Blowfish Privacy: Tuning Privacy-Utility Trade-offs using Policies

Xi He
Duke University
Durham, NC, USA
hexi88@cs.duke.edu

Ashwin Machanavajjhala
Duke University
Durham, NC, USA
ashwin@cs.duke.edu

Bolin Ding
Microsoft Research
Redmond, WA, USA
bolin.ding@microsoft.com

ABSTRACT

Privacy definitions provide ways for trading-off the privacy of individuals in a statistical database for the utility of downstream analysis of the data. In this paper, we present *Blowfish*, a class of privacy definitions inspired by the Pufferfish framework, that provides a rich interface for this trade-off. In particular, we allow data publishers to extend differential privacy using a *policy*, which specifies (a) *secrets*, or information that must be kept secret, and (b) *constraints* that may be known about the data. While the secret specification allows increased utility by lessening protection for certain individual properties, the constraint specification provides added protection against an adversary who knows correlations in the data (arising from constraints). We formalize policies and present novel algorithms that can handle general specifications of sensitive information and certain count constraints. We show that there are reasonable policies under which our privacy mechanisms for k-means clustering, histograms and range queries introduce significantly lesser noise than their differentially private counterparts. We quantify the privacy-utility trade-offs for various policies analytically and empirically on real datasets.

Categories and Subject Descriptors

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

Keywords

privacy, differential privacy, Blowfish privacy

1. INTRODUCTION

With the increasing popularity of "big-data" applications which collect, analyze and disseminate individual level information in literally every aspect of our life, ensuring that these applications do not breach the privacy of individuals is an important problem. The last decade has seen the development of a number of privacy definitions and mechanisms that trade-off the privacy of individuals in these databases for the utility (or accuracy) of data analysis (see [4] for a survey). Differential privacy [6] has emerged as a gold standard not only because it is not susceptible to attacks that other definition can't tolerate, but also since it provides a simple knob, namely ϵ, for trading off privacy for utility.

While ϵ is intuitive, it does not sufficiently capture the diversity in the privacy-utility trade-off space. For instance, recent work has shown two seemingly contradictory results. In certain applications (e.g., social recommendations [17]) differential privacy is too strong and does not permit sufficient utility. Next, when data are correlated (e.g., when constraints are known publicly about the data, or in social network data) differentially private mechanisms may not limit the ability of an attacker to learn sensitive information [12]. Subsequently, Kifer and Machanavajjhala [13] proposed a semantic privacy framework, called Pufferfish, which helps clarify assumptions underlying privacy definitions – specifically, the information that is being kept secret, and the adversary's background knowledge. They showed that differential privacy is equivalent to a specific instantiation of the Pufferfish framework, where (a) every property about an individual's record in the data is kept secret, and (b) the adversary assumes that every individual is independent of the rest of the individuals in the data (no correlations). We believe that these shortcomings severely limit the applicability of differential privacy to real world scenarios that either require high utility, or deal with correlated data.

Inspired by Pufferfish, we seek to better explore the trade-off between privacy and utility by providing a richer set of "tuning knobs". We explore a class of definitions called *Blowfish privacy.* In addition to ϵ, which controls the amount of information disclosed, Blowfish definitions take as input a privacy *policy* that specifies two more parameters – which information must be kept secret about individuals, and what constraints may be known publicly about the data. By extending differential privacy using these policies, we can hope to develop mechanisms that permit more utility since not all properties of an individual need to be kept secret. Moreover, we also can limit adversarial attacks that leverage correlations due to publicly known constraints.

We make the following contributions in this paper:

- We introduce and formalize sensitive information specifications, constraints, policies and Blowfish privacy. We consider a number of realistic examples of sensitive information specification, and focus on count constraints.
- We show how to adapt well known differential privacy mechanisms to satisfy Blowfish privacy, and using the example of k-means clustering illustrate the gains in ac-

SIGMOD '14 Snowbird, Utah USA
Copyright 2014 ACM 978-1-4503-2376-5/14/06 ...$15.00.
http://dx.doi.org/10.1145/2588555.2588581.

curacy for Blowfish policies having weaker sensitive information specifications.

- We propose the ordered mechanism, a novel strategy for releasing cumulative histograms and answering range queries. We show analytically and using experiments on real data that, for reasonable sensitive information specifications, the ordered hierarchical mechanism is more accurate than the best known differentially private mechanisms for these workloads.
- We study how to calibrate noise for policies expressing count constraints, and its applications in several practical scenarios.

Organization: Section 2 introduces the notation. Section 3 formalizes privacy policies. We define Blowfish privacy, and discuss composition properties and its relationship to prior work in Section 4. We define the policy specific global sensitivity of queries in Section 5. We describe mechanisms for kmeans clustering (Section 6), and releasing cumulative histograms & answering range queries (Section 7) under Blowfish policies without constraints and empirically evaluate the resulting privacy-utility trade-offs on real datasets. We show how to release histograms in the presence of count constraints in Section 8 and then conclude in Section 9.

2. NOTATION

We consider a dataset D consisting of n tuples. Each tuple t is considered to be drawn from a domain $\mathcal{T} = A_1 \times A_2 \times \ldots \times A_m$ constructed from the cross product of m categorical attributes. We assume that each tuple t corresponds to the data collected from a unique individual with identifier $t.\_id$. We will use the notation $x \in \mathcal{T}$ to denote a value in the domain, and $x.A_i$ to denote the i^{th} attribute value in x.

Throughout this paper, we will make an assumption that the set of individuals in the dataset D is known in advance to the adversary and does not change. Hence we will use the indistinguishability notion of differential privacy [7]. We will denote the set of possible databases using $\mathcal{I}_n$, or the set of databases with $|D| = n$.[1]

Definition 2.1 (Differential Privacy [6]). *Two datasets D_1 and D_2 are neighbors, denoted by $(D_1, D_2) \in N$, if they differ in the value of one tuple. A randomized mechanism M satisfies ϵ-differential privacy if for every set of outputs $S \subseteq range(M)$, and every pair of neighboring datasets $(D_1,\, D_2) \in N$,*

$$Pr[M(D_1) \in S] \leq e^{\epsilon} Pr[M(D_2) \in S] \qquad (1)$$

Many techniques that satisfy differential privacy use the following notion of *global sensitivity*:

Definition 2.2 (Global Sensitivity). *The global sensitivity of a function $f : \mathcal{I}_n \to \mathbb{R}^d$, denoted by $S(f)$ is defined as the largest L1 difference $||f(D_1) - f(D_2)||_1$, where D_1 and D_2 are databases that differ in one tuple. More formally,*

$$S(f) \;=\; \max_{(D_1,D_2)\in N} ||f(D_1) - f(D_2)||_1 \qquad (2)$$

A popular technique that satisfies ϵ-differential privacy is the Laplace mechanism [7] defined as follows:

Definition 2.3. *The Laplace mechanism, M^{Lap}, privately computes a function $f : \mathcal{I}_n \to \mathbb{R}^d$ by computing $f(D) + \eta$. $\eta \in \mathbb{R}^d$ is a vector of independent random variables, where each η_i is drawn from the Laplace distribution with parameter $S(f)/\epsilon$. That is, $P[\eta_i = z] \propto e^{-z\cdot\epsilon/S(f)}$.*

Given some partitioning of the domain $\mathcal{P} = (P_1, \ldots, P_k)$, we denote by $h_{\mathcal{P}} : \mathcal{I} \to Z^k$ the histogram query. $h_{\mathcal{P}}(D)$ outputs for each P_i the number of times values in P_i appears in D. $h_{\mathcal{T}}(\cdot)$ (or $h(\cdot)$ in short) is the *complete* histogram query that reports for each $x \in \mathcal{T}$ the number of times it appears in D. It is easy to see that $S(h_{\mathcal{P}}) = 2$ for all histogram queries, and the Laplace mechanism adds noise proportional to $Lap(2/\epsilon)$ to each component of the histogram. We will use Mean Squared Error as a measure of accuracy/error.

Definition 2.4. *Let M be a randomized algorithm that privately computes a function $f : \mathcal{I}_n \to \mathbb{R}^d$. The expected mean squared error of M is given by:*

$$\mathcal{E}_M(D) \;=\; \sum_i \mathbb{E}(f_i(D) - \tilde{f}_i(D))^2 \qquad (3)$$

where $f_i(\cdot)$ and $\tilde{f}_i(\cdot)$ denote the i^{th} component of the true and noisy answers, respectively.

Under this definition the accuracy of the Laplace mechanism for histograms is given by $|\mathcal{T}| \cdot \mathbb{E}(Laplace(2/\epsilon))^2 = 8|\mathcal{T}|/\epsilon^2$.

3. POLICY DRIVEN PRIVACY

In this section, we describe an abstraction called a *policy* that helps specify which information has to be kept secret and what background knowledge an attacker may possess about the correlations in the data. We will use this policy specification as input in our privacy definition, called Blowfish, described in Section 4.

3.1 Sensitive Information

As indicated by the name, Blowfish[2] privacy is inspired by the Pufferfish privacy framework [13]. In fact, we will show later (in Section 4.2) that Blowfish privacy is equivalent to specific instantiations of semantic definitions arising from the Pufferfish framework.

Like Pufferfish, Blowfish privacy also uses the notions of secrets and discriminative pairs of secrets. We define a secret to be an arbitrary propositional statement over the values in the dataset. For instance, the secret $s : t.\_id =$ 'Bob' $\wedge\, t.Disease =$ 'Cancer' is true in a dataset where Bob has Cancer. We denote by $\mathcal{S}$ a set of secrets that the data publisher would like to protect. As we will see in this section each individual may have multiple secrets. Secrets may also pertain to sets of individuals. For instance, the following secret $s : t_1.\_id =$ 'Alice' $\wedge\, t_2.\_id =$ 'Bob' $\wedge\, t_1.Disease = t_2.Disease$ is true when Alice and Bob have the same disease. However, in this paper, we focus on the case where each secret is about a single individual.

We call a pair of secrets $(s, s') \in \mathcal{S} \times \mathcal{S}$ *discriminative* if they are mutually exclusive. Each discriminative pair describes properties that an adversary must not be able to distinguish between. One input to a policy is a set of discriminative pairs of secrets $\mathcal{S}_{\text{pairs}}$.

We now present a few examples of sensitive information specified as a set of discriminative secrets.

[1] In Sec. 3 we briefly discuss how to generalize our results to other differential privacy notions by relaxing this assumption.

[2] Pufferfish and Blowfish are common names of the same family of marine fish, Tetraodontidae.

- *Full Domain*: Let s_x^i be the secret $(t\_id = i \wedge t = x)$, for some $x \in \mathcal{T}$. We define $\mathcal{S}_{\text{pairs}}^{\text{full}}$ as:

$$\mathcal{S}_{\text{pairs}}^{\text{full}} = \{(s_x^i, s_y^i) | \forall i, \forall (x,y) \in \mathcal{T} \times \mathcal{T}\} \quad (4)$$

  This means that for every individual, an adversary should not be able to distinguish whether that individual's value is x or y, for all $x, y \in \mathcal{T}$.

- *Attributes*: Let $\mathbf{x} \in \mathcal{T}$ denote a multidimensional value. Let $\mathbf{x}[A]$ denote value of attribute A, and $\mathbf{x}[\bar{A}]$ the value for the other attributes. Then a second example of sensitive information is:

$$\mathcal{S}_{\text{pairs}}^{\text{attr}} = \{(s_{\mathbf{x}}^i, s_{\mathbf{y}}^i) | \forall i, \exists A, \mathbf{x}[A] \neq \mathbf{y}[A] \wedge \mathbf{x}[\bar{A}] = \mathbf{y}[\bar{A}]\} \quad (5)$$

  $\mathcal{S}_{\text{pairs}}^{attr}$ ensures that an adversary should not be able to sufficiently distinguish between any two values for each attribute of every individual's value.

- *Partitioned*: Let $\mathcal{P} = \{P_1, \ldots, P_p\}$ be a partition that divides the domain into p disjoint sets ($\cup_i P_i = \mathcal{T}$ and $\forall 1 \leq i, j \leq p, P_i \cap P_j = \emptyset$). We define partitioned sensitive information as:

$$\mathcal{S}_{\text{pairs}}^{\mathcal{P}} = \{(s_x^i, s_y^i) | \forall i, \exists j, (x,y) \in P_j \times P_j\} \quad (6)$$

  In this case, an adversary is allowed to deduce whether an individual is in one of two different partitions, but can't distinguish between two values within a single partition. This is a natural specification for location data – an individual may be OK with releasing his/her location at a coarse granularity (e.g., a coarse grid), but location within each grid cell must be hidden from the adversary.

