Design of Policy-Aware Differentially Private Algorithms

Samuel Haney
Duke University
Durham, NC, USA
shaney@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

The problem of designing error optimal differentially private algorithms is well studied. Recent work applying differential privacy to real world settings have used variants of differential privacy that appropriately modify the notion of neighboring databases. The problem of designing error optimal algorithms for such variants of differential privacy is open. In this paper, we show a novel transformational equivalence result that can turn the problem of query answering under differential privacy with a modified notion of neighbors to one of query answering under standard differential privacy, for a large class of neighbor definitions.

We utilize the Blowfish privacy framework that generalizes differential privacy. Blowfish uses a *policy graph* to instantiate different notions of neighboring databases. We show that the error incurred when answering a workload $\mathbf{W}$ on a database $\mathbf{x}$ under a Blowfish policy graph G is identical to the error required to answer a transformed workload $f_G(\mathbf{W})$ on database $g_G(\mathbf{x})$ under standard differential privacy, where f_G and g_G are linear transformations based on G. Using this result, we develop error efficient algorithms for releasing histograms and multidimensional range queries under different Blowfish policies. We believe the tools we develop will be useful for finding mechanisms to answer many other classes of queries with low error under other policy graphs.

1. INTRODUCTION

The problem of private release of statistics from databases has become very important with the increasing use of databases with sensitive information about individuals in government and commercial organizations. ϵ-Differential privacy [3] has become the standard for private release of statistics due to its strong guarantee that "similar" inputs must yield "similar" outputs. Two input databases are similar if they are *neighbors*, meaning that they differ in the presence or absence of a single record. Output similarity is quantified by ϵ, which bounds the log-odds of generating the same output from any pair of neighbors. Thus, if a record corresponds to all the data from one individual, differential privacy ensures that a single individual does not influence the inferences that can be drawn from the released statistics. Small ϵ results in greater privacy but also lesser utility. Thus, ϵ can be used to trade-off privacy for utility.

However, in certain applications (e.g., [17]), the differential privacy guarantee is too strict to produce private release of data that has any non-trivial utility. Tuning the parameter ϵ is not helpful here: enlarging ϵ degrades the privacy guaranteed without a commensurate improvement in utility. Hence, recent work has considered relaxing differential privacy by modifying the notion of neighboring inputs by defining some metric over the space of all databases. Application designers can use this in addition to ϵ to better tradeoff privacy for utility. This idea has been applied to graphs (edge- versus vertex-differential privacy [17]), streams (event- instead of individual-privacy [6]), location privacy (geo-indistinguishability [1]) and to study fairness in targeted advertising ([4]), and has been formalized by multiple proposed frameworks ([2, 11, 12]). While these relaxations permit algorithms with significantly better utility than the standard notion of differential privacy, such algorithms must be designed from scratch. It is unknown how to derive algorithms that optimally leverage the relaxed privacy guarantee provided by the modified notion of neighbors to result in the least loss of utility. For instance, there are no known algorithms for releasing histograms or answering range queries with high utility under geo-indistinguishability. Moreover, there is no known method to utilize the literature on differentially private algorithms for this purpose.

In this paper we present a novel and theoretically sound methodology for designing algorithms for relaxed privacy notions using algorithms that satisfy differential privacy, thus bridging the algorithm design problem under different privacy notions. Our results apply to the Blowfish privacy framework [11], which generalizes differential privacy by allowing for different notions of neighboring databases (or *privacy policies*). We use this methodology to derive novel algorithms that satisfy Blowfish under relaxed privacy policies for releasing histograms and multi-dimensional range queries that have significantly better utility than the best known differentially private algorithms for these tasks. In the rest of this section, we present an overview of our results, describe the outline of the paper and then discuss related work.

