mid \theta) \\
&= P(\mathfrak{Data} = D \cup \{r_{j_i} = t_{j_i}\} \mid \sigma_{(r_{j_i},t_{j_i})}\theta) P(\sigma_{(r_{j_i},t_{j_i})} \mid \theta) \\
&= P(\mathfrak{Data} = D \mid \neg\sigma_{j_i}, \theta) P(\sigma_{(r_{j_i},t_{j_i})} \mid \theta) \\
&\quad \text{(by equivalence of the conditional probabilities)} \\
&= P(\mathfrak{Data} = D \wedge \neg\sigma_{j_1} \mid \theta) \frac{P(\sigma_{(r_{j_i},t_{j_i})} \mid \theta)}{P(\neg\sigma_{j_i} \mid \theta)} \\
&\quad \text{(whenever } P(\neg\sigma_{j_i} \mid \theta) > 0\text{)} \\
&= P(\mathfrak{Data} = D \mid \theta) \frac{P(\sigma_{(r_{j_i},t_{j_i})} \mid \theta)}{P(\neg\sigma_{j_i} \mid \theta)} \\
&\quad \text{(whenever } D \text{ contains no record about } h_{j_1} \text{ and } P(\neg\sigma_{j_i} \mid \theta) > 0\text{)}
\end{aligned}
$$

Step 2:
Now suppose that $P(\neg\sigma_j \mid \theta) > 0$ for all $j = 1 \ldots N$ (recall, N is the size of the population and n is a random variable denoting dataset size). We will later get rid of this assumption with a limiting argument. Then if $j_1, \ldots, j_n$ are distinct we have from Step 1:

$$P\left(\boldsymbol{\mathfrak{Data}} = \bigcup_{i=1}^{n} \{r_{j_i} = t_{j_i}\} \;\middle|\; \theta\right) = P(\boldsymbol{\mathfrak{Data}} = \emptyset \mid \theta) \prod_{i=1}^{n} \frac{P(\sigma_{(r_{j_i}, t_{j_i})} \mid \theta)}{P(\neg\sigma_{j_i} \mid \theta)}$$

$$= P(\boldsymbol{\mathfrak{Data}} = \emptyset \mid \theta) \frac{\prod_{i=1}^{n} P(\sigma_{(r_{j_i}, t_{j_i})} \mid \theta) \prod_{i \notin \{j_1, \ldots, j_n\}} P(\neg\sigma_{j_i} \mid \theta)}{\prod_{i=1}^{N} P(\neg\sigma_{j_i} \mid \theta)}$$

Integrating over all datasets (i.e. adding over all choices of n, records to place in the dataset, and values of those records) and solving for $P(\boldsymbol{\mathfrak{Data}} = \emptyset \mid \theta)$, we get

$$P(\boldsymbol{\mathfrak{Data}} = \emptyset \mid \theta) = \prod_{i=1}^{N} P(\neg\sigma_{j_i} \mid \theta)$$

and so we have

$$P\left(\boldsymbol{\mathfrak{Data}} = \bigcup_{i=1}^{n} \{r_{j_i} = t_{j_i}\} \;\middle|\; \theta\right) = \prod_{i=1}^{n} P(\sigma_{(r_{j_i}, t_{j_i})} \mid \theta) \prod_{i \notin \{j_1, \ldots, j_n\}} P(\neg\sigma_{j_i} \mid \theta)$$

$$= \prod_{i=1}^{n} P(\sigma_{(r_{j_i}, t_{j_i})} \mid \sigma_{j_i}) P(\sigma_{j_i} \mid \theta) \prod_{i \notin \{j_1, \ldots, j_n\}} P(\neg\sigma_{j_i} \mid \theta)$$

Setting $\pi_i = \sigma_i$ and $f_i(t) = P(\sigma_{r_i, t})$ for all $i = 1, \ldots, N$ and $t \in \mathcal{T}$ we see that $P(\boldsymbol{\mathfrak{Data}} \mid \theta)$ has the form given in Equation 8 and so $\theta \in \mathbb{D}^*$.
Step 3:
In the case where some of the $P(\neg\sigma_j) = 0$, we simply apply the previous result and take the limit as $P(\neg\sigma_j) \to 0$. Thus again we see that $\theta \in \mathbb{D}^*$. □

THEOREM 6.2. *Let $\mathbb{S}$ and $\mathbb{S}_{pairs}$ be defined as in Equations 6 and 7. Let $\mathbb{D}^*$ be the set of all distributions of the form specified in Equation 8. If we choose the data evolution scenarios $\mathbb{D}_{other}$ such that $\mathbb{D}_{other} \not\subseteq \mathbb{D}^*$ then ϵ-differential privacy is not equivalent to ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}_{other})$ (i.e. with the same ϵ-parameter) and hence does not bound the odds-ratio to the interval $[e^{-\epsilon}, e^{\epsilon}]$.*

PROOF. We will use the notation $r_i \in D$ to denote the statement that the record r_i belonging to individual h_i is in dataset D. We will also use the notation $\{r_i = t\} \in D$ to mean that the record r_i has value t in the dataset D. Hence we also use the notation $D \cup \{r_i = t\}$ to be the dataset that results from adding record r_i with value t to D.

Since $\mathbb{D} \not\subseteq \mathbb{D}^*$, by Lemma D.1 there exists a probability distribution $\theta \in \mathbb{D}$, an individual h_i and a tuple value t such that for some $D' \in \mathcal{I}$, the probabilities $P(\boldsymbol{\mathfrak{Data}} = D' \mid \neg\sigma_i, \theta)$ and $P(\boldsymbol{\mathfrak{Data}} = D' \cup \{r_i = t\} \mid \sigma_{(i,t)}, \theta)$ are different (i.e. the probability of the "rest" of the data is affected by whether or not the record belonging to h_i was collected by the data publisher) – in contrast, those conditional distributions are the same precisely for those $\theta^* \in \mathbb{D}^*$.

Thus there exists a collection $A \subset \mathcal{I}$ of datasets containing no record for h_i such that[13]:

$$P(\mathfrak{Data} \in A \mid \neg\sigma_i, \theta) > P\left(\mathfrak{Data} \in \left\{D \cup \{r_i = t\} \quad : \ D \in A\right\} \ \middle|\ \sigma_{(i,t)}, \theta\right) \geq 0$$

Thus define:

$$C = \left\{D \cup \{r_i = t\} \quad : \ D \in A\right\}$$
$$B = \left\{D \cup \{r_i = t^*\} \quad : \ D \in A, t^* \in \mathcal{T}\right\}$$

Note that by construction

$$C \subseteq B$$
$$P(\mathfrak{Data} \in A \mid \neg\sigma_i, \theta) > P(\mathfrak{Data} \in C \mid \sigma_{(i,t)}, \theta) \geq 0$$

Define the mechanism $\mathfrak{M}$ as follows:

$$P(\mathfrak{M}(D) = 1) = \begin{cases} \frac{e^\epsilon}{1+e^\epsilon} & \text{if } D \in A \\ \frac{1}{1+e^\epsilon} & \text{if } D \notin A \wedge r_i \notin \text{records}(D) \\ \frac{1}{1+e^\epsilon} & \text{if } D \in B \\ \frac{1}{e^\epsilon(1+e^\epsilon)} & \text{if } D \notin B \wedge r_i \in \text{records}(D) \end{cases}$$

$$P(\mathfrak{M}(D) = 2) = \begin{cases} \frac{1}{1+e^\epsilon} & \text{if } D \in A \\ \frac{e^\epsilon}{1+e^\epsilon} & \text{if } D \notin A \wedge r_i \notin \text{records}(D) \\ \frac{e^\epsilon}{1+e^\epsilon} & \text{if } D \in B \\ \frac{e^\epsilon(1+e^\epsilon)-1}{e^\epsilon(1+e^\epsilon)} & \text{if } D \notin B \wedge r_i \in \text{records}(D) \end{cases}$$

Note that $r_i \in \text{records}(D)$ means the record for individual h_i is in D but does not specify a value for that record. Also note that all 4 conditions on D used to define $\mathfrak{M}$ are mutually exclusive and exhaustive (the first two cover datasets with no information on h_i and the last two cover datasets with information about h_i).

Step 1: Prove that $\mathfrak{M}$ satisfies ϵ-differential privacy. First note that if D_1 and D_2 are datasets such that $D_1 \in A$ and $D_2 \notin B \wedge r_i \in \text{records}(D_2)$ then D_1 and D_2 cannot differ by the addition or deletion of exactly one tuple. The reason is that $r_i \in \text{records}(D_2)$ but $r_i \notin \text{records}(D_1)$ (because A is a set of datasets with no information about individual h_i) and so D_2 must be formed from D_1 by adding a record about individual h_i plus possibly some additional/deletion operations. By construction, B is precisely the collection of datasets that we can obtain by adding a record about h_i to a dataset in A with no further addition/deletion operations. Therefore constructing D_2 from D_1 involves adding a record about h_i and at least one other addition/deletion operation and so D_1 and D_2 cannot be neighbors.