- *Distance Threshold*: In many situations there is an inherent distance metric d associated with the points in the domain (e.g., L_1 distance on age or salary, or Manhattan distance on locations). Rather than requiring that an adversary should not be able to distinguish between any pairs of points x and y, one could require that each pair of points that are close are not distinguishable. So, for this purpose, the set of discriminative secrets is:

$$\mathcal{S}_{\text{pairs}}^{d,\theta} = \{(s_x^i, s_y^i) | \forall i, d(x,y) \leq \theta\} \quad (7)$$

  Under this policy, the adversary will not be able to distinguish any pair of values with certainty. However, the adversary may distinguish points that are farther apart better that points that are close.

All of the above specifications of sensitive information can be generalized using the *discriminative secret graph*, defined below. Consider a graph $G = (V, E)$, where $V = \mathcal{T}$ and the set of edges $E \subseteq \mathcal{T} \times \mathcal{T}$. The set of edges can be interpreted as values in the domain that an adversary must not distinguish between; i.e., the set of discriminative secrets is $\mathcal{S}_{\text{pairs}}^{G} = \{(s_x^i, s_y^i) \mid \forall i, \forall (x,y) \in E\}$. The above examples correspond to the following graphs: G^{full} corresponds to a complete graph on all the elements in $\mathcal{T}$. G^{attr} corresponds to a graph where two values are connected by an edge when only one attribute value changes. $G^{\mathcal{P}}$ has $|\mathcal{P}|$ connected components, where each component is a complete graph on vertices in P_i. Finally, in $G^{d,\theta}$, $(x,y) \in E$ iff $d(x,y) \leq \theta$.

We would like to note that a policy could have secrets and discriminative pairs about sets of individuals. However, throughout this paper, we only consider secrets pertaining to a single individual, and thus discriminative pairs refer to two secrets about the same individual. Additionally, the set of discriminative pairs is the same for all individuals. One can envision different individuals having different sets of discriminative pairs. For instance, we can model an individual who is privacy agnostic and does not mind disclosing his/her value exactly by having no discriminative pair involving that individual. Finally note that in all of the discussion in this section, the specification of what is sensitive information *does not depend on the original database* D. One could specify sensitive information that depends on D, but one must be wary that this might leak additional information to an adversary. In this paper, we focus on data-independent discriminative pairs, uniform secrets and secrets that only pertain to single individuals.

Throughout this paper, we will assume that the adversary knows the total number of tuples in the database (i.e., the set of possible instances is $\mathcal{I}_n$). Hence, we can limit ourselves to considering changes in tuples (and not additions or deletions). We can in principle relax this assumption about cardinality, by adding an additional set of secrets of the form $s_\perp^i$ which mean *"individual i is not in dataset"*. All of our definitions and algorithms can be modified to handle this case by adding $\perp$ to the domain and to the discriminative secret graph G. We defer these extensions to future work.

3.2 Auxiliary Knowledge

Recent work [12] showed that differentially private mechanisms could still lead to an inordinate disclosure of sensitive information when adversaries have access to publicly known constraints about the data that induce correlations across tuples. This can be illustrated by the following example. Consider a table D with one attribute R that takes values $r_1, \ldots, r_k$. Suppose, based on publicly released datasets the following $k-1$ constraints are already known: $c(r_1)+c(r_2) = a_1$, $c(r_2) + c(r_3) = a_2$, and so on, where $c(r_i)$ is the number of records with value r_i. This does not provide enough information to always reconstruct the counts in D (k unknowns but $k-1$ linear equations). However, if we knew the answer to some $c(r_i)$, then all counts can be reconstructed – in this way tuples are correlated.

Differential privacy allows answering all the count queries $c(r_i)$ by adding independent noise with variance $2/\epsilon^2$ to each count. While these noisy counts $\tilde{c}(r_i)$ themselves do not disclose information about any individual, they can be combined with the constraints to get very precise estimates of $c(r_i)$. That is, we can construct k independent estimators for each count as follow. For r_1, $\tilde{c}(r_1), a_1 - \tilde{c}(r_2), a_1 - a_2 + \tilde{c}(r_3), \ldots$ each equal $c(r_1)$ in expectation and have a variance of $2/\epsilon^2$. By averaging these estimators, we can predict the value of $c(r_i)$ with a variance of $2/(k\epsilon^2)$. For large k (e.g., when there are 2^d values in R), the variance is small so that the table D is reconstructed with very high probability, thus causing a complete breach of privacy.

Therefore, our policy specification also takes into account auxiliary knowledge that an adversary might know about the individuals in the private database. In Blowfish, we consider knowledge in the form of a set of deterministic constraints Q that are publicly known about the dataset. We believe these are easier to specify than probabilistic correlation functions for data publishers. The effect of the constraints in Q is to make only a subset of the possible database instances $\mathcal{I}_Q \subset \mathcal{I}_n$ possible; or equivalently, all instances in $\mathcal{I}_n \setminus \mathcal{I}_Q$ are impossible. For any database $D \in \mathcal{I}_n$, we denote by

$D \vdash Q$ if D satisfies the constraints in Q; i.e., $D \in \mathcal{I}_Q$. Examples of deterministic constraints include:

- *Count Query Constraints:* A count query on a database returns the number of tuples that satisfy a certain predicate. A count query constraints is a set of (count query, answer) pairs over the database that are publicly known.
- *Marginal Constraints:* A marginal is a projection of the database on a subset of attributes, and each row counts the number of tuples that agree on the subset of attributes. The auxiliary knowledge of marginals means these database marginals are known to the adversary.

3.3 Policy

Definition 3.1 (Policy). *A policy is a triple $P = (\mathcal{T}, G, \mathcal{I}_Q)$, where $G = (V, E)$ is a discriminative secret graph with $V \subseteq \mathcal{T}$. In P, the set of discriminative pairs $\mathcal{S}^G_{\text{pairs}}$ is defined as the set $\{(s^i_x, s^i_y) \,|\, \forall i \in \_id, \forall (x, y) \in E\}$, where s^i_x denotes the statement: $t.\_id = i \wedge t = x$. $\mathcal{I}_Q$ denotes the set of databases that are possible under the constraints Q that are known about the database.*

Note that the description of the policy can be exponential in the size of the input dataset. We will use shorthand to describe certain types of sensitive information (e.g., full domain, partition, etc), and specify the set of possible databases $\mathcal{I}_Q$ using the description of Q.

4. BLOWFISH PRIVACY

In this section, we present our new privacy definition, called Blowfish Privacy. Like differential privacy, Blowfish uses the notion of neighboring datasets. The key difference is that the set of neighbors in Blowfish depend on the policy P – both on the set of discriminative pairs as well as on the constraints known about the database.

Definition 4.1 (Neighbors). *Let $P = (\mathcal{T}, G, \mathcal{I}_Q)$ be a policy. For any pair of datasets D_1, D_2, let $T(D_1, D_2) \subseteq \mathcal{S}^G_{\text{pairs}}$ be the set of discriminative pairs (s^i_x, s^i_y) such that the i^{th} tuples in D_1 and D_2 are x and y, resp. Let $\Delta(D_1, D_2) = D_1 \setminus D_2 \cup D_2 \setminus D_1$. D_1 and D_2 are neighbors with respect to a policy P, denoted by $(D_1, D_2) \in N(P)$, if:*

1. *$D_1, D_2 \in \mathcal{I}_Q$. (i.e., both the datasets satisfy Q).*
2. *$T \neq \emptyset$. (i.e., $\exists (s^i_x, s^i_y) \in \mathcal{S}^G_{\text{pairs}}$ such that the i^{th} tuples in D_1 and D_2 are x and y, resp).*
3. *There is no database $D_3 \vdash Q$ such that*
   (a) $T(D_1, D_3) \subset T(D_1, D_2)$, or
   (b) $T(D_1, D_3) = T(D_1, D_2)$ & $\Delta(D_3, D_1) \subset \Delta(D_2, D_1)$.

When $P = (\mathcal{T}, G, \mathcal{I}_n)$ (i.e., no constraints), D_1 and D_2 are neighbors if some individual tuples value is changed from x to y, where (x, y) is an edge in G. Note that $T(D_1, D_2)$ is non-empty and has the smallest size (of 1). Neighboring datasets in differential privacy correspond to neighbors when G is a complete graph.

For policies having constraints, conditions 1 and 2 ensure that neighbors satisfy the constraints (i.e., are in $\mathcal{I}_Q$), and that they differ in at least one discriminative pair of secrets. Condition 3 ensures that D_1 and D_2 are minimally different in terms of discriminative pairs and tuple changes.

Definition 4.2 (Blowfish Privacy). *Let $\epsilon > 0$ be a real number and $P = (\mathcal{T}, G, \mathcal{I}_Q)$ be a policy. A randomized mechanism M satisfies (ϵ, P)-Blowfish privacy if for every pair of neighboring databases $(D_1, D_2) \in N(P)$, and every set of outputs $S \subseteq range(M)$, we have*

$$Pr[M(D_1) \in S] \;\leq e^{\epsilon} Pr[M(D_2) \in S] \qquad (8)$$

Note that Blowfish privacy takes in the policy P in addition to ϵ as an input, and is different from differential privacy in only the set of neighboring databases $N(P)$. For $P = (\mathcal{T}, G, \mathcal{I}_n)$ (i.e., no constraints), it is easy to check that for any two databases that arbitrarily differ in one tuple ($D_1 = D \cup \{x\}, D_2 = D \cup \{y\}$), and any set of outputs S,

$$Pr[M(D_1) \in S] \;\leq e^{\epsilon \cdot d_G(x,y)} Pr[M(D_2) \in S] \qquad (9)$$

where $d_G(x, y)$ is the shortest distance between x, y in G. This implies that an attacker may better distinguish pairs of points farther apart in the graph (e.g., values with many differing attributes in $\mathcal{S}^{\text{attr}}_{\text{pairs}}$), than those that are closer. Similarly, an attack can distinguish between x, y with probability 1, when x and y appear in different partitions under partitioned sensitive information $\mathcal{S}^{\mathcal{P}}_{\text{pairs}}$ ($d_G(x, y) = \infty$).

4.1 Composition

Composition [8] is an important property that any privacy notion should satisfy in order to be able to reason about independent data releases. Sequential composition ensures that a sequence of computations that each ensure privacy in isolation also ensures privacy. This allows breaking down computations into smaller building blocks. Parallel composition is crucial to ensure that too much error is not introduced on computations occurring on *disjoint* subsets of data. We can show that Blowfish satisfies sequential composition, and a weak form of parallel composition.

Theorem 4.1 (Sequential Composition). *Let $P = (\mathcal{T}, G, \mathcal{I}_Q)$ be a policy and $D \in \mathcal{I}_Q$ be an input database. Let $M_1(\cdot)$ and $M_2(\cdot, \cdot)$ be algorithms with independent sources of randomness that satisfy (ϵ_1, P) and (ϵ_2, P)-Blowfish privacy, resp. Then an algorithm that outputs both $M_1(D) = \omega_1$ and $M_2(\omega_1, D) = \omega_2$ satisfies $(\epsilon_1 + \epsilon_2, P)$-Blowfish privacy.*

Proof. See Appendix B. □

Theorem 4.2. (Parallel Composition with Cardinality Constraint). *Let $P = (\mathcal{T}, G, \mathcal{I}_n)$ be a policy where the cardinality of the input $D \in \mathcal{I}_n$ is known. Let $S_1, \ldots, S_p$ be disjoint subsets of _ids; $D \cap S_i$ denotes the dataset restricted to the individuals in S_i. Let M_i be mechanisms that each ensure (ϵ_i, P)-Blowfish privacy. Then the sequence of $M_i(D \cap S_i)$ ensures $(\max_i \epsilon_i, P)$-Blowfish privacy.*

Proof. See Appendix C. □

Reasoning about parallel composition in the presence of general constraints is non-trivial. Consider two neighboring datasets $D_a, D_b \in N(P)$. For instance, suppose one of the attributes is gender, we know the number of males and females in the dataset, and we are considering full domain sensitive information. Then there exist neighboring datasets such that differ in two tuples i and j that are alternately male and female in D_a and D_b. If i and j appear in different subsets S_1 and S_2 resp., then $D_a \cap S_1 \neq D_b \cap S_2$ and $D_a \cap S_1 \neq D_b \cap S_2$. Thus the sequence $M_i(D \cap S_i)$ does not ensure $(\max_i \epsilon_i, P)$-Blowfish privacy. We generalize this observation below.