Overview of Our Results. We first informally introduce the Blowfish privacy framework to help understand our theoretical and algorithmic results. The Blowfish framework instantiates a large class of "similarity" or neighbor defi-

Proceedings of the VLDB Endowment, Vol. 9, No. 4

nitions, using a "policy graph" defined over the domain of database records. Two input databases are neighbors if they differ in one record, and the differing values form an edge (u, v) in the policy graph. Thus, one can not infer whether an individual's value was u or v based on the released output. For example, consider a *grid policy graph* (we will study its general form later in this paper): uniformly divide a 2D map into $k \times k$ grid cells, and each database record is one of the k^2 grid cells; only "nearby" points, e.g., pairs within Manhattan distance θ, are connected by edges in the policy graph. Such a policy when used for location data implies that it is acceptable to reveal the rough location of an individual (e.g., the city), as two points belonging to two different cities are far away so no edge in the policy graph connects them; however, it requires that fine-grained location information (e.g., whether the individual is at home or at a nearby cafe) be hidden, when two grid points are close enough. This special instance of Blowfish framework is similar to a recently proposed notion called geo-indistinguishability [1].

Our main result is called *transformational equivalence*, and we aim to show that a mechanism $\mathcal{M}$ for answering a set of linear queries $\mathbf{W}$ on a database $\mathbf{x}$ satisfies (ϵ, G)-Blowfish privacy (i.e., differential privacy where neighboring databases are constructed with respect to the policy graph G) if and only if $\mathcal{M}$ is a mechanism for answering a transformed set of queries $f_G(\mathbf{W})$ on a transformed database $g_G(\mathbf{x})$ that satisfies ϵ-differential privacy. Here, f_G and g_G are linear transformations. However, we can not hope to prove such an equivalence in general. We prove (Theorem 4.4) that such an equivalence result implies that there exist a method to embed distances on any graph G to distances in the L_1 metric without any distortion. This is because the distance between two input datasets induced by the neighborhood relation for differential privacy is the L_1 metric, and the distance under Blowfish is related to distances on graph G. Such a distortion free embedding is not known for large classes of graphs (e.g., a cycle) [16].

Nevertheless, we are able to show this equivalence result for a large class of mechanisms and for a large class of graphs. First, we show that the transformational equivalence result holds for all algorithms that are instantiations of the matrix mechanism framework [14]. Matrix mechanisms algorithms, like Laplace mechanism for releasing histograms, and hierarchical mechanism [10] and Privelet [18] for answering range queries, are popular building blocks for differentially private algorithm design. Transformational equivalence holds for such *data independent* mechanisms since the noise introduced by such mechanisms is independent of the input database. (The negative result uses a data-dependent mechanism whose error depends on the input database).

Next, we are also able to show that when G is a tree there exist linear transformations f_G and g_G such that any mechanism M for answering $\mathbf{W}$ on database $\mathbf{x}$ satisfies (ϵ, G)-Blowfish privacy if and only if M is an ϵ-differentially private mechanism for answering $f_G(\mathbf{W})$ on database $g_G(\mathbf{x})$. The result follows (though not immediately) from the fact that trees permit a distortion free embedding into the L_1 metric [7, 16]. This result holds for all privacy mechanisms, including data-dependent mechanisms.

Finally, while the equivalence result does not hold for general mechanisms and general policy graphs, we can achieve an approximate equivalence. More specifically, we show the following *subgraph approximation* result: if G and G' are such that every edge (u, v) in G is connected by a path of length at most ℓ in G', then a mechanism M that ensures $(\ell \cdot \epsilon, G')$-Blowfish privacy also ensures (ϵ, G)-Blowfish privacy. Thus, if for a graph G there is a tree T such that distances in G are not distorted by more than a multiplicative factor of ℓ in the tree T, then there exist transformations f_T and g_T such that an ϵ-differentially private mechanism M for answering $f_T(\mathbf{W})$ on database $g_T(\mathbf{x})$, is also an $(\ell \cdot \epsilon, G)$-Blowfish private mechanism for answering $\mathbf{W}$ on $\mathbf{x}$.

Additionally, a direct consequence of transformational equivalence is it allows us to derive error lower bounds and general approximation algorithms for Blowfish private mechanisms by extending work on error lower bounds for differentially private mechanisms [9, 15]. We refer the reader to the full paper [8] for these results.