The fact that $\mathfrak{M}$ satisfies ϵ-differential privacy follows from: $\frac{e^\epsilon}{1+e^\epsilon} / \frac{1}{1+e^\epsilon} \in [e^{-\epsilon}, e^\epsilon]$ and $\frac{1}{1+e^\epsilon} / \frac{1}{e^\epsilon(1+e^\epsilon)} \in [e^{-\epsilon}, e^\epsilon]$ and

$$\left(\frac{e^\epsilon(1+e^\epsilon)-1}{e^\epsilon(1+e^\epsilon)}\right) / \left(\frac{e^\epsilon}{1+e^\epsilon}\right) = \frac{e^\epsilon(e^\epsilon+1)-1}{e^{2\epsilon}} = \frac{e^{2\epsilon}+e^\epsilon-1}{e^{2\epsilon}} = 1 + \frac{e^\epsilon-1}{e^{2\epsilon}} \in [e^{-\epsilon}, \epsilon]$$

since $e^{-\epsilon} < 1 \leq 1 + \frac{e^\epsilon-1}{e^{2\epsilon}} < 1 + \frac{e^{2\epsilon}(e^\epsilon-1)}{e^{2\epsilon}} = e^\epsilon$

[13]One choice for A is D' or the collection all datasets (except for D') not containing a record for h_i – the specific choice depends on whether $P(\mathfrak{Data} = D' \mid \neg\sigma_i, \theta) > P(\mathfrak{Data} = D' \cup \{r_i = t\} \mid \sigma_{(i,t)}, \theta)$ or $P(\mathfrak{Data} = D' \mid \neg\sigma_i, \theta) < P(\mathfrak{Data} = D' \cup \{r_i = t\} \mid \sigma_{(i,t)}, \theta)$.

Step 2: show that $\mathfrak{M}$ does not satisfy $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$. Recall that we chose a θ such that $P(\boldsymbol{\mathfrak{Data}} \in A \mid \neg\sigma_i, \theta) > P(\boldsymbol{\mathfrak{Data}} \in C \mid \sigma_{(i,t)} \mid \theta)$. Also recall that datasets in A do not contain records about individual h_i.

$$\begin{aligned}
P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = 1 \mid \neg\sigma_i) &= \int P(\mathfrak{M}(D) = 1) P(\boldsymbol{\mathfrak{Data}} = D \mid \neg\sigma_i, \theta)\, dD \\
&= \int_{\left\{D \,:\, \substack{D \notin A \\ r_i \notin \text{records}(D)}\right\}} P(\mathfrak{M}(D) = 1) P(\boldsymbol{\mathfrak{Data}} = D \mid \neg\sigma_i, \theta)\, dD \\
&\quad + \int_A P(\mathfrak{M}(D) = 1) P(\boldsymbol{\mathfrak{Data}} = D \mid \neg\sigma_i, \theta)\, dD \\
&\quad \text{(since those integrals cover the datasets without a record } r_i\text{)} \\
&= \frac{1}{1+e^\epsilon}(1 - P(A \mid \neg\sigma_i, \theta)) + \frac{e^\epsilon}{1+e^\epsilon} P(A \mid \neg\sigma_i, \theta) \\
&= e^\epsilon \left[\frac{1}{e^\epsilon(1+e^\epsilon)}(1 - P(A \mid \neg\sigma_i, \theta)) + \frac{1}{1+e^\epsilon} P(A \mid \neg\sigma_i, \theta)\right] \\
&= e^\epsilon \left[\frac{1}{e^\epsilon(1+e^\epsilon)} + \frac{e^\epsilon - 1}{e^\epsilon(1+e^\epsilon)} P(A \mid \neg\sigma_i, \theta)\right]
\end{aligned}$$

Similarly,

$$\begin{aligned}
&P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = 1 \mid \sigma_{(i,t)} \mid \theta) \\
&= \int P(\mathfrak{M}_1(D) = 1) P(\boldsymbol{\mathfrak{Data}} = D \mid \sigma_{(i,t)}, \theta)\, dD \\
&= \int_{\left\{D \,:\, \substack{D \notin C \\ \{r_i, t\} \in D}\right\}} P(\mathfrak{M}(D) = 1) P(\boldsymbol{\mathfrak{Data}} = D \mid \sigma_{(i,t)}, \theta)\, dD \\
&\quad + \int_C P(\mathfrak{M}(D) = 1) P(\boldsymbol{\mathfrak{Data}} = D \mid \sigma_{(i,t)}, \theta)\, dD \\
&\quad \text{(since those integrals cover the datasets containing record } r_i \text{ and where } r_i = t\text{)} \\
&= \frac{1}{e^\epsilon(1+e^\epsilon)}(1 - P(C \mid \sigma_{(i,t)}, \theta)) + \frac{1}{1+e^\epsilon} P(C \mid \sigma_{(i,t)}, \theta) \\
&\quad \text{(Since } C \subseteq B \text{ and } [D \notin C \wedge \{r_i, t\} \in D] \Rightarrow [D \notin B \wedge r_i \in \text{records}(D)]\text{)} \\
&= \frac{1}{e^\epsilon(1+e^\epsilon)} + \frac{e^\epsilon - 1}{e^\epsilon(1+e^\epsilon)} P(C \mid \sigma_{(i,t)}, \theta)
\end{aligned}$$

Since we had chosen A and C such that $P(\boldsymbol{\mathfrak{Data}} \in A \mid \neg\sigma_i, \theta) > P(\boldsymbol{\mathfrak{Data}} \in C \mid \sigma_{(i,t)}, \theta)$ then these calculations show that $P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = 1 \mid \neg\sigma_i, \theta) > e^\epsilon P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = 1 \mid \sigma_{(i,t)}, \theta)$ and therefore the ϵ-differentially private algorithm $\mathfrak{M}$ does not satisfy ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$. □

E. PROOF OF THEOREM 7.1

THEOREM 7.1. *Define the set of potential secrets as in Equation 12 and discriminative pairs as in Equation 13. If the set of evolution scenarios $\mathbb{D}$ consists only of the binomial distribution with parameters q and n then the mechanism in Algorithm 1 satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*)$.*

PROOF. Let $\mathfrak{M}_{q,n}$ denote this mechanism. Clearly, for any $x \in \text{range}(\mathfrak{M})$ and the single $\theta \in \mathbb{D}$,

$$\begin{aligned}
P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta) &= \sum_{k=0}^{n-1} \binom{n-1}{k} q^k (1-q)^{n-k-1} P(\mathfrak{M}_{q,n}(k+1) = x) \\
P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,0)}, \theta) &= \sum_{k=0}^{n-1} \binom{n-1}{k} q^k (1-q)^{n-k-1} P(\mathfrak{M}_{q,n}(k) = x) \\
P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \neg\sigma_i, \theta) &= q \sum_{k=0}^{n-1} \binom{n-1}{k} q^k (1-q)^{n-k-1} P(\mathfrak{M}_{q,n}(k+1) = x) \\
&\quad + (1-q) \sum_{k=0}^{n-1} \binom{n-1}{k} q^k (1-q)^{n-k-1} P(\mathfrak{M}_{q,n}(k) = x) \\
&= qP(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta) \\
&\quad + (1-q) P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,0)}, \theta)
\end{aligned}$$

Therefore

$$\begin{aligned}
&\left(e^{-\epsilon} \leq \frac{P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta)}{P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,0)}, \theta)} \leq e^{\epsilon}\right) \\
\Rightarrow\ &\left(e^{-\epsilon} \leq \frac{(1-q)P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta)}{(1-q)P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,0)}, \theta)} \leq e^{\epsilon}\right) \\
\Rightarrow\ &\left(e^{-\epsilon} \leq \frac{(1-q)P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta) + qP(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta)}{(1-q)P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,0)}, \theta) + qP(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta)} \leq e^{\epsilon}\right) \\
&\quad \text{since addition of the same term to the numerator and} \\
&\quad \text{denominator makes the fraction closer to } 1, \\
\Leftrightarrow\ &\left(e^{-\epsilon} \leq \frac{P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta)}{P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \neg\sigma_i, \theta)} \leq e^{\epsilon}\right)
\end{aligned}$$

so we just need to show that

$$e^{-\epsilon} \leq \frac{P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,1)}, \theta)}{P(\mathfrak{M}_{q,n}(\boldsymbol{\mathfrak{Data}}) = x \mid \sigma_{(i,0)}, \theta)} \leq e^{\epsilon} \tag{27}$$

Now, when x is a noisy count then clearly for all k,

$$e^{-\epsilon} \leq \frac{P(\mathfrak{M}_{q,n}(k) = x \mid \sigma_{(i,1)})}{P(\mathfrak{M}_{q,n}(k) = x \mid \sigma_{(i,0)})} \leq e^{\epsilon}$$

and therefore Equation 27 follows.