Define a pair of secrets (s, s') to be *critical* to a constraint q if there exist $D_s, D_{s'}$ such that $T(D_s, D_{s'}) = (s, s')$, and $D_s \vdash q$, but $D_{s'} \not\vdash q$. Let *crit*(q) denote the set of secret pairs that are critical to q. Next, consider $S_1, \ldots, S_k$ disjoint subsets of ids. We denote by $SP(S_i)$ the set of secret pairs that pertain to the ids in S_i. We say that a constraint q *affects* $D \cap S_i$ if $crit(q) \cap SP(S_i) \neq \emptyset$. We can now state a sufficient condition for parallel composition.

THEOREM 4.3. (PARALLEL COMPOSITION WITH GENERAL CONSTRAINTS). *Let $P = (\mathcal{T}, G, \mathcal{I}_Q)$ be a policy and $S_1, \ldots, S_p$ be disjoint subsets of _ids. Let M_i be mechanisms that each ensure (ϵ_i, P)-Blowfish privacy. Then the sequence of $M_i(D \cap S_i)$ ensures* $(\max_i \epsilon_i, P)$*-Blowfish privacy if there exist* disjoint *subsets of constraints $Q_1, \ldots, Q_p \subset Q$ such that all the constraints in Q_i only affects $D \cap S_i$.*

PROOF. See Appendix C. □

We conclude this section with an example of parallel composition. Suppose G contains two disconnected components on nodes S and $\mathcal{T} \setminus S$. The set of all secret pairs correspond to pairs of values that come either from S or from $\mathcal{T} \setminus S$. Suppose we know two count constraints q_S and $q_{\mathcal{T} \setminus S}$ that count the number of tuples with values in S and $\mathcal{T} \setminus S$, respectively. It is easy to see that $crit(q_S) = crit(q_{\mathcal{T} \setminus S}) = 0$. Therefore, running an $(\epsilon, (\mathcal{T}, G, \{q_S, q_{\mathcal{T} \setminus S}\}))$-Blowfish private mechanism on disjoint subsets results in no loss of privacy.

4.2 Relation to other definitions

In this section, we relate Blowfish privacy to existing notions of privacy. We discuss variants of differential privacy [6] (including restricted sensitivity [1]), the Pufferfish framework [13], privacy axioms [11], and a recent independent work on extending differential privacy with metrics [3].

Differential Privacy [6]: One can easily verify that a mechanism satisfies ϵ-differential privacy (Definition 2.1) if and only if it satisfies (ϵ, P)-Blowfish privacy, where $P = (\mathcal{T}, K, \mathcal{I}_n)$, and K is the complete graph on the domain. Thus, Blowfish privacy is a generalization of differential privacy that allows a data curator to trade-off privacy vs utility by controlling sensitive information G (instead of K) and auxiliary knowledge $\mathcal{I}_Q$ (instead of $\mathcal{I}_n$) in the policy.

Pufferfish Framework [13, 14]: Blowfish borrows the sensitive information specification from Pufferfish. Pufferfish defines adversarial knowledge using a set of data generating distributions, while Blowfish instantiates the same using publicly known constraints. We can show formal relationships between Blowfish and Pufferfish instantiations.

THEOREM 4.4. *Let $\mathcal{S}_{\text{pairs}}$ be the set of discriminative pairs corresponding to policy $P = (\mathcal{T}, G, \mathcal{I}_n)$. Let $\mathcal{D}$ denote the set of all product distributions $\{p_i(\cdot)\}_i$ over n tuples. $p_i(\cdot)$ denotes a probability distribution for tuple i over $\mathcal{T}$. Then a mechanism satisfies $(\epsilon, \mathcal{S}_{\text{pairs}}, \mathcal{D})$-Pufferfish privacy if and only if it satisfies (ϵ, P)-Blowfish privacy.*

THEOREM 4.5. *Consider a policy $P = (\mathcal{T}, G, \mathcal{I}_Q)$ corresponding to a set of constraints Q. Let $\mathcal{S}_{\text{pairs}}$ be defined as in Theorem 4.4. Let $\mathcal{D}_Q$ be the set of product distributions conditioned on the constraints in Q; i.e.,*

$$P[D = x_1, \ldots, x_k] \propto \begin{cases} \prod_i p_i(x_i) & \text{if } D \in \mathcal{I}_Q \\ 0 & \text{otherwise} \end{cases}$$

A mechanism M that satisfies $(\epsilon, \mathcal{S}_{\text{pairs}}, \mathcal{D}_Q)$-Pufferfish privacy also satisfies (ϵ, P)-Blowfish privacy.

Theorem 4.4 states that Blowfish policies without constraints are equivalent to Pufferfish instantiated using adversaries who believe tuples in D are independent (proof follows from Theorem 6.1 [14]). Theorem 4.5 states that when constraints are known, Blowfish is a necessary condition for any mechanism that satisfies a similar Pufferfish instantiation with constraints (we conjecture the sufficiency of Blowfish as well). Thus Blowfish privacy policies correspond to a subclass of privacy definitions that can be instantiated using Pufferfish.

Both Pufferfish and Blowfish aid the data publisher to customize privacy definitions by carefully defining sensitive information and adversarial knowledge. However, Blowfish improves over Pufferfish in three key aspects. First, there are no general algorithms known for Pufferfish instantiations. In this paper, we present of algorithms for various Blowfish policies. Thus, we can't compare Blowfish and Pufferfish experimentally. Second, all Blowfish privacy policies result in composable privacy definitions. This is not true for the Pufferfish framework. Finally, we believe Blowfish privacy is easier to understand and use than the Pufferfish framework for data publishers who are not privacy experts.[3] For instance, one needs to specify adversarial knowledge as sets of complex probability distributions in Pufferfish, while in Blowfish policies one only needs to specify conceptually simpler publicly known constraints.

Other Privacy Definitions: Kifer and Lin [11] stipulate that every "good" privacy definition should satisfy two axioms – transformation invariance, and convexity. We can show that Blowfish privacy satisfy both these axioms.

Recent papers have extended differential privacy to handle constraints. Induced neighbor privacy [12, 13] extends the notion of neighbors such that neighboring databases satisfy the constraints and are minimally far apart (in terms of tuple changes). Blowfish extends this notion of induced neighbors to take into account discriminative pairs of secrets and measures distance in terms of the set of different discriminative pairs. Restricted sensitivity [1] extends the notion of sensitivity to account for constraints. In particular, the restricted sensitivity of a function f given a set of constraints Q, or $RS_f(Q)$, is the maximum $|f(D_1) - f(D_2)|/d(D_1, D_2)$, over all $D_1, D_2 \in \mathcal{I}_Q$. However, tuning noise to $RS_f(Q)$ may not limit the ability of an attacker to learn sensitive information. For instance, if $\mathcal{I}_Q = \{0^n, 1^n\}$, then the restricted sensitivity of releasing the number of 1s is 1. Adding constant noise does not disallow the adversary from knowing whether the database was 0^n or 1^n.

A very recent independent work suggests extending differential privacy using a metric over all possible databases [3]. In particular, given a distance metric d over instances, they require an algorithm to ensure that $P[M(X) \subseteq S] \leq e^{\epsilon \cdot d(X,Y)} P[M(Y) \subseteq S]$, for all sets of outputs S and all instances X and Y. Thus differential privacy corresponds to a specific distance measure – Hamming distance. The sensitive information specification in Blowfish can also be thought of in terms of a distance metric over tuples. In addition we present novel algorithms (ordered mechanism) and allow incorporating knowledge of constraints. We defer a more detailed comparison to future work.

[3] We have some initial anecdotal evidence of this fact working with statisticians from the US Census.

5. BLOWFISH WITHOUT CONSTRAINTS

Given any query f that outputs a vector of reals, we can define a *policy specific sensitivity* of f. Thus, the Laplace mechanism with noise calibrated to the policy specific sensitivity ensures Blowfish privacy.

Definition 5.1. (Policy Specific Global Sensitivity). *Given a policy $(\mathcal{T}, G, \mathcal{I}_Q)$, $S(f, P)$ denotes the policy specific global sensitivity of a function f and is defined as $\max_{(D_1,D_2)\in N(P)} ||f(D_1) - f(D_2)||_1$.*

Theorem 5.1. *Let $P = (\mathcal{T}, G, \mathcal{I}_Q)$ be a policy. Given a function $f : \mathcal{I}_Q \rightarrow \mathbb{R}^d$, outputting $f(D) + \eta$ ensures (ϵ, P)-Blowfish privacy if $\eta \in \mathbb{R}^d$ is a vector of independent random numbers drawn from $Lap(S(f, p)/\epsilon)$.*

When policies do not have constraints $(P = (\mathcal{T}, G, \mathcal{I}_n))$, (ϵ, P)-Blowfish differs from ϵ-differential privacy only in the specification of sensitive information. Note that every pair $(D_1, D_2) \in N(P)$ differ in only one tuple when P has no constraints. Therefore, the following result trivially holds.

Lemma 5.2. *Any mechanism M that satisfies ϵ-differential privacy also satisfies $(\epsilon, (\mathcal{T}, G, \mathcal{I}_n))$-Blowfish privacy for all discriminative secret graphs G.*

The proof follows from the fact that ϵ-differential privacy is equivalent to $(\epsilon, (\mathcal{T}, K, \mathcal{I}_n))$-Blowfish privacy, where K is the complete graph.

In many cases, we can do better in terms of utility than differentially privacy mechanisms. It is easy to see that $S(f, P)$ is never larger than the global sensitivity $S(f)$. Therefore, just using the Laplace mechanism with $S(f, P)$ can provide better utility.

For instance, consider a linear sum query $f_{\mathbf{w}} = \sum_{i=1}^{n} w_i x_i$, where $\mathbf{w} \in \mathbb{R}^n$ is a weight vector, and each value $x_i \in \mathcal{T} = [a, b]$. For G^{full}, the policy specific sensitivity is $(b - a) \cdot (\max_i w_i)$ the same as the global sensitivity. For $G^{d,\theta}$, where $d(x, y) = |x-y|$, the policy specific sensitivity is $\theta \cdot (\max_i w_i)$, which can be much smaller than the global sensitivity when $\theta \ll (b - a)$.

As a second example, suppose $\mathcal{P}$ is a partitioning of the domain. If the policy specifies sensitive information partitioned by $\mathcal{P}$ $(G^{\mathcal{P}})$, then the policy specific sensitivity of $h_{\mathcal{P}}$ is 0. That is, the histogram of $\mathcal{P}$ or any coarser partitioning can be released without any noise. We will show more examples of improved utility under Blowfish policies in Sec 5.

However, for histogram queries, the policy specific sensitivity for most reasonable policies (with no constraints) is 2, the same as global sensitivity.[4] Thus, it cannot significantly improve the accuracy for histogram queries.

Next, we present examples of two analysis tasks – k-means clustering (Section 6), and releasing cumulative histograms (Section 7)– for which we can design mechanisms for Blowfish policies without constraints with more utility (lesser error) than mechanisms that satisfy differential privacy. In k-means clustering we will see that using Blowfish policies helps reduce the sensitivity of intermediate queries on the data. In the case of the cumulative histogram workload, we can identify novel query answering strategies given a Blowfish policy that helps reduce the error.

6. K-MEANS CLUSTERING

K-means clustering is widely used in many applications such as classification and feature learning. It aims to cluster proximate data together and is formally defined below.

Definition 6.1 (K-means clustering). *Given a data set of n points $(t_1, ..., t_n) \in \mathcal{T}^n$, k-means clustering aims to partition the points into $k \leq n$ clusters $S = \{S_1, ..., S_k\}$ in order to minimize*

$$\sum_{i=1}^{k} \sum_{t_j \in S_i} ||t_j - \mu_i||^2, \tag{10}$$

where $\mu_i = \frac{1}{|S_i|} \sum_{t_j \in S_i} t_j$, and $||x - y||$ denotes L_2 distance.

The non-private version of k-means clustering initializes the means/centroids $(\mu_1, ..., \mu_k)$ (e.g. randomly) and updates them iteratively as follows: 1) assign each point to the nearest centroid; 2) recompute the centroid of each cluster, until reaching some convergence criterion or a fixed number of iterations.