We apply the transformational equivalence theorems to derive novel (near) optimal algorithms for answering multidimensional range query and histogram workloads under reasonable Blowfish policy graphs G (like the grid graph). We reduce the problem of designing a Blowfish algorithm to that of finding a differentially private mechanism for a new workload $\mathbf{W}_G = f_G(\mathbf{W})$ and a database $\mathbf{x}_g = g_G(\mathbf{x})$. We design matrix mechanism algorithms for all the policy graphs, and data dependent techniques when G is a tree or can be approximated by a tree. For the policy graphs we consider, we show a polylogarithmic (in the domain size) improvement in error compared to the best data oblivious differentially private mechanism. We also present empirical results for the tasks of answering 1- and 2-dimensional range queries to show that our data dependent algorithms outperform their differentially private counterparts.

Organization. The rest of this section is a brief survey of related work. Section 2 presents notations and definitions that we will use throughout the paper. Section 3 introduces and motivates the Blowfish privacy framework. We describe our main result, transformational equivalence in Section 4. Section 5 presents novel mechanisms for answering multidimensional range queries and histogram queries under various instantiations of the Blowfish framework, and presents the subgraph approximation lemma. In Section 6, we perform experiments comparing the performance of our Blowfish private mechanisms to differentially private mechanisms, and explore data-dependent mechanisms.

Related Work. As mentioned earlier, works ([1],[4]) have developed relaxations of differential privacy that have specific applications (e.g. location privacy). Other work has focused on developing flexible privacy definitions that generalize all these application specific notions. The Pufferfish framework [12] generalizes differential privacy by specifying what information should be kept secret, and the adversary's prior knowledge. He et al. [11] propose the Blowfish framework which also generalizes differential privacy and is inspired by Pufferfish. [2] investigates notions of privacy that can be defined as metrics over the set of databases. All these frameworks allow finer grained control on what information about individuals is kept secret, and what prior knowledge an adversary might possess, and thus allow customizing privacy definitions to the requirements of different applications.

As far as we aware, all previous work on relaxed privacy definitions have developed mechanisms directly for their applications. We don't know of any work which shows how to map these relaxed privacy definitions to instances of differential privacy.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Figure 1: $\mathbf{I}_k$ (left) and $\mathbf{C}_k$ (right) workloads.

2. PRELIMINARIES

Databases and Query Workloads. Let $\mathcal{T} = \{v_1, v_2, \ldots, v_k\}$ be a domain of values with domain size $|\mathcal{T}| = k$. A database D is a set of entries whose values come from $\mathcal{T}$. Let $\mathcal{I}_n$ be the set of all databases D over $\mathcal{T}$ such that the number of entries in D is n, i.e., $|D| = n$. And let $\mathcal{I}$ be the set of all databases with any number of entries. We represent a database D as a vector $\mathbf{x} \in \mathbb{R}^k$ with $\mathbf{x}[i]$ denoting the true count of entries in D with the i^{th} value of the domain $\mathcal{T}$. That is, the database is represented as a histogram over the domain. A *linear query* $\mathbf{q}$ is a k-dimensional row vector of real numbers with answer $\mathbf{q} \cdot \mathbf{x}$. If the entries of $\mathbf{q}$ are restricted to 1's and 0's, we sometimes call $\mathbf{q}$ a linear counting query, since it counts the number of entries in D with a particular subset of values in the domain. A *workload* is a set of q linear queries. So a workload can be represented as a $q \times k$ matrix $\mathbf{W} = [\mathbf{q}_1 \mathbf{q}_2 \ldots \mathbf{q}_q]^\top \in \mathbb{R}^{q \times k}$, where each vector $\mathbf{q}_i \in \mathbb{R}^k$ corresponds to a linear query. The answer to the workload $\mathbf{W}$ is $\mathbf{W} \cdot \mathbf{x}$, whose entries will be answers to the individual linear queries.

EXAMPLE 2.1. *Figure 1 shows examples of two well studied workloads.* $\mathbf{I}_k$ *is the identity matrix representing the histogram query on* $\mathcal{T}$ *reporting* $[\mathbf{x}[1]\mathbf{x}[2] \ldots \mathbf{x}[k]]^\top$. $\mathbf{C}_k$ *corresponds to the* cumulative histogram *workload, where each query corresponds to the* prefix sum $\sum_{j=1}^{i} \mathbf{x}[j]$.