In the case where the mechanism asserts that k is the true count, first note that since $1 \leq k_{\text{lo}}, k_{\text{hi}} \leq n-1$ we have that the mechanism never outputs a true answer if $k = 0$ and $k = n$. This also implies that when $q = 0$ or $q = 1$, the mechanism never outputs a true answer (since $k = 0$ and n respectively in those cases). For $q \neq 0, 1$ and

$k \neq 0, n$, we have:

$$\begin{aligned}\frac{P(\mathfrak{M}_{q,n}(\mathfrak{Data}) = \text{“true count k”} \mid \sigma_{(i,1)}, \theta)}{P(\mathfrak{M}_{q,n}(\mathfrak{Data}) = \text{“true count k”} \mid \sigma_{(i,0)}, \theta)} &= \frac{\binom{n-1}{k-1} q^{k-1}(1-q)^{n-k}}{\binom{n-1}{k} q^{k}(1-q)^{n-k-1}} \\ &= \frac{k(1-q)}{(n-k)q}\end{aligned}$$

so

$$\begin{aligned}\left(e^{-\epsilon} \leq \frac{k}{n-k}\frac{1-q}{q} \leq e^{\epsilon}\right) &\Leftrightarrow \left(\frac{q}{1-q}e^{-\epsilon} \leq \frac{k}{n-k} \leq \frac{q}{1-q}e^{\epsilon}\right) \\ &\Leftrightarrow \left(n\frac{\frac{q}{1-q}e^{-\epsilon}}{1+\frac{q}{1-q}e^{-\epsilon}} \leq k \leq n\frac{\frac{q}{1-q}e^{\epsilon}}{1+\frac{q}{1-q}e^{\epsilon}}\right) \\ &\Leftrightarrow \left(n\frac{qe^{-\epsilon}}{qe^{-\epsilon}+1-q} \leq k \leq n\frac{qe^{\epsilon}}{qe^{\epsilon}+1-q}\right) \\ &\Leftrightarrow \left(\left\lceil n\frac{qe^{-\epsilon}}{qe^{-\epsilon}+1-q}\right\rceil \leq k \leq \left\lfloor n\frac{qe^{\epsilon}}{qe^{\epsilon}+1-q}\right\rfloor\right)\end{aligned}$$

The last implication follows since k is an integer. The upper and lower bounds on k are k_{hi} and k_{lo}, respectively, from Algorithm 1 and therefore this mechanism satisfies this instantiation of the pufferfish framework. □

F. PROOF OF THEOREM 7.2

THEOREM 7.2. *Define the set of potential secrets as in Equation 12 and discriminative pairs as in Equation 13. If the set of evolution scenarios $\mathbb{D}$ consists only of the multinomial distribution with parameters $\vec{q}$ and n then the mechanism in Algorithm 2 satisfies 2ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*)$.*

PROOF. Let $\mathfrak{M}_{\vec{q},n}$ denote the mechanism. Let $\tilde{H}$ be the sanitized histogram that is output. Let T denote the index of all of the cells for which the true count was released (i.e. $T = \left\{j \ : \ H[j] = \tilde{H}[j]\right\}$). The same proof as in Theorem 7.1 shows that

$$e^{-2\epsilon} \leq \frac{P(\mathfrak{M}_{\vec{q},n}(\mathfrak{Data}) = \tilde{H} \mid \sigma_{(i,\ell)}, \theta)}{P(\mathfrak{M}_{\vec{q},n}(\mathfrak{Data}) = \tilde{H} \mid \neg\sigma_i, \theta)} \leq e^{2\epsilon}$$

for all $\sigma_{(i,\ell)}$ where $\ell \in T$. Now consider $\ell \notin T$. Divide the original histogram H into two parts, H_T and $H_{\neg T}$, where H_T consists of the cells that were released deterministically and $H_{\neg T}$ consists of the cells for which noisy versions were released. Similarly, divide $\tilde{H}$ into $\tilde{H}_T$ and $\tilde{H}_{\neg T}$, noting that $\tilde{H}_{\neg T}$ are the noisy counts while $H_{\neg T}$ are the true counts of those cells.

For $\sigma_{(i,\ell)}$ where $\ell \notin T$,

$$\frac{P(\mathfrak{M}_{\vec{q},n}(\mathfrak{Data}) = \tilde{H} \mid \sigma_{(i,\ell)}, \theta)}{P(\mathfrak{M}_{\vec{q},n}(\mathfrak{Data}) = \tilde{H} \mid \neg\sigma_i, \theta)} = \frac{P(\tilde{H}_{\neg T} \mid \tilde{H}_T, \sigma_{(i,\ell)}, \theta)}{P(\tilde{H}_{\neg T} \mid \tilde{H}_T, \neg\sigma_i, \theta)} \times \frac{P(\tilde{H}_T \mid \sigma_{(i,\ell)}, \theta)}{P(\tilde{H}_T \mid \neg\sigma_i, \theta)} \tag{28}$$

Now, an argument essentially the same as the second part (analysis of the true count being output) of the proof of Theorem 7.1 (with $q' = \sum_{i \in T} q_i$) shows that

$$e^{-\epsilon} \leq \frac{P(\tilde{H}_T \mid \sigma_{(i,\ell)}, \theta)}{P(\tilde{H}_T \mid \neg\sigma_i, \theta)} \leq e^{-\epsilon} \tag{29}$$

Similarly, once $\tilde{H}_T$ (the true counts) are released, $P(\tilde{H}_{\neg T} \mid \tilde{H}_T, \sigma_{(i,\ell)}, \theta)$ is mathematically equivalent to $P(\mathfrak{M}_{\vec{q}',n'}(H_{\neg T}) = \tilde{H}_{\neg T} \mid \sigma_{(i,\ell)}, \theta')$ where $n' = \sum_{\ell \notin T} H[\ell]$ (the sum of the cells for which exact counts were not released), $\vec{q}' = \left(\frac{q_\ell}{\sum_{j \notin T} q_j} \ : \ \ell \notin T \right)$ (the restriction of $\vec{q}$ to cells whose deterministic counts were not released), and θ' is the multinomial distribution with parameters $\vec{q}'$ and n'. Similarly, $P(\tilde{H}_{\neg T} \mid \tilde{H}_T, \neg\sigma_i, \theta)$ is mathematically equivalent to $P(\mathfrak{M}_{\vec{q}',n'}(H_{\neg T}) = \tilde{H}_{\neg T} \mid \sigma_{(i,\ell)}, \theta')$. Since all of the cells in $\tilde{H}_{\neg T}$ are noisy counts, the same proof as in the first part of this theorem shows that

$$e^{-\epsilon} \leq \frac{P(\tilde{H}_{\neg T} \mid \tilde{H}_T, \sigma_{(i,\ell)}, \theta)}{P(\tilde{H}_{\neg T} \mid \tilde{H}_T, \neg\sigma_i, \theta)} \leq e^{\epsilon} \tag{30}$$

Combining Equations 28, 29, 30, we get

$$e^{-2\epsilon} \leq \frac{P(\mathfrak{M}_{\vec{q},n}(\mathfrak{Data}) = \tilde{H} \mid \sigma_{(i,\ell)}, \theta)}{P(\mathfrak{M}_{\vec{q},n}(\mathfrak{Data}) = \tilde{H} \mid \neg\sigma_i, \theta)} \leq e^{2\epsilon}$$

and so the algorithm satisfies this instantiation of Pufferfish with parameter 2ϵ. □

G. PROOF OF THEOREM 7.3

To prove the theorem, we need the following claims.

CLAIM G.1. *Let q and q^* be arbitrary distinct numbers in the interval $(0, 1)$.*

(1) if $q < q^*$ then $(q^*)^k(1-q^*)^{n-k} \geq \lambda q^k(1-q)^{n-k}$ if and only if $k \geq n\frac{\log \frac{1-q}{1-q^*}}{\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right)} + \frac{\log \lambda}{\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right)}$

(2) If $q > q^*$ then $(q^*)^k(1-q^*)^{n-k} \geq \lambda q^k(1-q)^{n-k}$ if and only if $k \leq n\frac{\log \frac{1-q^*}{1-q}}{\log\left(\frac{q}{1-q}/\frac{q^*}{1-q^*}\right)} - \frac{\log \lambda}{\log\left(\frac{q}{1-q}/\frac{q^*}{1-q^*}\right)}$

PROOF.

$$\begin{aligned}
&(q^*)^k(1-q^*)^{n-k} \geq \lambda q^k(1-q)^{n-k} \\
&\Leftrightarrow\ k\log q^* + (n-k)\log(1-q^*) \geq \log\lambda + k\log q + (n-k)\log(1-q) \\
&\Leftrightarrow\ k\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right) \geq \log\lambda + n\log\frac{1-q}{1-q^*}
\end{aligned}$$

If $q^* > q$ then $\frac{q^*}{1-q^*}/\frac{q}{1-q} > 1$ and so its log is positive, and therefore

$$k \geq n\frac{\log\frac{1-q}{1-q^*}}{\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right)} + \frac{\log\lambda}{\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right)}$$

if $q > q^*$ then $\frac{q^*}{1-q^*}/\frac{q}{1-q} < 1$ and so its log is negative, and therefore

$$k \leq n\frac{\log\frac{1-q}{1-q^*}}{\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right)} + \frac{\log\lambda}{\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right)}$$

which is equivalent to

$$k \le n\frac{\log\frac{1-q^*}{1-q}}{\log\left(\frac{q}{1-q}/\frac{q^*}{1-q^*}\right)} - \frac{\log\lambda}{\log\left(\frac{q}{1-q}/\frac{q^*}{1-q^*}\right)}$$

□

CLAIM G.2. *Let q and q^* be arbitrary distinct numbers in the interval $(0,1)$ and let $\lambda \ge 1$.*

(1) if $q < q^*$ and $(q^*)^k(1-q^*)^{n-k} \ge \lambda q^k(1-q)^{n-k}$ then
  (a) $k/n \ge q$ and
  (b) $(q^*)^k(1-q^*)^{n-k} \ge \lambda(q')^k(1-q')^{n-k}$ for $q' \le q$.
(2) If $q > q^*$ and $(q^*)^k(1-q^*)^{n-k} \ge \lambda q^k(1-q)^{n-k}$ then
  (a) $k/n \le q$ and
  (b) $(q^*)^k(1-q^*)^{n-k} \ge \lambda(q')^k(1-q')^{n-k}$ for $q' \ge q$.

PROOF. To prove (1a), note that the lower bound on k from Claim G.1 is an increasing function of λ. When $k/n = q$ then clearly $\lambda \le 1$ (in order for $(q^*)^k(1-q^*)^{n-k} \ge \lambda q^k(1-q)^{n-k}$ to hold) so, when $\lambda \ge 1$ we must have $k/n \ge q$.