The first differentially private k-means clustering algorithm was proposed by Blum et al. [2] as SuLQ k-means. Observe that only two queries are required explicitly: 1) the number of points in each new cluster, $q_{size} = (|S_1|, ..., |S_k|)$ and 2) the sum of the data points for each cluster, $q_{sum} = (\sum_{t_j \in S_1} t_j, ..., \sum_{t_j \in S_k} t_j)$, to compute the centroid. The sensitivity of q_{size} is 2 (same as a histogram query). Let $d(\mathcal{T})$ denote the diameter of the domain, or the largest L_1 distance $(||x - y||_1)$ between any two points $x, y \in \mathcal{T}$. The sensitivity of q_{sum} could be as large as the diameter $2 \cdot d(\mathcal{T})$ since a tuple from x to y can only change the sums for two clusters by at most $d(\mathcal{T})$.

Under Blowfish privacy policies, the policy specific sensitivity of q_{sum} can be much smaller than $|\mathcal{T}|$ under Differential privacy (i.e. complete graph G^{full} for Blowfish policies). Since q_{size} is the histogram query, the sensitivity of q_{size} under Blowfish is also 2.

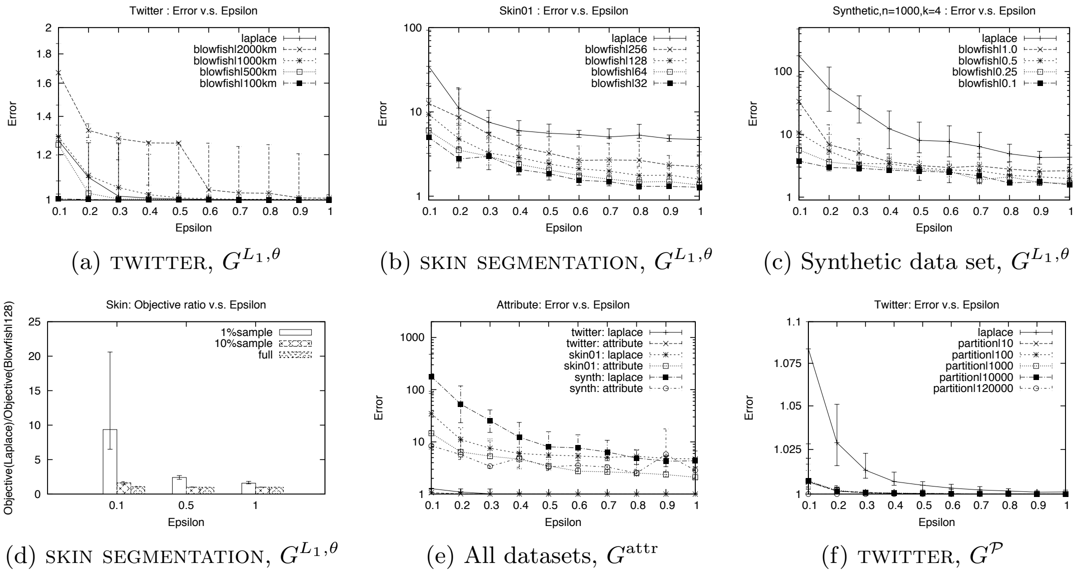
Lemma 6.1. *Policy specific global sensitivities of q_{sum} under the attribute G^{attr}, L_1-distance $G^{(L_1,\theta)}$, and partition $G^{\mathcal{P}}$ discriminative graphs (from Section 3) are smaller than the global sensitivity of q_{sum} under differential privacy.*

Proof. First, in the attribute discriminative graph G^{attr}, edges correspond to $(x, y) \in \mathcal{T} \times \mathcal{T}$ that differ only in any one attribute. Thus, if $|A|$ denotes maximum distance between two elements in A, then the policy specific sensitivity of q_{sum} under G^{attr} is $\max_A(2 \cdot |A|) < 2 \cdot d(\mathcal{T})$. Next, suppose we use $G^{L_1,\theta}$, where $x, y \in \mathcal{T}$ are connected by an edge if $||x - y||_1 \leq \theta$. Thus, policy specific sensitivity of q_{sum} is 2θ.

Finally, consider the policy specified using the partitioned sensitive graph $G^{\mathcal{P}}$, where $\mathcal{P} = \{P_1, P_2, \ldots, P_k\}$ is some data independent partitioning of the domain $\mathcal{T}$. Here, an adversary should not distinguish between an individual's tuple taking a pair of values $x, y \in \mathcal{T}$ only if x and y appear in the same partition P_i for some i. Under this policy the sensitivity of q_{sum} is at most $\max_{P \in \mathcal{P}} 2 \cdot d(P) < 2 \cdot d(\mathcal{T})$. □

Thus, by Theorem 5.1, we can use the SULQ k-means mechanism with the appropriate policy specific sensitivity for q_{sum} (from Lemma 6.1) and thus satisfy privacy under the Blowfish policy while ensuring better accuracy.

[4] The one exception is partitioned sensitive information.



(a) TWITTER, $G^{L_1,\theta}$

(b) SKIN SEGMENTATION, $G^{L_1,\theta}$

(c) Synthetic data set, $G^{L_1,\theta}$

(d) SKIN SEGMENTATION, $G^{L_1,\theta}$

(e) All datasets, G^{attr}

(f) TWITTER, $G^{\mathcal{P}}$

Figure 1: K-Means: Error under Laplace mechanism v.s Blowfish privacy for different discriminative graphs.

6.1 Empirical Evaluation

We empirically evaluate the accuracy of k-means clustering for $(\epsilon, (\mathcal{T}, G, \mathcal{I}_n))$-Blowfish privacy on three data sets. The first two datasets are real-world datasets – TWITTER and SKIN SEGMENTATION[5]. The TWITTER data set consists of a total of 193563 tweets collected using Twitter API that all contained a latitude/longitude within a bounding box of $50N, 125W$ and $30N, 110W$ (western USA) – about 2222×1442 square km. By setting the precision of latitude/longitude coordinates to be 0.05, we obtain a 2D domain of size 400×300. The SKIN SEGMENTATION data set consists of 245057 instances. Three ordinal attributes are considered and they are B, G, R values from face images of different classes. Each of them has a range from 0 to 255. To understand the effect of Blowfish policies on datasets of different sizes, we consider the full dataset SKIN, as well as a 10% and 1% sub-sample (SKIN10, SKIN01) of the data.

The third dataset is a synthetic dataset where we generate 1000 points from $(0, 1)^4$ with k randomly chosen centers and a Gaussian noise with $\sigma(0, 0.2)$ in each direction.

In Figures 1(a)-1(d), we report the ratio of the mean of the objective value in Eqn. (10) between private clustering methods including Laplace mechanism and Blowfish privacy with $\mathcal{S}^{d,\theta}_{\text{pairs}}$, and the non-private k-means algorithm, for various values of $\epsilon = \{0.1, 0.2, ..., 0.9, 1.0\}$. For all datasets, $d(\cdot)$ is L_1 (or Manhattan) distance. The number of iterations is fixed to be 10 and the number of clusters is $k = 4$. Each experiment is repeated 50 times to find mean, lower and upper quartile. Figure 1(a) clusters according to latitude/longitude of each tweet. We consider five different policies: G^{full} (Laplace mechanism), $G^{d,2000km}$, $G^{d,1000km}$, $G^{d,500km}$, $G^{d,100km}$. Here, $\theta = 100$km means that the adversary cannot distinguish locations within a 20000 square km region. Figure 1(b) clusters the 1% subsample SKIN01 based on three attributes: B, G, R values and considers 5 policies as well: G^{full}, $G^{d,256}$, $G^{d,128}$, $G^{d,64}$ and $G^{d,32}$. Lastly, we also consider five policies for the synthetic dataset in Figure 1(c): G^{full}, $G^{d,1.0}$, $G^{d,0.5}$, $G^{d,0.25}$, $G^{d,0.1}$.

From Figures 1(a)-1(c), we observe that the objective value of Laplace mechanism could deviate up to 100 times away from non-private method, but under Blowfish policies objective values could be less than 5 times that for non-private k-means. Moreover, the error introduced by Laplace mechanism becomes larger with higher dimensionality – the ratio for Laplace mechanism in Figure Figure 1(c) and 1(b) (4 and 3 dimensional resp.) is much higher than that in the 2D TWITTER dataset in Figure 1(a). From Figure 1(b), we observe that the error introduced by private mechanisms do not necessarily reduce monotonically as we reduce Blowfish privacy protection (i.e. reduce θ). The same pattern is observed in Figure 1(a) and Figure 1(c). One possible explanation is that adding a sufficient amount of noise could be helpful to get out of local minima for clustering, but adding too much noise could lead to less accurate results.

To study the interplay between dataset size and Blowfish, we plot (Figure 1(d)) for SKIN, SKIN10 and SKIN01 the ratio of the objective value attained by the Laplace method to the objective value attained by one of the Blowfish policies: $G^{d,128}$. In all cases, we see an improvement in the objective under Blowfish. The improvement in the objective is smaller for larger ϵ and larger datasets (since the Laplace mechanism solution is close to the non-private solutions on SKIN).

Finally, Figures 1(e) and 1(f) summarize our results on the G^{attr} and $G^{\mathcal{P}}$ discriminative graphs. Figure 1(e) shows that under the G^{attr} Blowfish policy, the error decreases by an order of magnitude compared to the Laplace mechanism for SKIN01 and the synthetic dataset due to higher dimensionality and small dataset size. On the other hand, there is little gain by using G^{attr} for the larger 2D TWITTER dataset.

Figure 1(f) shows ratio of the objective attained by the private methods to that of the non-private k-means under $G^{\mathcal{P}}$, for partitions $\mathcal{P}$ of different sizes. In each case, the 300x400 grid is uniformly divided; e.g., in `partition|100`, we consider a uniform partitioning in 100 coarse cells, where

[5] `http://archive.ics.uci.edu/ml/datasets/Skin+Segmentation`

each new cell contains 30x40 cells from the original grid. Thus an adversary will not be able to tell whether an individual's location was within an area spanned by the 30x40 cells (about 36,300 sq km). `partition|120000` corresponds to the original grid; thus we only protects pairs of locations within each cell in the original grid (about 30 sq km). We see that the objective value for Blowfish policies are smaller than the objective values under Laplace mechanisms, suggesting more accurate clustering. We also note that under `partition|120000`, we can do the clustering exactly, since the sensitivity of both q_{size} and q_{sum} are 0.

To summarize, Blowfish policies allow us to effectively improve utility by trading off privacy. In certain cases, we observe that Blowfish policies attain an objective value that is close to 10 times smaller than that for the Laplace mechanism. The gap between Laplace and Blowfish policies increases with dimensionality, and reduces with data size.

7. CUMULATIVE HISTOGRAMS

In this section, we develop novel query answering strategies for two workloads – cumulative histograms and range queries. Throughout this section, we will use Mean Squared Error as a measure of accuracy/error defined in Def 2.4.

Definition 7.1 (Cumulative Histogram). *Consider a domain $\mathcal{T} = \{x_1, ..., x_{|\mathcal{T}|}\}$ that has a total ordering $x_1 \leq ... \leq x_{|\mathcal{T}|}$. Let $c(x_i)$ denote the number of times x_i appears in the database D. Then, the cumulative histogram of $\mathcal{T}$, denoted by $S_{\mathcal{T}}(\cdot)$ is a sequence of cumulative counts*

$$\left\{ s_i \mid s_i = \sum_{j=1}^{i} c(x_j), \forall i = 1, ..., |\mathcal{T}| \right\} \tag{11}$$

Since we know the total size of the dataset $|D| = n$, dividing each cumulative count in $S_{\mathcal{T}}(\cdot)$ by n gives us the cumulative distribution function (CDF) over $\mathcal{T}$. Releasing the CDF has many applications including computing quantiles and histograms, answering range queries and constructing indexes (e.g. k-d tree). This motivates us to design a mechanism for releasing cumulative histograms.

The cumulative histogram has a global sensitivity of $|\mathcal{T}| - 1$ because all the counts in cumulative histogram except $s_{|\mathcal{T}|}$ will be reduced by 1 when a record in D changes from x_1 to $x_{|\mathcal{T}|}$. Similar to k-means clustering, we could reduce the sensitivity of cumulative histogram $S_{\mathcal{T}}(\cdot)$ by specifying the sensitive information, such as $\mathcal{S}^{\mathcal{P}}_{\text{pairs}}$ and $\mathcal{S}^{d,\theta}_{\text{pairs}}$. For this section, we focus on $\mathcal{S}^{d,\theta}_{\text{pairs}}$, where $d(\cdot)$ is the $L1$ distance on the domain and we assume that all the domains discussed here have a total ordering.