Differential privacy is based on the concept of *neighbors*. Two databases are neighbors if they differ in one entry.

DEFINITION 2.1 (NEIGHBORS [5]). *Any two databases* D *and* D' *are neighbors iff they differ in the presence of a single entry. That is,* $\exists v \in \mathcal{T}$, $D = D' \cup \{v\}$ *or* $D' = D \cup \{v\}$.

An algorithm satisfies differential privacy if its outputs on any two neighboring databases are indistinguishable.

DEFINITION 2.2 (ϵ-DIFFERENTIAL PRIVACY [5]). *A randomized algorithm (mechanism)* $\mathcal{M}$ *satisfies* ϵ*-differential privacy if for any subset of outputs* $S \subseteq range(\mathcal{M})$*, and for any pair of neighboring databases* D *and* D'*,*

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(D') \in S].$$

A slightly different definition of neighbors yields a common variant of differential privacy: one database can be obtained from its neighbor by replacing one entry x with a different value $y \in \mathcal{T}$. The resulting privacy notation is called ϵ*-indistinguishability* or *bounded* ϵ*-differential privacy*. Unless otherwise specified, we use the term differential privacy to mean the original *unbounded* version (Definitions 2.1-2.2).

Sensitivity and Private Mechanisms. Suppose we use a differentially private algorithm $\mathcal{M}$ to publish the result of a workload $\mathbf{W}$ on a database D that is represented as a vector $\mathbf{x}$. $\mathcal{M}$ is called *data independent* if the amount of noise $\mathcal{M}$ adds does not depend on the database $\mathbf{x}$, and *data dependent* otherwise. In both cases, the amount of noise depends the *sensitivity* of a workload. $||\cdot||_1$ denotes the L_1 norm.

DEFINITION 2.3 (SENSITIVITY [5, 15]). *Let* $\mathcal{N}$ *denote the set of pairs of neighbors. The* L_1 *sensitivity of* $\mathbf{W}$ *is:*

$$\Delta_{\mathbf{W}} = \max_{(\mathbf{x},\mathbf{x}') \in \mathcal{N}} \|\mathbf{W}\mathbf{x} - \mathbf{W}\mathbf{x}'\|_1.$$

EXAMPLE 2.2. *The* L_1 *sensitivities of* $\mathbf{I}_k$ *and* $\mathbf{C}_k$ *are* 1 *and* k*, resp.*

A well-studied class of differentially private algorithms is called *Laplace mechanism* [5]. Let $\text{Lap}(\sigma)^m$ be a m-dimensional vector of independent samples, where each sample is drawn from $\eta \propto \exp(-\frac{|x|}{\sigma})$.

Measuring Errors. We use *mean squared error* to measure the amount of noise injected in private algorithms.

DEFINITION 2.4 (ERROR). *Let* $\mathbf{W} = [\mathbf{q}_1 \mathbf{q}_2 \ldots \mathbf{q}_q]^\top$ *be a workload of linear queries, and* $\mathcal{M}$ *be a mechanism to publish the query result privately. Let* $\mathbf{x}$ *be the vector representing the database. The mean squared error of answering a workload* $\mathbf{W}$ *on the database* $\mathbf{x}$ *using* $\mathcal{M}$ *is*

$$\text{ERROR}_{\mathcal{M}}(\mathbf{W}, \mathbf{x}) = \sum_{i=1}^{q} \mathbb{E}\left[(\mathbf{q_i}\mathbf{x} - \mathcal{M}(\mathbf{q_i}, \mathbf{x}))^2\right]$$

where $\mathcal{M}(\mathbf{q}, \mathbf{x})$ *is the noisy answer of query* $\mathbf{q}$*. We define the data-independent error of a mechanism* $\mathcal{M}$ *to be*

$$\text{ERROR}_{\mathcal{M}}(\mathbf{W}) = \max_{\mathbf{x}} \{\text{ERROR}_{\mathcal{M}}(\mathbf{W}, \mathbf{x})\}.$$

Laplace mechanism is known to provide ϵ-differential privacy with mean squared error as a function of L_1 sensitivity.