To prove (1b), we again consider the lower bound on k from Claim G.1 . We will show that the lower bound on k is an increasing function of q. Thus when q is replaced by q', k is still greater than the resulting lower bound and so by the if-and-only-if condition of Claim G.1 we would have $(q^*)^k(1-q^*)^{n-k} \ge \lambda(q')^k(1-q')^{n-k}$. Thus to show the lower bound is an increasing of q, we first note that $\frac{\log\lambda}{\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right)}$ is nondecreasing in q since $\lambda \ge 1$. Meanwhile,

$$\frac{\log\frac{1-q}{1-q^*}}{\log\left(\frac{q^*}{1-q^*}/\frac{q}{1-q}\right)} = \frac{\log\frac{1-q}{1-q^*}}{\log\left(\frac{1-q}{1-q^*}/\frac{q}{q^*}\right)} = \frac{\log\frac{1-q}{1-q^*}}{\log\left(\frac{1-q}{1-q^*}\right) - \log\left(\frac{q}{q^*}\right)} = \frac{1}{1-\frac{\log(q/q^*)}{\log((1-q)/(1-q^*))}}$$

which is clearly increasing in q hence the lower bound on k is increasing in q.

Parts (2a) and (2b) are proved similarly. □

CLAIM G.3. *Let $q_{lo} = \frac{e^{-\epsilon/3}q^*}{e^{-\epsilon/3}q^*+1-q^*}$ and $q_{hi} = \frac{e^{\epsilon/3}q^*}{e^{\epsilon/3}q^*+1-q^*}$. Let $\lambda \ge 1$, let k_1 be the smallest value of k for which $(q^*)^k(1-q^*)^{n-k} \ge \lambda q_{lo}^k(1-q_{lo})^{n-k}$ and k_2 be the largest value of k for which $(q^*)^k(1-q^*)^{n-k} \ge \lambda q_{hi}^k(1-q_{hi})^{n-k}$. Then for large enough n, the interval $[k_1,k_2]$ is nonempty and $q_{lo} \le k_1/n \le q^* \le k_2/n \le q_{hi}$.*

PROOF. First note that $q_{\text{lo}} \le q^* \le q_{\text{hi}}$ so that by Claim G.1, k_1 and k_2 are well-defined.

Define the real number k_0 by the relation $(q^*)^{k_0}(1-q^*)^{n-k_0} = q_{\text{lo}}^{k_0}(1-q_{\text{lo}})^{n-k_0}$ (i.e. use both parts of Claim G.1 with $\lambda = 1$). By Claim G.1, k_0/n and k_1/n are arbitrarily close to each other as $n \to \infty$. By two applications of Claim G.2, we have $q_{\text{lo}} \le k_0/n \le q^*$ and therefore $q_{\text{lo}} \le k_1/n \le q^*$. Similar reasoning gives $q^* \le k_1/n \le q_{\text{hi}}$. The claim now follows. □

CLAIM G.4. *Let $q_{lo} = \frac{e^{-\epsilon/3}q^*}{e^{-\epsilon/3}q^*+1-q^*}$ and $q_{hi} = \frac{e^{\epsilon/3}q^*}{e^{\epsilon/3}q^*+1-q^*}$ as in Algorithm 3. Then if $q \in [q_{lo}, q_{hi}]$ and $k/n \in [q_{lo}, q_{hi}]$ then*

$$e^{-2\epsilon/3} \le \frac{n-k}{k}\frac{q}{1-q} \le e^{2\epsilon/3}$$

PROOF.

$$\frac{n-k}{k}\frac{q}{1-q} \;=\; \frac{1-k/n}{k/n}\frac{q}{1-q} \le \frac{1-q_{\text{lo}}}{q_{\text{lo}}}\frac{q_{\text{hi}}}{1-q_{\text{hi}}} = \frac{1-q^*}{e^{-\epsilon/3}q^*}\frac{e^{\epsilon/3}q^*}{1-q^*} = e^{2\epsilon/3}$$

The other inequality is proved similarly. □

CLAIM G.5. *Let q^* and α be as in Algorithm 3 (in particular, $\alpha < 0.5$) and set λ, q_{lo}, q_{hi}, k_{lo}, and k_{hi} as in Algorithm 3. If the interval $[k_{lo}, k_{hi}]$ is nonempty (which, by Claim G.3, happens when n is large enough), then for any $q \notin [q_{lo}, q_{hi}]$ and any $k \in [k_{lo}, k_{hi}]$, we have*

$$e^{-\epsilon} \le \frac{\alpha\binom{n-1}{k}(q^*)^k(1-q^*)^{n-k-1} + (1-\alpha)\binom{n-1}{k}q^k(1-q)^{n-k-1}}{\alpha\binom{n-1}{k-1}(q^*)^{k-1}(1-q^*)^{n-k} + (1-\alpha)\binom{n-1}{k-1}q^{k-1}(1-q)^{n-k}} \le e^{\epsilon}$$

PROOF. First note that $\lambda > 1$ because $\alpha < 0.5$.

First note that the setting of $k \ge k_{\text{lo}}$ ensures that $(q^*)^k(1-q^*)^{n-k-1} \ge \lambda q^k(1-q)^{n-k-1}$ and $(q^*)^{k-1}(1-q^*)^{n-k} \ge \lambda q^{k-1}(1-q)^{n-k}$ for $q = q_{\text{lo}}$ (by Claim G.1) and this holds for $q \le q_{\text{lo}}$ by Claim G.2 (and the fact that $q_{\text{lo}} \le q^*$). Similarly, setting $k \le k_{\text{hi}}$ ensures that $(q^*)^k(1-q^*)^{n-k-1} \ge \lambda q^k(1-q)^{n-k-1}$ and $(q^*)^{k-1}(1-q^*)^{n-k} \ge \lambda q^{k-1}(1-q)^{n-k}$ for $q \ge q_{\text{hi}}$.

Thus for any $q \notin [q_{\text{lo}}, q_{\text{hi}}]$ and any $k \in [k_{\text{lo}}, k_{\text{hi}}]$ we have $(q^*)^k(1-q^*)^{n-k-1} \ge \lambda q^k(1-q)^{n-k-1}$ and $(q^*)^{k-1}(1-q^*)^{n-k} \ge \lambda q^{k-1}(1-q)^{n-k}$. Therefore

$$\begin{aligned}
&\frac{\alpha\binom{n-1}{k}(q^*)^k(1-q^*)^{n-k-1} + (1-\alpha)\binom{n-1}{k}q^k(1-q)^{n-k-1}}{\alpha\binom{n-1}{k-1}(q^*)^{k-1}(1-q^*)^{n-k} + (1-\alpha)\binom{n-1}{k-1}q^{k-1}(1-q)^{n-k}} \\
&\le \left(\frac{n-k}{k}\right)\frac{\alpha(q^*)^k(1-q^*)^{n-k-1} + (1-\alpha)\lambda^{-1}(q^*)^k(1-q^*)^{n-k-1}}{\alpha(q^*)^{k-1}(1-q^*)^{n-k}} \\
&= \frac{n-k}{k}\frac{q^*}{1-q^*}\frac{\alpha + (1-\alpha)\lambda^{-1}}{\alpha} \\
&= \frac{n-k}{k}\frac{q^*}{1-q^*}\frac{\alpha + \alpha(e^{\epsilon/3}-1)}{\alpha} \quad \text{(by definition of } \lambda) \\
&\le e^{2\epsilon/3}e^{\epsilon/3} \quad \text{(by Claim G.4)} \\
&= e^{\epsilon}
\end{aligned}$$

$$\begin{aligned}
&\frac{\alpha\binom{n-1}{k}(q^*)^k(1-q^*)^{n-k-1} + (1-\alpha)\binom{n-1}{k}q^k(1-q)^{n-k-1}}{\alpha\binom{n-1}{k-1}(q^*)^{k-1}(1-q^*)^{n-k} + (1-\alpha)\binom{n-1}{k-1}q^{k-1}(1-q)^{n-k}} \\
&\ge \left(\frac{n-k}{k}\right)\frac{\alpha(q^*)^k(1-q^*)^{n-k-1}}{\alpha\binom{n-1}{k-1}(q^*)^{k-1}(1-q^*)^{n-k} + (1-\alpha)\lambda^{-1}(q^*)^{k-1}(1-q^*)^{n-k}} \\
&= \frac{n-k}{k}\frac{q^*}{1-q^*}\frac{\alpha}{\alpha + (1-\alpha)\lambda^{-1}} \\
&= \frac{n-k}{k}\frac{q^*}{1-q^*}\frac{\alpha}{\alpha + \alpha(e^{\epsilon/3}-1)} \quad \text{(by definition of } \lambda) \\
&\ge e^{-2\epsilon/3}e^{-\epsilon/3} \quad \text{(by Claim G.4)} \\
&= e^{-\epsilon}
\end{aligned}$$

□

Now we are ready to prove our main result.

THEOREM 7.3. *Algorithm 3 satisfies ϵ-hedging privacy and Algorithm 4 satisfies 2ϵ-hedging privacy.*

PROOF. The proof proceeds in essentially the same way as the proofs of Theorems 7.1 (proof of correctness for the single count query in single prior privacy) and 7.2 (proof of correctness for the histogram in single prior privacy). In particular, once the correctness of Algorithm 3 (single counting query for hedging privacy) is established, the proof of correctness of Algorithm 4 is essentially the same as the proof of Theorem 7.2. Thus we show only the proof of correctness for Algorithm 3.