7.1 Ordered Mechanism

Let us first consider a policy $P_\theta = (\mathcal{T}, G^{d,\theta}, \mathcal{I}_n)$ with $\theta = 1$. The discriminative secret graph is a line graph, $G^{d,1} = (V, E)$, where $V = \mathcal{T}$ and $E = \{(x_i, x_{i+1}) | \forall i = 1, ..., |\mathcal{T}| - 1\}$. This means that only adjacent domain values $(x_i, x_{i+1}) \in \mathcal{T} \times \mathcal{T}$ can form a secret pair. Therefore, the policy specific sensitivity of $S_{\mathcal{T}}(\cdot)$ for a line graph is 1. Based on this small sensitivity, we propose a mechanism, named Ordered Mechanism $M^O_{G^{d,1}}$ to perturb cumulative histogram $S_{\mathcal{T}}(\cdot)$ over line graph $G^{d,1}$ in the following way. For each s_i, we add $\eta_i \sim Laplace(\frac{1}{\epsilon})$ to get $\tilde{s}_i$ to ensure (ϵ, P)-Blowfish privacy. Each $\tilde{s}_i$ has an error with an expectation equals to $\frac{2}{\epsilon^2}$. Note that Theorem 5.1 already ensures that releasing $\tilde{s}_i$'s satisfies (ϵ, P_1)-Blowfish privacy. Furthermore, observe that the counts in $S_{\mathcal{T}}(\cdot)$ are in ascending order. Hence, we can boost the accuracy of $\tilde{S}_{\mathcal{T}}(\cdot)$ using constrained inference proposed by Hay et al. in [9]. In this way, the new cumulative histogram, denoted by $\hat{S}_{\mathcal{T}}(\cdot)$, satisfies the ordering constraint and has an error $\mathcal{E}_{\hat{S}} = O(\frac{p \log^3 |\mathcal{T}|}{\epsilon^2})$, where p represents the number of distinct values in $\hat{S}_{\mathcal{T}}(\cdot)$ [9]. Note that, if we additionally enforce the constraint that $s_1 > 0$, then all the counts are also positive. In particular, when $p = 1$, $\mathcal{E}_{\hat{S}} = O(\frac{\log^3 |\mathcal{T}|}{\epsilon^2})$ and when $p = |\mathcal{T}|$, $\mathcal{E}_{\hat{S}} = O(\frac{|\mathcal{T}|}{\epsilon^2})$. Many real datasets are sparse, i.e. the majority of the domain values have zero counts, and hence have fewer distinct cumulative counts, i.e. $p \ll |\mathcal{T}|$. This leads to much smaller $\mathcal{E}_{\hat{S}}$ compared to $\mathcal{E}_{\tilde{S}}$. The best known strategy for releasing the cumulative histogram is using the hierarchical mechanism [9], which results in a total error of $O(\frac{|\mathcal{T}| \log^3 |\mathcal{T}|}{\epsilon^2})$. Moreover, the SVD bound [16] suggests that no strategy can release the cumulative histogram with $O(\frac{|\mathcal{T}|}{\epsilon^2})$ error. Thus under the line graph policy, the Ordered Mechanism is a much better strategy for cumulative histogram.

One important application of cumulative histogram is answering range query, defined as follows.

Definition 7.2 (Range Query). *Let D has domain $\mathcal{T} = \{x_1, ..., x_{|\mathcal{T}|}\}$, where $\mathcal{T}$ has a total ordering. A range query, denoted by $q[x_i, x_j]$ counts the number of tuples falling within the range $[x_i, x_j]$ where $x_i, x_j \in \mathcal{T}$ and $x_i \leq x_j$.*

Range queries can be directly answered using cumulative histogram $\hat{S}_{\mathcal{T}}(\cdot)$, $q[x_i, x_j] = \hat{s}_j - \hat{s}_{i-1}$. As each range query requires at most two noisy cumulative counts, it has an error smaller than $2 \cdot \frac{2}{\epsilon^2}$ (even without constrained inference). Hence, we have the following theorem.

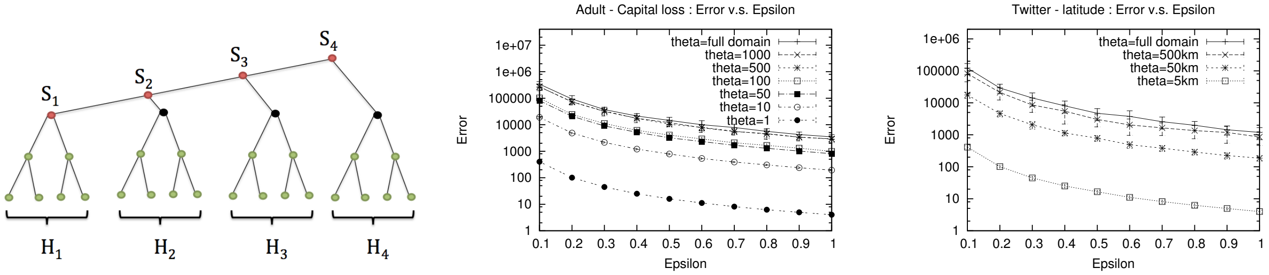
Theorem 7.1. *Consider a policy $(\mathcal{T}, G^{d,1}, \mathcal{I}_n)$, where $G^{d,1}$ is a line graph. Then the expected error of a range query $q[x_i, x_j]$ for Ordered Mechanism is given by:*

$$\mathcal{E}_{q[x_i,x_j], M^O_{G^{d,1}}} \leq 4/\epsilon^2 \tag{12}$$

This error bound is independent of $|\mathcal{T}|$, much lower than the expected error using hierarchical structure with Laplace mechanism to answer range queries, $\mathcal{E}_{q[x_i,x_j],lap} = \frac{\log^3 |\mathcal{T}|}{\epsilon^2}$. Again, the SVD bound [16] suggests that no differentially private strategy can answer each range query with $O(\frac{1}{\epsilon^2})$ error. Other applications of $S_{\mathcal{T}}(\cdot)$ including computing quantiles and histograms and constructing indexes (e.g. k-d tree) could also use cumulative histogram in a similar manner as range query to obtain a much smaller error by trading utility with privacy under $G^{d,1}$. Next, we describe the *ordered hierarchical mechanism* that works for general graphs, $G^{d,\theta}$.

7.2 Ordered Hierarchical Mechanism

For a more general graph $G^{d,\theta} = (V, E)$, where $V = \mathcal{T}$ and $E = \{(x_i, x_{i\pm1}), .., (x_i, x_{i\pm\theta}) | \forall i = 1, ..., |\mathcal{T}|\}$, the sensitivity of releasing cumulative histogram $S_{\mathcal{T}}(\cdot)$ becomes θ. The Ordered Mechanism would add noise from $Lap(\frac{\theta}{\epsilon})$ to each cumulative counts s_i. The total error in the released cumulative histogram and range queries would still be asymptotically smaller than the error achieved by any differentially private mechanism for small θ. However, the errors become comparable as the θ reaches $\log |\mathcal{T}|$, and the Ordered Mechanism's error exceeds the error from the hierarchical mechanism when $\theta = O(\log^{3/2} |\mathcal{T}|)$. In this section, we present



(a) Ordered Hierarchical Tree $\theta = 4$ (b) Adult - capital loss (c) Twitter - latitude

Figure 2: Ordered Hierarchical Mechanism. 2(a) gives an example of OH, where $\theta = 4$. 2(b) and 2(c) shows privacy-utility trade-offs for range query, using $G^{d,\theta}$ for sensitive information.

a hybrid strategy for releasing cumulative histograms (and hence range queries), called Ordered Hierarchical Mechanism, that always has an error less than or equal to the hierarchical mechanism for all θ.

Various hierarchical methods have been proposed in the literature [9, 19, 15, 20, 18]. A basic hierarchical structure is usually described as a tree with a regular fan-out f. The root records the total size of the dataset D, i.e. the answer to the range query $q[x_1, x_{|\mathcal{T}|}]$. This range is then partitioned into f intervals. If $\delta = \lceil \frac{|\mathcal{T}|}{f} \rceil$, the intervals are $[x_1, x_\delta]$, $[x_{\delta+1}, x_{2\delta}]$,..., $[x_{|\mathcal{T}|-\delta+1}, x_{|\mathcal{T}|}]$ and answers to the range queries over those intervals are recorded by the children of the root node. Recursively, the interval represented by the current node will be further divided into f subintervals. Leaf nodes correspond to unit length interval $q[x_i, x_i]$. The height of the tree is $h = \lceil \log_f |\mathcal{T}| \rceil$. In the above construction, the counts at level i are released using the Laplace mechanism with parameter $\frac{2}{\epsilon}$, and $\sum_i \epsilon_i = \epsilon$. Prior work has considered distributing the ϵ uniformly or geometrically [5]. We use uniform budgeting in our experiments.

Inspired by ordered mechanism for line graph, we propose a hybrid structure, called Ordered Hierarchical Structure OH for $(\epsilon, (\mathcal{T}, G^{d,\theta}, \mathcal{I}_n))$-Blowfish privacy. As shown in Figure 2(a), OH has two types of nodes, S nodes and H nodes. The number of S nodes is $k = \lceil \frac{n}{\theta} \rceil$, which is dependent on the threshold θ. In this way, we could guarantee a sensitivity of 1 among the S nodes. Let us represent S nodes as $s_1, ..., s_k$, where $s_1 = q[x_1, x_\theta]$,..., $s_{k-1} = q[x_1, x_{(k-1)\theta}]$, $s_k = q[x_1, x_{|\mathcal{T}|}]$. Note that the s_i nodes here are not the same as the count for cumulative histogram, so we will use range query $q[x_1, x_i]$ to represent the count s_i in a cumulative histogram. The first S node, s_1 is the root of a subtree consisting of H nodes. This subtree is denoted by H_1 and is used for answering all possible range queries within this interval $[x_1, x_\theta]$. For all $1 < i \leq k$, s_i has two children: s_{i-1} and the root of a subtree made of H nodes, denoted by H_i. Similarly, it also has a fan-out of f and represents counts for values $[(i-1)\theta + 1, i\theta]$. We denote the height of the subtree by $h = \lceil \log_f \theta \rceil$. Using this hybrid structure, we could release cumulative counts in this way: $q[x_1, x_{l\theta}] + q[x_{l\theta+1}, x_j]$, where $l\theta \leq j < (l+1)\theta$. Here $q[x_1, x_{l\theta}]$ is answered using s_l and $q[x_{l\theta+1}, x_j]$ is answered using H_l. Then any range query could be answered as $q[x_i, x_j] = q[x_1, x_j] - q[x_1, x_{i-1}]$.

Privacy Budgeting Strategy Given total privacy budget ϵ, we denote the privacy budget assigned to all the S nodes by ϵ_S and to all the H nodes by ϵ_H. When a tuple change its value from x to y, where $d_G(x, y) \leq \theta$, at most one S node changes its count value and at most $2h$ H nodes change their count values. Hence, for $i = 2, ..., k$, we add Laplace noise drawn from $Lap(\frac{1}{\epsilon_S})$ to each s_i and we add Laplace noise drawn from $Lap(\frac{2h}{\epsilon_H})$ to each H node in the subtree H_i. As S_1 is the root of H_1, we assign $\epsilon = \epsilon_S + \epsilon_H$ to the tree H_1 and hence we add Laplace noise drawn from $Lap(\frac{2h}{\epsilon_H + \epsilon_S})$ to each H node in H_1, including s_1. In this way, we could claim that this OH tree satisfies $(\epsilon, (\mathcal{T}, G^{d,\theta}, \mathcal{I}_n))$-Blowfish privacy. When $\theta = |\mathcal{T}|$, H_1 is going to be the only tree to have all the privacy budget. This is equivalent to the hierarchical mechanism for differential privacy.

THEOREM 7.2. *Consider a policy $P = (\mathcal{T}, G^{d,\theta}, \mathcal{I}_n)$. (1) The Ordered Hierarchical structure satisfies (ϵ, P)-Blowfish privacy. (2) The expected error of releasing a single count in cumulative histogram or answering a range query this structure over $\mathcal{T}$ is ,*

$$\mathcal{E}_{q[x_i,x_j], M^{OH}_{G^{d,\theta}}} = O\left(\frac{|\mathcal{T}| - \theta}{|\mathcal{T}|\epsilon_S^2} + \frac{(f-1)\log_f^3 \theta}{\epsilon_H^2}\right) \tag{13}$$

PROOF. (sketch) 1) Increasing count of x by 1 and decreasing count of y by 1, where $d(x, y) < \theta$ only affect the counts of at most one S node and $2h$ H nodes. Since we draw noise from $Lap(\frac{1}{\epsilon_S})$ for S nodes and from $Lap(\frac{2h}{\epsilon_H})$ for H nodes (where $\epsilon = \epsilon_S + \epsilon_H$), we get (ϵ, P)-Blowfish privacy based on sequential composition.
2) Consider all the counts in cumulative histogram, and there are $|\mathcal{T}|$ of them, only $|\mathcal{T}| - \theta$ requires S nodes. Each S node has a error of $\frac{2}{\epsilon_S^2}$. This gives the first fraction in Eqn. (13). On average, the number of H nodes used for each count in cumulative histogram is bounded by the height of the H tree and each H node has an error of $\frac{8h^2}{\epsilon_H^2}$, which explains the second fraction in Eqn. (13). □

Each range query $q[x_i, x_j]$ requires at most 2 counts from cumulative histogram and its exact form of expected error is shown below,

$$\mathcal{E}_{q[x_i,x_j], M^{OH}_{G^{d,\theta}}} = \frac{c_1}{\epsilon_S^2} + \frac{c_2}{\epsilon_H^2}, \tag{14}$$

where $c_1 = \frac{4(|\mathcal{T}| - \theta)}{|\mathcal{T}|+1}$ and $c_2 = \frac{8(f-1)\log_f \theta^3 |\mathcal{T}|}{|\mathcal{T}|+1}$. Given θ and

f, at $\epsilon_S^* = \frac{c_1^{1/3}}{c_1^{1/3}+c_2^{1/3}}\epsilon$, we obtain the minimum

$$\mathcal{E}^*_{q[x_i,x_j],M^{OH}_{G^{d,\theta}}} = \frac{(c_1^{1/3}+c_2^{1/3})^3}{\epsilon^2} \quad (15)$$

In particular, when $\theta = |\mathcal{T}|$, $c_1 = 0$, this is equivalent to the classical hierarchical mechanism without S nodes and we have $\mathcal{E}_{q[x_i,x_j],M^{OH}_{G^{d,\theta}}} = O(\frac{\log^3|\mathcal{T}|}{\epsilon^2})$. When $\theta = 1$, $c_2 = 0$, this is the pure Ordered Mechanism without using H nodes and $\mathcal{E}_{q[x_i,x_j],M^{OH}_{G^{d,\theta}}} = O(\frac{1}{\epsilon^2})$.

Complexity Analysis The complexity of construction of the hybrid tree OH and answering range query are $O(|\mathcal{T}|)$ and $O(\log\theta)$ respectively, where $\theta \leq |\mathcal{T}|$, which is not worse than the classical hierarchical methods.

7.3 Empirical Evaluation

We empirically evaluate the error of range queries for the Ordered Hierarchical Mechanism with $(\epsilon, (\mathcal{T}, G^{d,\theta}, \mathcal{I}_n))$-Blowfish privacy on two real-world datasets – ADULT and TWITTER. The ADULT data set[6] consists of Census records of 48842 individuals. We consider the ordinal attribute capital loss with a domain size of 4357. The TWITTER data set is the same dataset used for k-means clustering (Sec 6.1). Here, in order to have a total ordering for the dataset, we project the TWITTER data set on its latitude with a domain size of 400, around 2222 km. The fan-out f is set to be 16 and each experiment is repeated 50 times. Figure 2(b) shows the mean square error $\mathcal{E}$ of 10000 random range queries for various values of $\epsilon = \{0.1, 0.2, ..., 0.9, 1.0\}$. Seven threshold values $\theta = \{full, 1000, 500, 100, 50, 10, 1\}$ are considered. For ADULT, for example, $\theta = 100$ means the adversary cannot distinguish between values of capital loss within a range of 100 and $\theta = full$ means the adversary cannot distinguish between all the domain values (same as differential privacy). Figure 2(c) considers 4 threshold values $\theta = \{full, 500km, 50km, 5km\}$ for TWITTER. When $\theta = 1$ (ADULT) or $\theta = 5km$ (TWITTER), the ordered hierarchical mechanism is same as the ordered mechanism. From both figures, we see that as the θ increases, $\mathcal{E}$ decreases and orders of magnitude difference in error between $\theta = 1$ and $\theta = |\mathcal{T}|$.

8. BLOWFISH WITH CONSTRAINTS

In this section, we consider query answering under Blowfish policies with constraints $P = (\mathcal{T}, G, \mathcal{I}_Q)$, where $(\mathcal{I}_Q \subsetneq \mathcal{I}_n)$. In the presence of general deterministic constraints Q, pairs of neighboring databases can differ in any number of tuples, depending on structures of Q and the discriminative graph G. Computing the policy specific sensitivity in this general case is a hard problem, as shown next.

THEOREM 8.1. *Given a function f and a policy $P = (\mathcal{T}, G, \mathcal{I}_Q)$. Checking whether $S(f, P) > 0$ is NP-hard. The same is true for the complete histogram query h.*

PROOF. (sketch) The proof follows from the hardness of checking whether 3SAT has at least 2 solutions. The reduction uses $\mathcal{I} = \{0,1\}^n$, constraints corresponding to clauses in the formula, and $\{s_0^i, s_1^i\}_i$ as secret pairs. □

Theorem 8.1 implies that checking whether $S(f, P) \leq z$ is co-NP-hard for general constraints Q. In fact, the hardness result holds even if we just consider the histogram query h and general *count query constraints*.

Hence, in the rest of this section, we will focus on releasing histograms under a large subclass of constraints called *sparse count query constraints*. In Section 8.1 we show that when the count query constraint is "sparse", we can efficiently compute $S(h, P)$, and thus we can use the Laplace mechanism to release the histogram. In Section 8.2, we will show that our general result about $S(h, P)$ subject to sparse count query constraints can be applied to several important practical scenarios.

8.1 Global Sensitivity for Sparse Constraints

A *count query* q_ϕ returns the number of tuples satisfying predicate ϕ in a database D, i.e., $\mathsf{q}_\phi(D) = \sum_{t\in D}\mathbf{1}_{\phi(t)=\text{true}}$. The auxiliary knowledge we consider here is a *count query constraint* Q, which can be expressed as a conjunction of query-answer pairs:

$$\mathsf{q}_{\phi_1}(D) = \mathsf{cnt}_1 \wedge \mathsf{q}_{\phi_2}(D) = \mathsf{cnt}_2 \wedge \ldots \wedge \mathsf{q}_{\phi_p}(D) = \mathsf{cnt}_p. \quad (16)$$

Since the answers $\mathsf{cnt}_1, \mathsf{cnt}_2, \ldots, \mathsf{cnt}_p$ do not affect our analysis, we denote the auxiliary knowledge or count query constraint as $Q = \{\mathsf{q}_{\phi_1}, \mathsf{q}_{\phi_2}, \ldots, \mathsf{q}_{\phi_p}\}$. Note that this class of auxiliary knowledge is already very general and commonly seen in practice. For example, marginals of contingency tables, range queries, and degree distributions of graphs can all be expressed in this form.

Even for this class of constraints, calculating $S(h, P)$ is still hard. In fact, the same hardness result in Theorem 8.1 holds for count query constraints (using a reduction from the Vertex Cover problem).

8.1.1 Sparse Auxiliary Knowledge

Consider a secret pair $(s_x^i, s_y^i) \in \mathcal{S}^G_{\text{pairs}}$ about a tuple t with $t.\_id = i$, and a count query $\mathsf{q}_\phi \in Q$. If the tuple $t \in D$ changes from x to y, there are three mutually exclusive cases about $\mathsf{q}_\phi(D)$: i) increases by one $(\neg\phi(x) \wedge \phi(y))$, ii) decreases by one $(\phi(x) \wedge \neg\phi(y))$, or iii) stays the same (otherwise).

DEFINITION 8.1 (LIFT AND LOWER). *A pair $(x, y) \in \mathcal{T} \times \mathcal{T}$ is said to lift a count query q_ϕ iff $\phi(x) = \text{false} \wedge \phi(y) = \text{true}$, or lower q_ϕ iff $\phi(x) = \text{true} \wedge \phi(y) = \text{false}$.*

Note that one pair may lift or lower many count queries simultaneously. We now define sparse auxiliary knowledge.

DEFINITION 8.2 (SPARSE KNOWLEDGE). *The auxiliary knowledge $Q = \{\mathsf{q}_{\phi_1}, \mathsf{q}_{\phi_2}, \ldots, \mathsf{q}_{\phi_p}\}$ is* sparse *w.r.t. the discriminative secret graph $G = (V, E)$, iff each pair $(x, y) \in E$ lifts at most one count query in Q and lowers at most one count query in Q.*

EXAMPLE 8.1. (Lift, Lower, and Sparse Knowledge) *Consider databases from domain $\mathcal{T} = A_1 \times A_2 \times A_3$, where $A_1 = \{a_1, a_2\}$, $A_2 = \{b_1, b_2\}$, and $A_3 = \{c_1, c_2, c_3\}$ and count query constraint $Q = \{\mathsf{q}_1, \mathsf{q}_2, \mathsf{q}_3, \mathsf{q}_4\}$ as in Figure 3(a). With full-domain sensitive information, any pair in $\mathcal{T} \times \mathcal{T}$ is a discriminative secret and thus the discriminative secret graph G is a complete graph. A pair $((a_1, b_1, c_1), (a_2, b_2, c_2))$ lifts q_4 and lowers q_1; and a pair $((a_1, b_2, c_1), (a_1, b_2, c_2))$ neither lifts nor lowers a query. We can verify every pair either (i) lifts exactly one query in Q and lowers exactly one in Q, or (ii) lifts or lowers no query in Q. So Q is sparse w.r.t. the discriminative secret graph G.*

[6] http://mlr.cs.umass.edu/ml/datasets/Adult

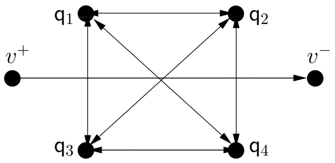
$\mathsf{q}_1: \ t.A_1 = a_1 \wedge t.A_2 = b_1$

$\mathsf{q}_2: \ t.A_1 = a_1 \wedge t.A_2 = b_2$

$\mathsf{q}_3: \ t.A_1 = a_2 \wedge t.A_2 = b_1$

$\mathsf{q}_4: \ t.A_1 = a_2 \wedge t.A_2 = b_2$

(a) $Q = \{\mathsf{q}_1, \mathsf{q}_2, \mathsf{q}_3, \mathsf{q}_4\}$



(b) $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$

Figure 3: Policy graph $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$ of databases with three attributes $A_1 = \{a_1, a_2\}$, $A_2 = \{b_1, b_2\}$, and $A_3 = \{c_1, c_2, c_3\}$ subject to count query constraint $Q = \{\mathsf{q}_1, \mathsf{q}_2, \mathsf{q}_3, \mathsf{q}_4\}$ and full-domain sensitive information

We will show that when the auxiliary knowledge Q is sparse w.r.t. the discriminative secret graph $G = (V, E)$ in a policy $P = (\mathcal{T}, G, \mathcal{I}_Q)$, it is *possible* to analytically bound the policy specific global sensitivity $S(h, P)$. To this end, let's first construct a directed graph called *policy graph* $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$ from P, the count queries in Q forming the vertices, and the relationships between count queries and secret pairs forming edges.

Definition 8.3 (Policy Graph). *Given a policy $P = (\mathcal{T}, G(V, E), \mathcal{I}_Q)$, and a sparse count constraint Q, the policy graph $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$ is a directed graph, where*

- *$\mathcal{V}_P = Q \cup \{v^+, v^-\}$: Create a vertex for each count query $\mathsf{q}_\phi \in Q$, and two additional special vertices v^+ and v^-.*
- *$\mathcal{E}_P$: i) add a directed edge $(\mathsf{q}_\phi, \mathsf{q}_{\phi'})$ iff there exists a secret pair $(x, y) \in E$ lifting $\mathsf{q}_{\phi'}$ and lowering q_ϕ; ii) add a directed edge (v^+, q_ϕ) iff there is a secret pair in E lifting q_ϕ but not lowering any other $\mathsf{q}_{\phi'}$; iii) add a directed edge (q_ϕ, v^-) iff there is a secret pair in E lowering q_ϕ but not lifting any other $\mathsf{q}_{\phi'}$; and iv) add edge (v^+, v^-).*

Let $\alpha(\mathcal{G}_P)$ denote the length (number of edges) of the longest simple cycle in $\mathcal{G}_P$. $\alpha(\mathcal{G}_P)$ is defined to be 0 if $\mathcal{G}_P$ has no directed cycle. Let $\xi(\mathcal{G}_P)$ be the length (number of edges) of a longest simple path from v^+ to v^- in $\mathcal{G}_P$.