As with the proof of Theorem 7.1, the main challenge is to bound the privacy leakage when a true count is released. After that, the rest of the proof is the same as in Theorem 7.1. Thus we must show that for $k \in [k_{\text{lo}}, k_{\text{hi}}]$ (the only possible k for which a true count can possibly be released),

$$e^{-\epsilon} \leq \frac{\alpha\binom{n-1}{k}(q^*)^k(1-q^*)^{n-k-1} + (1-\alpha)\binom{n-1}{k}q^k(1-q)^{n-k-1}}{\alpha\binom{n-1}{k-1}(q^*)^{k-1}(1-q^*)^{n-k} + (1-\alpha)\binom{n-1}{k-1}q^{k-1}(1-q)^{n-k}} \leq e^{\epsilon}$$

for all $q \in [0,1]$.

When $q \notin [q_{\text{lo}}, q_{\text{hi}}]$, this result was established by Claim G.5. When $q \in [q_{\text{lo}}, q_{\text{hi}}]$ we observe that

$$\begin{aligned}
e^{-\epsilon} &\leq \min\left(\frac{n-k}{k}\frac{q^*}{1-q^*} \quad , \quad \frac{n-k}{k}\frac{q}{1-q}\right) \quad \text{(by Claim G.4 since } q \text{ and } q^* \text{ are in } [q_{\text{lo}}, q_{\text{hi}}]\text{)} \\
&= \min\left(\frac{\binom{n-1}{k}(q^*)^k(1-q^*)^{n-k-1}}{\binom{n-1}{k-1}(q^*)^{k-1}(1-q^*)^{n-k}} \quad , \quad \frac{\binom{n-1}{k}q^k(1-q)^{n-k-1}}{\binom{n-1}{k-1}q^{k-1}(1-q)^{n-k}}\right) \\
&\leq \frac{\alpha\binom{n-1}{k}(q^*)^k(1-q^*)^{n-k-1} + (1-\alpha)\binom{n-1}{k}q^k(1-q)^{n-k-1}}{\alpha\binom{n-1}{k-1}(q^*)^{k-1}(1-q^*)^{n-k} + (1-\alpha)\binom{n-1}{k-1}q^{k-1}(1-q)^{n-k}} \\
&\leq \max\left(\frac{\binom{n-1}{k}(q^*)^k(1-q^*)^{n-k-1}}{\binom{n-1}{k-1}(q^*)^{k-1}(1-q^*)^{n-k}} \quad , \quad \frac{\binom{n-1}{k}q^k(1-q)^{n-k-1}}{\binom{n-1}{k-1}q^{k-1}(1-q)^{n-k}}\right) \\
&= \max\left(\frac{n-k}{k}\frac{q^*}{1-q^*} \quad , \quad \frac{n-k}{k}\frac{q}{1-q}\right) \\
&\leq e^{\epsilon} \quad \text{(by Claim G.4)}
\end{aligned}$$

and the theorem follows. □

H. PROOF OF LEMMA 8.1

LEMMA 8.1. *With $\mathbb{S}$ and $\mathbb{S}_{pairs}$ defined in Equations 16 and 17 let $\mathbb{D}$ be the set of all probability distributions having the form specified in Equation 18. The algorithm $\mathfrak{M}$ which returns $X + \sum_{i=1}^{n} t_i$ where X has density $\frac{\epsilon}{8k}e^{-\epsilon|x|/4k}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D})$.*

PROOF. We use the notation $f_i(t)$ to represent the probability (or density, as appropriate) that the record for individual h_i has value t and $f_i(t \in A)$ to represent the probability that t is in the set A. We also use the notation $f_i(t \mid A)$ to represent the conditional probability/density of t given that $t \in A$. When the f_i are not continuous, replace the corresponding integrals with summations. First consider individual h_1, real value

y, and $\theta = [f_1, \ldots, f_n]$ such that $f_1(t_1 \in [y-k, y+k)) \neq 0$ and $f_1(t_1 \in [y-k, y+3k)) \neq 0$,

$$
\begin{aligned}
& P\Big(\mathfrak{M}(\mathfrak{Data}) = \omega \mid t_1 \in [y-k,\ y+k), \theta\Big) \\
&= \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - \sum_{i=1}^n t_i\big|\big) \\ \times f_1(t_1 \mid t_1 \in [y-k, y+k)) f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_1 \cdots dt_n \\
&\geq \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - y - \sum_{i=2}^n t_i\big| - \frac{\epsilon}{4k}|y - t_1|\big) \\ \times f_1(t_1 \mid t_1 \in [y-k, y+k)) f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_1 \cdots dt_n \\
&\quad \text{(because: } -|A - t_1| = -|A - y + y - t_1| \geq -|A - y| - |y - t_1|\text{)} \\
&\geq \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - y - \sum_{i=2}^n t_i\big| - \frac{\epsilon}{4k} k\big) \\ \times f_1(t_1 \mid t_1 \in [y-k, y+k)) f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_1 \cdots dt_n \\
&\quad \text{(because we conditioned on } t_1 \in [y-k, y+k) \text{ so } |y - t_1| \leq k\text{)} \\
&= \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - y - \sum_{i=2}^n t_i\big| - \frac{\epsilon}{4}\big) \\ \times f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_2 \cdots dt_n \\
&\quad \text{(we cancel } k \text{ and the integrand no longer depends on } t_1\text{)} \\
&= e^{-\frac{\epsilon}{4}} \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - y - \sum_{i=2}^n t_i\big|\big) \\ \times f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_2 \cdots dt_n \qquad (31)
\end{aligned}
$$

Meanwhile

$$
\begin{aligned}
& P\Big(\mathfrak{M}(\mathfrak{Data}) = \omega \mid t_i \in [y+k,\ y+3k), \theta\Big) \\
&= \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - \sum_{i=1}^n t_i\big|\big) \\ \times f_1(t_1 \mid t_1 \in [y+k, y+3k)) f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_1 \cdots dt_n \\
&\leq \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - y - \sum_{i=2}^n t_i\big| + \frac{\epsilon}{4k}|y - t_1|\big) \\ \times f_1(t_1 \mid t_1 \in [y+k, y+3k)) f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_1 \cdots dt_n \\
&\quad \text{(because } |A - t_1| = |A - y\ +\ y - t_1| \geq |A - y| - |y - t_1| \\
&\quad \text{and so } -|A - t_1| \leq -|A - y| + |y - t_1|\text{)} \\
&\leq \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - y - \sum_{i=2}^n t_i\big| + \frac{\epsilon}{4k} 3k\big) \\ \times f_1(t_1 \mid t_1 \in [y+k, y+3k)) f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_1 \cdots dt_n \\
&\quad \text{(because we conditioned on } t_1 \in [y+k, y+3k) \text{ so } |y - t_1| \leq 3k\text{)} \\
&= \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - y - \sum_{i=2}^n t_i\big| + \frac{3\epsilon}{4}\big) \\ \times f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_2 \cdots dt_n \\
&\quad \text{(we cancel } k \text{ and the integrand no longer depends on } t_1\text{)} \\
&= e^{\frac{3\epsilon}{4}} \int \cdots \int \begin{pmatrix} \frac{\epsilon}{8k} \exp\big(-\frac{\epsilon}{4k}\big|\omega - y - \sum_{i=2}^n t_i\big|\big) \\ \times f_2(t_2) f_3(t_3) \cdots f_n(t_n) \end{pmatrix} dt_2 \cdots dt_n \qquad (32)
\end{aligned}
$$

Comparing equations 31 and 32 we see that the only difference between them is the constant multiplier outside the integral. Thus dividing we get:

$$P\Big(\mathfrak{M}(\mathfrak{Data}) = \omega \mid t_1 \in [y+k,\ y+3k), \theta\Big) \ \leq\ e^{\epsilon} P\Big(\mathfrak{M}(\mathfrak{Data}) = \omega \mid t_1 \in [y-k,\ y+k), \theta\Big)$$

A similar calculation results in

$$P\Big(\mathfrak{M}(\mathfrak{Data}) = \omega \mid t_1 \in [y-k,\ y+k), \theta\Big) \ \leq\ e^{\epsilon} P\Big(\mathfrak{M}(\mathfrak{Data}) = \omega \mid t_1 \in [y+k,\ y+3k), \theta\Big)$$

We can repeat this calculation for other individuals (not just h_1), for other choices of $f_1, \ldots, f_n$ and for other y and so $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$. $\square$

I. PROOF OF THEOREM 8.1

THEOREM 8.1. *Let μ be a metric over database instances such that whenever $\mu(D_1, D_2) \leq \delta$ there exists*[14] *a $D^* \in \mathcal{I}$ with $\mu(D_1, D^*) \leq \delta$ and $\mu(D_2, D^*) > \delta$. Let $\epsilon > 0$ and $\delta > 0$. Set $\mathbb{S}_\delta$ as in Equation 20 and $\mathbb{S}_{pairs_\delta}$ as in Equation 21. Define $\mathbb{D}$ to be the set of distributions over dataset instances with n records where record values are independent (e.g., all distributions of the form given in Equation 19). If $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}_\delta, \mathbb{S}_{pairs_\delta}, \mathbb{D})$ then it also satisfies (ϵ, δ)-modified ZLW privacy; conversely, if $\mathfrak{M}$ satisfies (ϵ, δ)-modified ZLW privacy then it satisfies the definition 4ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}_\delta, \mathbb{S}_{pairs_\delta}, \mathbb{D})$ (i.e. up to a four-fold degradation of semantic guarantees in terms of odds-ratio).*

PROOF.

Step 1:

First suppose that $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}_\delta, \mathbb{S}_{\text{pairs}_\delta}, \mathbb{D})$. Let D_1 and D_2 be any two databases such that $\mu(D_1, D_2) \leq \delta$. The conditions in the theorem imply that there exists a D^* such that $\mu(D^*, D_1) \leq \delta$ and $\mu(D^*, D_2) > \delta$. Choose the discriminative pair $(\sigma_{[\mu, D^*, \leq\delta]}, \sigma_{[\mu, D^*, >\delta]}) \in \mathbb{S}_{\text{pairs}_\delta}$ and choose a $\theta \in \mathbb{D}$ such that $P(D_1 \mid \sigma_{[\mu, D^*, \leq\delta]}, \theta) = 1$ and $P(D_2 \mid \sigma_{[\mu, D^*, >\delta]}, \theta) = 1$. The conditions imposed by Pufferfish (Equations 1 and 2) then imply that

$$e^{-\epsilon} P(\mathfrak{M}(D_2) = \omega) \leq P(\mathfrak{M}(D_1) = \omega) \leq e^{\epsilon} P(\mathfrak{M}(D_2) = \omega)$$

Thus $\mathfrak{M}$ satisfies (ϵ, δ)-modified ZLW privacy.

Step 2: Now suppose that $\mathfrak{M}$ satisfies (ϵ, δ)-modified ZLW privacy.

$$\begin{aligned}
&P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid \sigma_{[\mu, D^*, \leq\delta]}, \theta) \\
&= \int P(\mathfrak{M}(D) = \omega) P(\mathfrak{Data} = D \mid \sigma_{[\mu, D^*, \leq\delta]}, \theta)\, dD \\
&= \int \left(\begin{array}{c} P(\mathfrak{M}(D)=\omega) P(\mathfrak{Data}=D \mid \sigma_{[\mu, D^*, \leq\delta]}, \theta) \\ \times \int P(\mathfrak{Data}=D' \mid \sigma_{[\mu, D^*, >\delta]}, \theta)\, dD' \end{array} \right) dD \\
&\leq e^4 \epsilon \int \left(\begin{array}{c} P(\mathfrak{M}(D')=\omega) P(\mathfrak{Data}=D \mid \sigma_{[\mu, D^*, \leq\delta]}, \theta) \\ \times \int P(\mathfrak{Data}=D' \mid \sigma_{[\mu, D^*, >\delta]}, \theta)\, dD' \end{array} \right) dD \\
&\quad \text{(using the definition of modified ZLW privacy and} \\
&\quad \text{because } \mu(D, D') \leq 4\delta\text{)} \\
&= e^{4\epsilon} \int P(\mathfrak{Data} = D \mid \sigma_{[\mu, D^*, \leq\delta]}, \theta)\, dD \\
&\quad \times \int P(\mathfrak{M}(D') = \omega) P(\mathfrak{Data} = D' \mid \sigma_{[\mu, D^*, >\delta]}, \theta)\, dD' \\
&= e^{4\epsilon} \int P(\mathfrak{M}(D') = \omega) P(\mathfrak{Data} = D' \mid \sigma_{[\mu, D^*, >\delta]}, \theta)\, dD' \\
&= e^{4\epsilon} P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid \sigma_{[\mu, D^*, >\delta]}, \theta)
\end{aligned}$$

and similarly, $P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid \sigma_{[\mu, D^*, >\delta]}, \theta) \leq e^{4\epsilon} P(\mathfrak{M}(\mathfrak{Data}) = \omega \mid \sigma_{[\mu, D^*, \leq\delta]}, \theta)$. □

J. PROOF OF THEOREM 9.1

THEOREM 9.1. *Given $\mathbb{S}$ and $\mathbb{S}_{pairs}$, the probability distribution θ is a universally composable evolution scenario for $\mathbb{S}_{pairs}$ if and only if for all $(s_i, s_j) \in \mathbb{S}_{pairs}$ having*

[14]This condition is achieved, for example, by the L_1 norm, L_2 norm, etc. as long as δ is less than the radius of $\mathcal{I}$.

$P(s_i \mid \theta) \neq 0$ *and* $P(s_j \mid \theta) \neq 0$ *there exist datasets* $D_i, D_j \in \mathcal{I}$ *such that* $P(\mathfrak{Data} = D_i \mid s_i, \theta) = 1$ *and* $P(\mathfrak{Data} = D_j \mid s_j, \theta) = 1$

PROOF. **Step 1:** First we show that if (1) $\mathbb{D}$ is arbitrary, (2) the definition $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ self-composes linearly, and (3) $\theta^* \notin \mathbb{D}$ has the property that for every $(s_I, s_J) \in \mathbb{S}_{\text{pairs}}$ with nonzero probability under θ^* there exist $D_i, D_j \in \mathcal{I}$ with $P(\mathfrak{Data} = D_i \mid s_i, \theta^*) = 1$ and $P(\mathfrak{Data} = D_j \mid s_j, \theta^*) = 1$ then the privacy definition $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D} \cup \{\theta^*\})$ self-composes linearly.

Choose any $\epsilon_1, \epsilon_2 > 0$. Choose any $\mathfrak{M}_1$ that satisfies ϵ_1-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D} \cup \{\theta^*\})$ and any $\mathfrak{M}_2$ that satisfies ϵ_2-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D} \cup \{\theta^*\})$ (with $\mathfrak{M}_1$ and $\mathfrak{M}_2$ having independent sources of randomness). Note that this means $\mathfrak{M}_1$ also satisfies ϵ_1-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$ while $\mathfrak{M}_2$ also satisfies ϵ_2-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$.

Now, for any $(s_i, s_j) \in \mathbb{S}$ and any $\theta \in \mathbb{D}$ for which $P(\cdot \mid s_i, \theta)$ and $P(\cdot \mid s_j, \theta)$ are defined, we must have (due to the linear self-composition property of $\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D})$):

$$\begin{aligned} &P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \wedge \mathfrak{M}_2(\mathfrak{Data}) = \omega_2 \mid s_i, \theta) \\ &\leq e^{\epsilon_1+\epsilon_2} P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \wedge \mathfrak{M}_2(\mathfrak{Data}) = \omega_2 \mid s_j, \theta) \end{aligned}$$

and

$$\begin{aligned} &P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \wedge \mathfrak{M}_2(\mathfrak{Data}) = \omega_2 \mid s_j, \theta) \\ &\leq e^{\epsilon_1+\epsilon_2} P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \wedge \mathfrak{M}_2(\mathfrak{Data}) = \omega_2 \mid s_i, \theta) \end{aligned}$$

Thus we need to show the same holds for θ^*. Let $(s_i, s_j) \in \mathbb{S}$ be any discriminative pair for which $P(s_i \mid \theta^*) \neq 0$ and $P(s_j \mid \theta^*) \neq 0$. Let D_i and D_j be the corresponding datasets for which $P(D_i \mid s_i, \theta) = 1$ and $P(D_j \mid s_j, \theta) = 1$. Then since we chose $\mathfrak{M}_1$ to satisfy ϵ_1-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D} \cup \{\theta^*\})$ and $\mathfrak{M}_2$ to satisfy ϵ_2-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D} \cup \{\theta^*\})$ then by construction we have for all $\omega_1 \in \text{range}(\mathfrak{M}_1)$ and $\omega_2 \in \text{range}(\mathfrak{M}_2)$:

$$\begin{aligned} P(\mathfrak{M}_1(D_i) = \omega_1) &= P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \mid s_i, \theta^*) \\ &\leq e^{\epsilon_1} P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \mid s_j, \theta^*) \\ &= e^{\epsilon_1} P(\mathfrak{M}_1(D_j) = \omega_1) \end{aligned}$$

similarly

$$\begin{aligned} P(\mathfrak{M}_1(D_j) = \omega_1) &\leq e^{\epsilon_1} P(\mathfrak{M}_1(D_i) = \omega_1) \\ P(\mathfrak{M}_2(D_j) = \omega_2) &\leq e^{\epsilon_2} P(\mathfrak{M}_2(D_i) = \omega_2) \\ P(\mathfrak{M}_2(D_i) = \omega_2) &\leq e^{\epsilon_2} P(\mathfrak{M}_2(D_j) = \omega_2) \end{aligned}$$

and thus

$$\begin{aligned} &P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \wedge \mathfrak{M}_2(\mathfrak{Data}) = \omega_2) \mid s_i, \theta^*) \\ &= P(\mathfrak{M}_1(D_i) = \omega_1 \wedge \mathfrak{M}_2(D_i) = \omega_2) \\ &= P(\mathfrak{M}_1(D_i) = \omega_1) P(\mathfrak{M}_2(D_i) = \omega_2) \\ &\leq e^{\epsilon_1+\epsilon_2} P(\mathfrak{M}_1(D_j) = \omega_1) P(\mathfrak{M}_2(D_j) = \omega_2) \\ &= e^{\epsilon_1+\epsilon_2} P(\mathfrak{M}_1(D_j) = \omega_1 \wedge \mathfrak{M}_2(D_j) = \omega_2) \\ &= e^{\epsilon_1+\epsilon_2} P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \wedge \mathfrak{M}_2(\mathfrak{Data}) = \omega_2) \mid s_j, \theta^*) \end{aligned}$$

and similarly

$$\begin{aligned} &P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \wedge \mathfrak{M}_2(\mathfrak{Data}) = \omega_2) \mid s_j, \theta^*) \\ &\leq e^{\epsilon_1+\epsilon_2} P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \wedge \mathfrak{M}_2(\mathfrak{Data}) = \omega_2) \mid s_i, \theta^*) \end{aligned}$$

Thus the algorithm $\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}$ whose range is $\mathrm{range}(\mathfrak{M}_1) \times \mathrm{range}(\mathfrak{M}_2)$ and output probabilities are $P\big[\mathfrak{M}^*_{\mathfrak{M}_1,\mathfrak{M}_2}(D) = (\omega_1,\omega_2)\big] = P[\mathfrak{M}_1(D) = \omega_1]P[\mathfrak{M}_2(D) = \omega_2]$ for all $D \in \mathcal{I}$ satisfies all of the conditions imposed by the definition $(\epsilon_1+\epsilon_2)$-$\mathcal{P}\mathrm{uffer}\mathcal{F}\mathrm{ish}(\mathbb{S},\mathbb{S}_{\mathrm{pairs}},\mathbb{D}\cup\{\theta^*\})$ and so this privacy definition self-composes linearly and therefore θ^* is a universally composable evolution scenario for $\mathbb{S}_{\mathrm{pairs}}$.