Example 8.2. (Policy Graph) *Followed by Example 8.1, since the count query constraint Q is sparse w.r.t. the discriminative secret graph, we have its policy graph $\mathcal{G}_P = (\mathcal{V}_P, \mathcal{E}_P)$ as in Figure 3(b). For example, a pair $((a_1, b_1, c_1), (a_2, b_2, c_2))$ in $\mathcal{T} \times \mathcal{T}$ lifts q_4 and lowers q_1, so there is an edge $(\mathsf{q}_1, \mathsf{q}_4)$. There is no edge from v^+ or to v^-, except (v^+, v^-), because every pair in $\mathcal{T} \times \mathcal{T}$ either lifts one query and lowers one, or lifts/lowers no query. In this policy graph $\mathcal{G}_P$, we have $\alpha(\mathcal{G}_P) = 4$ and $\xi(\mathcal{G}_P) = 1$.*

Theorem 8.2. *Let h be the complete histogram query. In a policy $P = (\mathcal{T}, G, \mathcal{I}_Q)$, if the auxiliary knowledge Q is sparse w.r.t. G, then we have:*

$$S(h, P) \leq 2 \max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\},$$

If there exist two neighboring databases $(D_1, D_2) \in N(P)$ s.t. $||h(D_1) - h(D_2)||_1 = 2|T(D_1, D_2)|$ when $|T(D_1, D_2)| = \max_{(D', D'') \in N(P)} |T(D', D'')|$, then we have the equality:

$$S(h, P) = 2 \max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}.$$

Proof. See Appendix. □

For count query constraint Q that is sparse with respect to policy $P = (\mathcal{T}, G, \mathcal{I}_Q)$, we have the following immediate corollary about an upper bound.

Corollary 8.3. *In a policy $P = (\mathcal{T}, G, \mathcal{I}_Q)$, if Q is sparse w.r.t. G, then $S(h, P) \leq 2 \max\{|Q|, 1\}$.*

Thus, drawing noise from Laplace$(2 \max\{|Q|, 1\}/\epsilon)$ suffices (but may not be necessary) for releasing the complete histogram while ensuring (ϵ, P)-Blowfish privacy.

8.2 Applications

The problem of calculating $\alpha(\mathcal{G}_P)$ and $\xi(\mathcal{G}_P)$ exactly in a general policy graph $\mathcal{G}_P$ is still a hard problem, but becomes tractable in a number of practical scenarios. We give three such examples: i) the policy specific global sensitivity $S(h, P)$ subject to auxiliary knowledge of one marginal for full-domain sensitive information; ii) $S(h, P)$ subject to auxiliary knowledge of multiple marginals for attribute sensitive information; and iii) $S(h, P)$ subject to auxiliary knowledge of range queries for distance-threshold sensitive information.

8.2.1 Marginals and Full-domain Secrets

Marginals are also called *cuboids* in data cubes. Intuitively, in a marginal or a cuboid C, we project the database of tuples onto a subset of attributes $[C] \subseteq \{A_1, A_2, \ldots, A_k\}$ and count the number of tuples that have the same values on these attributes. Here, we consider the scenario when the adversaries have auxiliary knowledge about one or more marginals, i.e., the counts in some marginals are known.

Definition 8.4 (Marginal). *Given a database D of n tuples from a k-dim domain $\mathcal{T} = A_1 \times A_2 \times \ldots \times A_k$, a d-dim marginal C is the (exact) answer to the query:*

SELECT A_{i_1}, A_{i_2}, ..., A_{i_d}, COUNT($*$) FROM D
GROUP BY A_{i_1}, A_{i_2}, ..., A_{i_d}

Let $[C]$ denote the set of d attributes $\{A_{i_1}, A_{i_2}, \ldots, A_{i_d}\}$.

A marginal $[C] = \{A_{i_1}, A_{i_2}, \ldots, A_{i_d}\}$ is essentially a set of count queries $C^{\mathsf{q}} = \{\mathsf{q}_\phi\}$, where the predicate $\phi(t) := (t.A_{i_1} = a_{i_1}) \wedge (t.A_{i_2} = a_{i_2}) \wedge \ldots \wedge (t.A_{i_2} = a_{i_d})$, for all possible $(a_{i_1}, a_{i_2}, \ldots, a_{i_d}) \in A_{i_1} \times A_{i_2} \times \ldots \times A_{i_d}$.

Let $\mathcal{A} = \{A_1, A_2, \ldots, A_k\}$ be the set of all attributes. For a marginal C, define size$(C) = \prod_{A_i \in [C]} |A_i|$, where $|A_i|$ is the cardinality of an attribute A_i. So size(C) is the number of possible rows in the marginal C, or the number of the count queries in C^{q} constructed as above.

Suppose a marginal with $[C] \subsetneq \mathcal{A}$ is known to the adversary. Let $\mathcal{I}_{Q(C)}$ denote the set of databases with marginal C equal to certain value. Recall that we want to publish the complete histogram h of a database D from a domain $\mathcal{T} = A_1 \times A_2 \times \ldots \times A_k$. For the full-domain sensitive information, using Theorem 8.2, we have the global sensitivity equal to 2 size(C).

Theorem 8.4. *Let h be the complete histogram. For a policy $P = (\mathcal{T}, G, \mathcal{I}_{Q(C)})$, where G represents the full-domain sensitive information $\mathcal{S}^{\text{full}}_{\text{pairs}}$ and $[C] \subsetneq \mathcal{A}$ is a marginal, we have $S(h, P) = 2$ size(C).*

Proof. (sketch) Consider the set of count queries C^{q}. It is not hard to show that C^{q} is sparse w.r.t. the complete graph G. So we can construct a policy graph from $(\mathcal{T}, G, \mathcal{I}_{C^{\mathsf{q}}})$, which is a complete graph with vertex set C^{q}. From Theorem 8.2, we have $S(f, P) = 2|C^{\mathsf{q}}| = 2$ size(C). The upper bound $S(h, P) \leq 2|C^{\mathsf{q}}|$ is directly from Theorem 8.2, and it is not hard to construct two neighboring databases to match this upper bound as $[C] \subsetneq \mathcal{A}$. □

EXAMPLE 8.3. *Continuing with Example 8.2, note that the constraints in Figure 3(a) correspond to the marginal* $[C] = \{A_1, A_2\}$. *So from (i) in Theorem 8.2, we have* $S(h, P) \leq 2 \times 4 = 8$. *The worst case* $S(h, P) = 8$ *can be verified by considering the two neighboring databases* D_1 *and* D_2*, each with four rows:* $a_1b_1c_1$ *(in* D_1*)/*$a_1b_2c_2$ *(in* D_2*),* $a_1b_2c_1/a_2b_1c_2$, $a_2b_1c_1/a_2b_2c_2$, *and* $a_2b_2c_1/a_1b_1c_2$.

8.2.2 Marginals and Attribute Secrets

Now suppose a set of p marginals $C_1, \ldots, C_p$ with $[C_1], \ldots, [C_p] \subsetneq \{A_1, A_2, \ldots, A_k\}$ are auxiliary knowledge to the adversary. Let $\mathcal{I}_{Q(C_1,\ldots,C_p)}$ be the set of databases with these p marginals equal to certain values. For the attribute sensitive information, if the p marginals are disjoint, using Theorem 8.2 the global sensitivity is $2\max_{1\leq i\leq p} \mathrm{size}(C_i)$.

THEOREM 8.5. *Let h be the complete histogram. Consider a policy* $P = (\mathcal{T}, G^{\mathrm{attr}}, \mathcal{I}_{Q(C_1,\ldots,C_p)})$*, where* $[C_i] \subsetneq \mathcal{A}$ *for any marginal* C_i*, and* $[C_i] \cap [C_j] = \emptyset$ *for any two* C_i *and* C_j*. Then we have* $S(h, P) = 2\max_{1\leq i\leq p} \mathrm{size}(C_i)$.

PROOF. (sketch) Consider the set of count queries $Q = C_1^{\mathsf{q}} \cup \ldots \cup C_p^{\mathsf{q}}$, it is not hard to show that Q is sparse w.r.t. G^{attr}. The policy graph from $(\mathcal{T}, G^{\mathrm{attr}}, Q)$ is the union of p cliques with vertex sets $C_1^{\mathsf{q}}, \ldots, C_p^{\mathsf{q}}$. From Theorem 8.2, we have $S(h, P) = 2\max_i |C_i^{\mathsf{q}}| = 2\max_i \mathrm{size}(C_i)$. The upper bound $S(h, P) \leq 2\max_i |C_i^{\mathsf{q}}|$ is directly from Theorem 8.2, and it is not hard to construct two neighboring databases to match this upper bound as $[C_i] \neq \mathcal{A}$. □

8.2.3 Grid and Distance-threshold Secrets

Our general theorem about $S(f, P)$ can be also applied to databases with geographical information.

Consider a domain $\mathcal{T} = [m]^k$, where $[m] = \{1, 2, \ldots, m\}$. When $k = 2$ or 3, $\mathcal{T}$ can be used to approximately encode a 2-dim plane or a 3-dim space. For two points $x, y \in \mathcal{T}$, we define *distance* $d(x, y)$ to be the L^p distance $||x - y||_p$. For two point sets $X, Y \subset \mathcal{T}$, we define $d(X, Y) = \min_{x\in X, y\in Y} d(x, y)$. A geographical database D consists of n points, each of which is drawn from the domain $\mathcal{T}$ and may represent the location of an object.

Define a rectangle $R = [l_1, u_1] \times [l_2, u_2] \times \ldots \times [l_k, u_k]$, where $l_i \in [m]$, $u_i \in [m]$, and $l_i \leq u_i$. A range count query q_R returns the number of tuples whose locations fall into the rectangle R. R is called a *point query* if $l_i = u_i$ for all i.

In this scenario, suppose the answers to a set of p range count queries are known to the adversary. So we can represent the auxiliary knowledge as $Q = \{\mathsf{q}_{R_1}, \mathsf{q}_{R_2}, \ldots, \mathsf{q}_{R_p}\}$. Also, suppose we aim to protect the distance-threshold sensitive information $\mathcal{S}^{d,\theta}_{\mathrm{pairs}}= \{(s^i_x, s^i_y) \mid d(x, y) \leq \theta\}$ while the publishing complete histogram h.

Using Theorem 8.2, we can calculate the global sensitivity if all rectangles are disjoint, i.e., $R_i \cap R_j = \emptyset$ for any $i \neq j$, as follows. Construct a graph $\mathcal{G}_R(Q) = (\mathcal{V}_R, \mathcal{E}_R)$ on the set of rectangles in Q: i) create a vertex in $\mathcal{V}_R$ for rectangle R_i in each range count query q_{R_i} in Q; and ii) add an edge (R_i, R_j) into $\mathcal{E}_R$ iff $d(R_i, R_j) \leq \theta$. We can prove that the policy specific global sensitivity equals to $2(\mathrm{maxcomp}(Q)+1)$ when there are no point query constraints, where $\mathrm{maxcomp}(Q)$ is the number of nodes in the largest connected component in $\mathcal{G}_R(Q)$. Note that $\mathrm{maxcomp}(Q)$ (and hence $S(h, P)$) can be computed efficiently.

THEOREM 8.6. *Let h be the complete histogram. For a policy* $P = (\mathcal{T}, G, \mathcal{I}_Q)$*, where* $\mathcal{T} = [m]^k$*,* G *represents the distance-threshold sensitive information* $\mathcal{S}^{d,\theta}_{\mathrm{pairs}}$ $(\theta > 0)$*, and* Q *is a set of disjoint range count queries* $\{\mathsf{q}_{R_1}, \mathsf{q}_{R_2}, \ldots, \mathsf{q}_{R_p}\}$ *with* $R_i \cap R_j = \emptyset$ *for* $i \neq j$*. We have* $S(h, P) \leq 2(\mathrm{maxcomp}(Q) + 1)$*. If none of the constraints are point queries, then* $S(h, P) = 2(\mathrm{maxcomp}(Q) + 1)$.