Step 2: To show the other direction, let θ be a universally composable evolution scenario that does not satisfy the hypothesis of the theorem. **To simplify the proof, we assume θ is a discrete distribution rather than a density. For the general case, simply take an appropriate neighborhood of the dataset $D^\star$ that will be used.** Thus we assume that θ is a universally composable evolution scenario that does not satisfy the hypothesis of the theorem. That is, there exists a discriminative pair $(s_i,s_j) \in \mathbb{S}_{\mathrm{pairs}}$ such that $P(s_i \mid \theta) > 0$ and $P(s_j \mid \theta) > 0$ yet there also exists a dataset $D^\star$ (other than D_i and D_j) such that $0 < P(\mathfrak{Data} = D^\star \mid s_i,\theta) < 1$ or $0 < P(\mathfrak{Data} = D^\star \mid s_j,\theta) < 1$. Without loss of generality, since the roles of s_i and s_j are symmetric, we shall assume that $0 < P(\mathfrak{Data} = D^\star \mid s_i,\theta) < 1$, in which case $P(\mathfrak{Data} = D^\star \mid s_j,\theta) = 0$ since s_i must be true for $D^\star$ (because the conditional probability is nonzero) and because s_i and s_j are mutually exclusive (by definition of discriminative pair).

Now consider the definition ϵ-$\mathcal{P}\mathrm{uffer}\mathcal{F}\mathrm{ish}(\{s_i,s_j\},\{(s_i,s_j)\},\emptyset)$ (i.e. there are only two potential secrets $\mathbb{S} = \{s_i,s_j\}$, one discriminative pair $\mathbb{S}_{\mathrm{pairs}} = \{(s_i,s_j)\}$ and vacuous distributional assumptions: $\mathbb{D} = \emptyset$). Then every algorithm satisfies this definition for every ϵ and thus it trivially composes. We show that by adding the evolution scenario θ we no longer have linear self-composition (this would show that θ is not a universally composable evolution scenario after all, and would complete the proof).

Consider the following algorithm $\mathfrak{M}_1$ that has only two possible outputs ω_1 and ω_2.

$$P(\mathfrak{M}_1(D)=\omega_1) = \begin{cases} 1 & \text{if } D = D^\star \\ 0 & \text{if } D \neq D^\star \text{ but } s_i \text{ is true for } D \\ 1/2 & \text{if } s_j \text{ is true for } D \end{cases}$$

$$P(\mathfrak{M}_1(D)=\omega_2) = \begin{cases} 0 & \text{if } D = D^\star \\ 1 & \text{if } D \neq D^\star \text{ but } s_i \text{ is true for } D \\ 1/2 & \text{if } s_j \text{ is true for } D \end{cases}$$

Now set

$$e^{\epsilon^\star} > \max\left\{\begin{array}{l} 2P(\mathfrak{Data} = D^\star \mid s_i,\theta), \\ 2P(\mathfrak{Data} \neq D^\star \mid s_i,\theta), \\ \frac{1}{2P(\mathfrak{Data}=D^\star \mid s_i,\theta)}, \\ \frac{1}{2P(\mathfrak{Data}\neq D^\star \mid s_i,\theta)} \end{array}\right\}$$

Using the definition of $\mathfrak{M}$ and ϵ^* and recalling that s_i is true for $D^\star$ but s_j is not, and $0 < P(D^\star \mid s_i,\theta) < 1$:

$$\begin{aligned} P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \mid s_i,\theta) &= P(\mathfrak{Data} = D^\star \mid s_i,\theta) \\ P(\mathfrak{M}_1(\mathfrak{Data}) = \omega_1 \mid s_j,\theta) &= 1/2 \end{aligned}$$

so that

$$P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_1 \mid s_i, \theta) \;\leq\; e^{\epsilon^*} P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_1 \mid s_j, \theta)$$
$$P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_1 \mid s_j, \theta) \;\leq\; e^{\epsilon^*} P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_1 \mid s_i, \theta)$$

Similarly,

$$P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_2 \mid s_i, \theta) \;=\; P(\boldsymbol{\mathfrak{Data}} \neq D^\star \mid s_i, \theta)$$
$$P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_2 \mid s_j, \theta) \;=\; 1/2$$

so that

$$P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_2 \mid s_i, \theta) \;\leq\; e^{\epsilon^*} P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_2 \mid s_j, \theta)$$
$$P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_2 \mid s_j, \theta) \;\leq\; e^{\epsilon^*} P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_2 \mid s_i, \theta)$$

Therefore $\mathfrak{M}_1$ satisfies the privacy definition
$\epsilon^\star$-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\{s_i, s_j\}, \{(s_i, s_j)\}, \{\theta\})$.

Now consider an algorithm $\mathfrak{M}_2$ that has the same probabilistic behavior as $\mathfrak{M}_1$ but its random bits are independent from $\mathfrak{M}_1$. Hence $\mathfrak{M}_2$ also satisfies the privacy definition $\epsilon^\star$-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\{s_i, s_j\}, \{(s_i, s_j)\}, \{\theta\})$. However, when we run both algorithms independently over the same data,

$$P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_1 \;\wedge\; \mathfrak{M}_2(\boldsymbol{\mathfrak{Data}}) = \omega_2 \mid s_i, \theta) = 0$$

because $\mathfrak{M}_1$ and $\mathfrak{M}_2$ behave deterministically and in the same way whenever s_i is true for the input dataset and so either both return ω_1 or both return $omega_2$. However,

$$P(\mathfrak{M}_1(\boldsymbol{\mathfrak{Data}}) = \omega_1 \;\wedge\; \mathfrak{M}_2(\boldsymbol{\mathfrak{Data}}) = \omega_2 \mid s_j, \theta) = 1/4$$

because whenever s_j is true, $\mathfrak{M}_1$ and $\mathfrak{M}_2$ return either output with probability $1/2$.

Thus in order for $\epsilon^\star$-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\{s_i, s_j\}, \{(s_i, s_j)\}, \{\theta\})$ to self-compose linearly, we would need $1/4 \leq e^{2\epsilon^*} \times 0$ (which cannot happen). This means that θ was not a universally composable evolution scenario after all. □

K. PROOF OF THEOREM 10.1

THEOREM 10.1. (Necessary Condition). *Given a set of general constraint $\mathcal{Q}$, if $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*_{\mathcal{Q}})$ (with $\mathbb{S}$, $\mathbb{S}_{pairs}$, and $\mathbb{D}^*_{\mathcal{Q}}$ given by Equations 23, 24, and 25) then $\mathfrak{M}$ satisfies ϵ-induced neighbors privacy with respect to $\mathcal{Q}$.*

PROOF. Consider any pair of contingency tables D_1 and D_2 that (a) are induced neighbors with respect to $\mathcal{Q}$, and (b) differ in the value of record r_i (s_1 and s_2, resp.). The set of moves that change D_1 to D_2 is a permutation of adding tuples in $T_{2\setminus 1} = D_2 \setminus D_1$, and removing tuples in $T_{1\setminus 2} = D_1 \setminus D_2$.

Consider $\theta^\star$ such that:

— for all individuals i that do not appear in either D_1 or D_2,

$$\pi_1 = 0$$

— for all individuals i that appear in $D_\cap = D_1 \cap D_2$ with the same value s_1,

$$\pi_i = 1 \text{ and } f_i(s) = \begin{cases} 0, & s \neq s_1; \\ 1, & s = s_1. \end{cases}$$

— for individuals i that appear in D_1 with value s_1 and in D_2 with value s_2,

$$\pi_i = 1, \text{and} f_i(s) = \begin{cases} 0, & s \neq s_1, s_2; \\ 1/2, & s = s_1 \text{ or } s = s_2. \end{cases}$$

— for individuals i that appear in only one of D_1 or D_2 with value s_1 (i.e., in $D_\Delta = T_{2\setminus 1} \cup T_{1\setminus 2}$),

$$\pi_i = 1/2 \text{ and } f_i(s) = \begin{cases} 0, & s \neq s_1; \\ 1, & s = s_1. \end{cases}$$

That is, the set of possible databases is given by

$$\mathcal{D}_\theta = \{D_\cap \cup D | D \subseteq D_\Delta\}$$

The following statements about $\mathcal{D}_\theta$ are true:

— D_1 is the only database in $\mathcal{D}_\theta$ such that (a) $r_i = s_1 \in D_1$ and (b) $D_1 \vdash \mathcal{Q}$.
— D_2 is the only database in $\mathcal{D}_\theta$ such that (a) $r_i = s_2 \in D_2$ and (b) $D_2 \vdash \mathcal{Q}$.

We will prove the first statement (and the second follows analogously). Suppose there exists a D_3 that satisfies $\mathcal{Q}$ and has $r_i = s_1$. Since $D_3 \neq D_1$, we have that $D_3 = D_\cap \cup D_x \cup D_y$, where $D_x \subseteq T_{1\setminus 2}$ and $D_y \subseteq T_{2\setminus 1}$; and one of these is a strict subset relationship. Therefore, the set of moves needed to go from D_3 to D_2 would be – delete tuples in D_x, and add tuples in $T_{2\setminus 1} \setminus D_y$. But this is a strict subset of the moves needed to move from D_1 to D_2; thus contradicting our assumption that D_1 and D_2 are induced neighbors.

Since we assume that $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}})$, for all outputs ω,

$$\begin{aligned} & P(\mathfrak{M}(D_1) = \omega) \\ &= \int_D P(\mathfrak{M}(D) = \omega) P(\boldsymbol{\mathfrak{Data}} = D | \sigma_{i,s_1}, \theta^\star) dD \\ &= P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = \omega | \sigma_{i,s_1}, \theta^\star) \\ &\leq e^\epsilon P(\mathfrak{M}(\boldsymbol{\mathfrak{Data}}) = \omega | \sigma_{i,s_2}, \theta^\star) \\ &= e^\epsilon \int_D P(\mathfrak{M}(D) = \omega) P(\boldsymbol{\mathfrak{Data}} = D | \sigma_{i,s_2}, \theta^\star) dD \\ &= e^\epsilon P(\mathfrak{M}(D_2) = \omega) \end{aligned}$$

Therefore, any mechanism that satisfied ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{\text{pairs}}, \mathbb{D}^*_{\mathcal{Q}})$ also satisfies ϵ-induced neighbors privacy with respect to $\mathcal{Q}$. □

L. PROOF OF THEOREM 10.2

We will need the following technical lemma:

LEMMA L.1. *Suppose X and Y are two finite sets of real numbers such that $\exists \epsilon, \forall x \in X, y \in Y$, $x \leq \epsilon \cdot y$. Let μ_X and μ_Y be arbitrary measures on X and Y respectively. Then $E_{\mu_X}(X) \leq \epsilon \cdot E_{\mu_Y}(Y)$.*

PROOF. The proof follows from:

$$E_{\mu_X}(X) \leq \max_{x \in X} x \leq \epsilon \cdot \min_{y \in Y} y \leq E_{\mu_Y}(Y)$$

□

We first prove Theorem 10.2 for a single count constraint.

LEMMA L.2. *(SUFFICIENT CONDITION FOR SINGLE COUNT CONSTRAINT). Let Q be a single count constraint $\sum_{t \in \mathfrak{Data}} g(t) = C$, where $g : \mathcal{T} \rightarrow \{0, 1\}$ is a function from the domain of tuples to 0 or 1. Define $\mathbb{S}$ and $\mathbb{S}_{pairs}$ as in Equations 23 and 24, and let $\mathbb{D}^*_Q$ be the set of all distributions with form specified in Equation 25. Then $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*_Q)$ if ϵ-induced neighbors privacy with respect to Q is satisfied.*

PROOF. We need to prove that if ϵ-induced neighbors privacy is satisfied then for all $\theta \in \mathbb{D}^*_Q$, and for all outputs w, the following condition holds,

$$P(\mathfrak{M}(\mathfrak{Data}) = w | \theta, Q, s_1) \leq P(\mathfrak{M}(\mathfrak{Data}) = w | \theta, Q, s_2)$$

for all discriminative pairs (s_1, s_2) of the form $(\sigma_{i,t}, \neg\sigma_i)$ and $(\sigma_{i,t}, \sigma_{i,t'})$. Here, $\sigma_{i,t}$ denotes the condition that record r_i appears in the data with value t, whereas $\neg\sigma_i$ denotes that r_i does not appear in the data. We present the proof for the former discriminative pair $(\sigma_{i,t}, \neg\sigma_i)$; the proof for the latter follows analogously.

We consider two cases

— $\sigma_{i,t}$ is such that $g(t) = 0$. Thus, for any dataset D where $\sigma_{i,t}$ is true, if D satisfies Q, then $D - \{r_i\}$ also satisfies Q.
— $\sigma_{i,t}$ is such that $g(t) = 1$. Thus, for any dataset D where $\sigma_{i,t}$ is true, if D satisfies Q, then $D - \{r_i\}$ *does not* satisfy Q.

Case (i): Consider a discriminative pair $(\sigma_{i,t}, \neg\sigma_i)$ such that $g(t) = 0$. Therefore, $\forall D$ that satisfies $\sigma_{i,t}$, $D' = D - \{r_i\}$ is a dataset that satisfies $\neg\sigma_i$ such that

$$\sum_{t \in D'} g(t) = \sum_{t \in D} g(t) = C \text{ and } P(\mathfrak{Data} = D \mid \theta, Q, \sigma_{i,t}) = P(\mathfrak{Data} = D' \mid \theta, Q, \sigma_i)$$

Moreover, D and D' are neighbors induced by the count constraint Q. Therefore,

$$\begin{aligned}
&P(\mathfrak{M}(\mathfrak{Data}) = w | \sigma_{i,t}, Q, \theta) \\
&= \int P(\mathfrak{M}(D) = w) P(\mathfrak{Data} = D | \sigma_{i,t}, Q, \theta) dD \\
&\leq e^\epsilon \cdot \int P(\mathfrak{M}(D') = w) P(\mathfrak{Data} = D - \{r_i\} | \neg\sigma_i, Q, \theta) dD \\
&= e^\epsilon P(\mathfrak{M}(\mathfrak{Data}) = w | \neg\sigma_i, Q, \theta)
\end{aligned}$$

Case (ii): Consider a discriminative pair $(\sigma_{i,t}, \neg\sigma_i)$ such that $g(t) = 1$.

Let D_{-ij} denote a database such that (a) $\forall k < j$, $k \neq i$, $\exists s$ such that $g(s) = 1$, and $\{r_k = s\}$ is in D_{-ij}, (b) r_i and r_j do not appear in D_{-ij}, and (c) $\sum_{t \in D_{-ij}} g(t) = C - 1$.

Let $A_{D_{-ij}}$ denote the following set of databases:

$$\{D_{-ij} + \{r_i = t\}\} \cup \{D_{-ij} + \{r_i = t, r_j = s'\} \mid g(s') = 0\}$$

That is, for all $D \in A_{D_{-ij}}$, (i) $\sigma_{i,t}$ is true, (ii) either $\neg\sigma_j$ or $\sigma_{j,s'}$ is true, for some $g(s') = 0$, and (iii) D satisfies Q.

Analogously, let $B_{D_{-ij}}$ denote the following set of databases:

$$\{D_{-ij} + \{r_j = s\} \mid g(s) = 1\}$$

For all $D \in B_{D_{-ij}}$, (i) $\neg\sigma_i$ is true, (ii) $\sigma_{j,s}$ is true, for some $g(s) = 1$, and (iii) D satisfies Q.

First note that, $\{A_{D_{-ij}}\}_{D_{-ij}, j}$ and $\{B_{D_{-ij}}\}_{D_{-ij}, j}$ partition the set of databases satisfying $\sigma_{i,t}$ and $\neg\sigma_i$, respectively. Second, for each D_{-ij}, every $D_a \in A_{D_{-ij}}$ is an induced

neighbor of each $D_b \in B_{D_{-ij}}$. Third,

$$P(\mathfrak{Data} \in A_{D_{-ij}} | \sigma_{i,t}, Q, \theta) \;=\; P(\mathfrak{Data} \in B_{D_{-ij}} | \neg\sigma_i, Q, \theta)$$

Therefore, we get

$$\begin{aligned}
&P(\mathfrak{M}(\mathfrak{Data}) = w | \sigma_{i,t}, Q, \theta) \\
&= \int P(\mathfrak{M}(D) = w | D \in A_{D_{-ij}}) P(\mathfrak{Data} \in A_{D_{-ij}} | \sigma_{i,t}, Q, \theta) dD \\
&\leq e^{\epsilon} \cdot \int P(\mathfrak{M}(D) = w | D \in B_{D_{-ij}}) P(\mathfrak{Data} \in A_{D_{-ij}} | \sigma_{i,t}, Q, \theta) dD \text{ (Lemma L.1)} \\
&= e^{\epsilon} \cdot \int P(\mathfrak{M}(D) = w | D \in B_{D_{-ij}}) P(\mathfrak{Data} \in B_{D_{-ij}} | \neg\sigma_i, Q, \theta) dD \\
&= e^{\epsilon} P(\mathfrak{M}(\mathfrak{Data}) = w | \neg\sigma_i, Q, \theta)
\end{aligned}$$

□

THEOREM 10.2. (SUFFICIENT CONDITION FOR UNIVARIATE HISTOGRAMS). *Given a univariate histogram constraint $\mathcal{Q}_{uni} : \{\sum_{t \in \mathfrak{Data}} g_i(t) = C\}$, define $\mathbb{S}$ and $\mathbb{S}_{pairs}$ as in Equations 23 and 24 and let $\mathbb{D}^*_Q$ be the set of all distributions with form specified in Equation 25. Then $\mathfrak{M}$ satisfies ϵ-$\mathcal{P}\text{uffer}\mathcal{F}\text{ish}(\mathbb{S}, \mathbb{S}_{pairs}, \mathbb{D}^*_{\mathcal{Q}_{uni}})$ if $\mathfrak{M}$ satisfies ϵ-induced neighbors privacy with respect to $\mathcal{Q}_{uni}$.*

PROOF. Since each tuple can contribute positively to at most one count in a univariate histogram, the proof follows from case (ii) of the proof of Lemma L.2. □