9. CONCLUSIONS

We propose a new class of privacy definitions, called Blowfish privacy, with the goal of seeking better trade-off between privacy and utility. The key feature of Blowfish is a policy, where users can specify sensitive information that needs to be protected and knowledge about their databases which has been released to potential adversaries. Such a rich set of "tuning knobs" in the policy enable users to improve the utility by customizing sensitive information and to limit attacks from adversaries with auxiliary knowledge. Using examples of kmeans clustering, cumulative histograms and range queries, we show how to tune utility using reasonable policies with weaker specifications of privacy. For the latter, we develop strategies that are more accurate than any differentially private mechanism. Moreover, we study how to calibrate noise for Blowfish policies with count constraints when publishing histograms, and the general result we obtain can be applied in several practical scenarios.

Acknowledgements: We would like to thank Jiangwei Pan and the anonymous reviewers for their comments. This work was supported by the National Science Foundation under Grant # 1253327 and a gift from Google.

10. REFERENCES

[1] J. Blocki, A. Blum, A. Datta, and O. Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In *ACM ITCS*, 2013.

[2] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: the sulq framework. In *PODS*, 2005.

[3] K. Chatzikokolakis, M. AndrâĹŽÂl's, N. Bordenabe, and C. Palamidessi. Broadening the scope of differential privacy using metrics. In *Privacy Enhancing Technologies*. 2013.

[4] B.-C. Chen, D. Kifer, K. Lefevre, and A. Machanavajjhala. Privacy-preserving data publishing. *Foundations and Trends in Databases*, 2(1-2):1–167, 2009.

[5] G. Cormode, C. M. Procopiuc, D. Srivastava, E. Shen, and T. Yu. Differentially private spatial decompositions. In *ICDE*, pages 20–31, 2012.

[6] C. Dwork. Differential privacy. In *ICALP*, 2006.

[7] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[8] S. R. Ganta, S. P. Kasiviswanathan, and A. Smith. Composition attacks and auxiliary information in data privacy. In *KDD*, pages 265–273, 2008.

[9] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially-private queries through consistency. In *PVLDB*, pages 1021–1032, 2010.

[10] X. He, A. Machanavajjhala, and B. Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. *CoRR*, abs/1312.3913, 2014.

[11] D. Kifer and B.-R. Lin. Towards an axiomatization of statistical privacy and utility. In *PODS*, 2010.

[12] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *SIGMOD*, pages 193–204, 2011.

[13] D. Kifer and A. Machanavajjhala. A rigorous and customizable framework for privacy. In *PODS*, 2012.

[14] D. Kifer and A. Machanavajjhala. Pufferish: A framework for mathematical privacy definitions. *To appear ACM Transactions on Database Systems*, 39(1), 2014.

[15] C. Li, M. Hay, V. Rastogi, G. Miklau, and A. McGregor. Optimizing histogram queries under differential privacy. In *PODS*, pages 123–134, 2010.

[16] C. Li and G. Miklau. Optimal error of query sets under the differentially-private matrix mechanism. In *ICDT*, 2013.

[17] A. Machanavajjhala, A. Korolova, and A. D. Sarma. Personalized social recommendations - accurate or private? In *PVLDB*, volume 4, pages 440–450, 2011.

[18] W. Qardaji, W. Yang, and N. Li. Understanding hierarchical methods for differentially private histogram. In *PVLDB*, 2013.

[19] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. In *ICDE*, pages 225–236, 2010.

[20] J. Xu, Z. Zhang, X. Xiao, Y. Yang, and G. Yu. Differentially private histogram publication. In *ICDE*, pages 32–43, 2012.

APPENDIX

A. PROOF OF THEOREM 7.1

Proof. (sketch) Recall that the policy specific global sensitivity is defined as

$$S(h, P) = \max_{(D_1,D_2)\in N(P)} ||h(D_1) - h(D_2)||_1.$$

Direction I ($S(h, P) \leq 2\max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$). It suffices to prove that for any two databases $D_1, D_2 \in \mathcal{I}_Q$, if $|T(D_1, D_2)| > \max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$, there must exist another database $D_3 \in \mathcal{I}_Q$ s.t. $T(D_1, D_3) \subsetneq T(D_1, D_2)$, i.e., $(D_1, D_2) \notin N(P)$; and thus for any two databases $(D_1, D_2) \in N(P)$, we have $||h(D_1)-h(D_2)||_1 \leq 2|T(D_1, D_2)| \leq 2\max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$ which implies $S(h, P) \leq 2\max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$.

To complete the proof, we consider two databases $D_1, D_2 \in \mathcal{I}_Q$ with $|T(D_1, D_2)| > \max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$, and show how to construct the D_3 defined above.

First of all, for any secret pair $(s^i_x, s^i_y) \in T(D_1, D_2)$, it must lift and/or lower some count query $\mathsf{q}_\phi \in Q$; otherwise, we can construct D_3 by changing the value of tuple t with $t.\_id = i$ in D_1 into its value in D_2.

To construct D_3, now let's consider a directed graph $\mathcal{G}_{D_1|D_2} = (\mathcal{V}_{D_1|D_2}, \mathcal{E}_{D_1|D_2})$, where $\mathcal{V}_{D_1|D_2} \subseteq \mathcal{V}_P$ and $\mathcal{E}_{D_1|D_2}$ is a multi-subset of $\mathcal{E}_P$ (i.e., an edge in $\mathcal{E}_P$ may appear multiple times in $\mathcal{E}_{D_1|D_2}$). $\mathcal{E}_{D_1|D_2}$ is constructed as follows: for each $(s^i_x, s^i_y) \in T(D_1, D_2)$, i) if (x, y) lifts $\mathsf{q}_{\phi'}$ and lowers q_ϕ, add a directed edge $(\mathsf{q}_\phi, \mathsf{q}_{\phi'})$ into $\mathcal{E}_{D_1|D_2}$; ii) if (x, y) lifts q_ϕ but not lowering any other $\mathsf{q}_{\phi'}$, add an edge (v^+, q_ϕ); and iii) if (x, y) lowers q_ϕ but not lifting any other $\mathsf{q}_{\phi'}$, add an edge (q_ϕ, v^-). $\mathcal{V}_{D_1|D_2}$ is the set of count queries involved in $\mathcal{E}_{D_1|D_2}$.

$\mathcal{G}_{D_1|D_2}$ is Eulerian, i.e., each vertex has the same in-degree as out-degree except v^+ and v^- (if existing in $\mathcal{G}_{D_1|D_2}$), because of the above construction and the fact that $D_1, D_2 \in \mathcal{I}_Q$. As $|\mathcal{E}_{D_1|D_2}| = |T(D_1, D_2)| > \max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$ (i.e., $\mathcal{G}_{D_1|D_2}$ is larger than any simple cycle or simple v^+-v^- path in $\mathcal{G}_P$) and $\mathcal{G}_{D_1|D_2}$ is Eulerian, $\mathcal{G}_{D_1|D_2}$ must have a proper subgraph which is either a simple cycle or a simple v^+-v^- path. Let $\mathcal{E}_{D_1\to D_2}$ be the edge set of this simple cycle/path. Construct D_3 that is identical to D_1, except that for each secret pair (s^i_x, s^i_y) associated with each edge in $\mathcal{E}_{D_1\to D_2}$, the value of tuple t with $t.\_id = i$ is changed from x to y. We can show that D_3 satisfies its definition, and thus the proof for Direction I is completed.

Direction II ($S(h, P) \geq 2\max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$). Let's first prove a weaker inequality:

$$\max_{(D_1,D_2)\in N(P)} |T(D_1, D_2)| \geq \max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}. \qquad (17)$$

It implies $S(h, P) \geq 2\max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$ if the condition in (ii) of the theorem holds. Combined with Direction I, we can conclude $S(h, P) = 2\max\{\alpha(\mathcal{G}_P), \xi(\mathcal{G}_P)\}$.

To prove (17), it suffices to show that for any simple cycle/v^+-v^- path in $\mathcal{G}_P$, we can construct two databases D_1 and D_2 s.t. $(D_1, D_2) \in N(P)$ and $|T(D_1, D_2)| =$ its length. Consider a simple cycle $\mathsf{q}_{\phi_1}, \mathsf{q}_{\phi_2}, \ldots, \mathsf{q}_{\phi_l}, \mathsf{q}_{\phi_{l+1}} = \mathsf{q}_{\phi_1}$. Starting with any database $D \in \mathcal{I}_Q$, let $D_1 \leftarrow D$ and $D_2 \leftarrow D$ initially. For each edge $(\mathsf{q}_{\phi_i}, \mathsf{q}_{\phi_{i+1}})$, from the definition of policy graphs, we can find a secret pair $(x, y) \in E(G)$ s.t. $(\neg\mathsf{q}_{\phi_i}(x) \wedge \mathsf{q}_{\phi_i}(y)) \wedge (\mathsf{q}_{\phi_{i+1}}(x) \wedge \neg\mathsf{q}_{\phi_{i+1}}(y))$; create two new tuples: $t_1.\_id = t_2.\_id = i$, $t_1 = x$, and $t_2 = y$; and then let $D_1 \leftarrow D_1 \cup \{t_1\}$ and $D_2 \leftarrow D_2 \cup \{t_2\}$. It is not hard to verify that finally we get two databases D_1 and D_2 s.t. $(D_1, D_2) \in N(P)$ and $|T(D_1, D_2)| =$ cycle length. The proof is similar for a simple v^+-v^- path. □

B. PROOF OF THEOREM 4.1

Proof. (sketch) Let M_{M_1,M_2} denote the mechanism that outputting the results of M_1 and M_2 sequentially. As M_1 satisfies (ϵ_1, P)-Blowfish privacy, for every pair of neighboring databases $(D_a, D_b) \in N(P)$, and every result $r_1 \in range(M_1)$, we have

$$Pr[M_1(D_a) = r_1] \leq e^{\epsilon_1} Pr[M_1(D_b) = r_1] \qquad (18)$$

The result of M_1 is outputted before the result of M_2, so r_1 will turn out to be another input of M_2, together with the original dataset. As M_1 satisfies (ϵ_1, P)-Blowfish privacy, for every pair of neighboring databases $(D_a, D_b) \in N(P)$ coupling with the same r_1, and for every result $r_2 \in range(M_2)$, we have

$$Pr[M_2(D_a, r_1) = r_2] \leq e^{\epsilon_1} Pr[M_2(D_b, r_1) = r_2] \qquad (19)$$

Therefore, for every pair of neighboring databases $(D_a, D_b) \in N(P)$, and every set of output sequence (r_1, r_2), we have

$$\begin{aligned} & Pr[M_{M_1,M_2}(D_a) = (r_1, r_2)] \\ = \; & Pr[M_1(D_a) = r_1] Pr[M_2(D_a, r_1) = r_2] \\ \leq \; & e^{\epsilon_1} Pr[M_1(D_b) = r_1] e^{\epsilon_2} Pr[M_2(D_b, r_1) = r_2] \\ \leq \; & e^{\epsilon_1+\epsilon_2} Pr[M_1(D_b) = r_1] Pr[M_2(D_b, r_1) = r_2] \\ = \; & e^{\epsilon_1+\epsilon_2} Pr[M_{M_1,M_2}(D_b) = (r_1, r_2)] \qquad (20) \end{aligned}$$

□

C. PROOF OF THEOREM 4.2-4.3

Proof. (sketch) For every pair of neighboring databases $(D_a, D_b) \in N(P)$ with the cardinality constraint or with disjoint subsets of constraints $Q_1, ..., Q_p$, there is only one subset of _ids, let's say S_i*, with different values in D_a and D_b while $D_a \cap S_i = D_b \cap S_i$ for all $i \neq i^*$. Hence, for every set of output sequence r,

$$\begin{aligned} & Pr[M(D_a) = r] = \prod_i Pr[M_i(D_a \cap S_i) = r_i] \\ \leq \; & e^{\epsilon_{i^*}} Pr[M_{i^*}(D_b \cap S_{i^*}) = r_{i^*}] \prod_{i, i\neq i^*} Pr[M_i(D_b \cap S_i) = r_i] \\ \leq \; & e^{\max_i \epsilon_i} \prod_i Pr[M_i(D_b \cap S_i) = r_i] \\ = \; & e^{\max_i \epsilon_i} Pr[M(D_b) = r] \qquad (21) \end{aligned}$$

□