THEOREM 2.1 ([5]). *Let* $\mathbf{W}$ *be a* $q \times k$ *workload. The Laplace mechanism* $\mathcal{L}(\mathbf{W}, \mathbf{x}) = \mathbf{W}\mathbf{x} + \text{Lap}(\sigma)^q$ *satisfies* ϵ*-differential privacy, with* $\text{ERROR}_{\mathcal{L}}(\mathbf{W}) = 2q\Delta_{\mathbf{W}}^2/\epsilon^2$.

3. BLOWFISH PRIVACY

The *Blowfish privacy* framework, originally introduced by He et al. [11], is a class of privacy notations that generalize neighboring databases in differential privacy. It allows privacy policy to focus only on neighbors that users are sensitive about. The major building block of an instantiation of Blowfish is called *policy graph*. A policy graph encodes users' private and sensitive information by specifying which pairs of domain values in $\mathcal{T}$ should *not* be distinguished between by an adversary. By carefully choosing a policy graph (or equivalently, restricting the set of neighboring databases), Blowfish trades-off privacy for potential gains in utility.

DEFINITION 3.1 (POLICY GRAPH). *A policy graph is a graph* $G = (V, E)$ *with* $V \subseteq \mathcal{T} \cup \{\perp\}$*, where* $\perp$ *is the name of a special vertex, and* $E \subseteq (\mathcal{T} \cup \{\perp\}) \times (\mathcal{T} \cup \{\perp\})$.

The above definition of policy graph is slightly different from the one in [11] with an additional special vertex $\perp$ to generalize both unbounded and bounded versions of differential privacy. Intuitively, an edge $(u, v) \in E$ defines a pair of domain values that an adversary should not be able to distinguish between. $\perp$ is a dummy value not in $\mathcal{T}$, and an edge $(u, \perp) \in E$ means that an adversary should not be able to distinguish between the presence of a tuple with value u or the absence of the tuple from the database. For technical reasons, if there is some edge incident on $\perp$, we add a

zero column vector $\mathbf{0}$ into the workload $\mathbf{W}$ to correspond to the dummy value $\perp$, as well as a zero entry in the database vector $\mathbf{x}$ correspondingly. So it is ensured that every node in V is associated with a column in $\mathbf{W}$ and an entry in $\mathbf{x}$.

We next revisit the Blowfish privacy framework.

DEFINITION 3.2 (BLOWFISH NEIGHBORS). *We consider a policy graph $G = (V, E)$. Let D and D' be two databases. D and D' are neighbors, denoted $(D, D') \in \mathcal{N}(G)$, iff exactly one of the following is true:*

- *D and D' differ in the value of exactly one entry such that $(u, v) \in E$, where u is the value of the entry in D and v is the value of the entry in D';*
- *D differs from D' in the presence or absence of exactly one entry, with value u, such that $(u, \perp) \in E$.*

DEFINITION 3.3 ((ϵ, G)-BLOWFISH PRIVACY). *Let G be a policy graph. A mechanism $\mathcal{M}$ satisfies (ϵ, G)-Blowfish privacy if for any subset of outputs $S \subseteq range(\mathcal{M})$, and for any pair of neighboring databases $(D, D') \in \mathcal{N}(G)$,*

$$\Pr[\mathcal{M}(D) \in S] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(D') \in S].$$

Policy graph and Privacy guarantee. Policy graphs are used to define how privacy will be guaranteed to users, independent of an adversary's knowledge. For example, when the users require the strongest privacy guarantee, the following two policy graphs can be used,

$$G = (V, E) \text{ such that } E = \{(u, \perp) \,|\, \forall u \in \mathcal{T}\}, \text{ and}$$

$$G = (V, E) \text{ such that } E = \{(u, v) \,|\, \forall u, v \in \mathcal{T}\},$$

which corresponds to the unbounded and bounded versions of differential privacy, respectively. More generally, if a policy graph does not include $\perp$, we are essentially focusing on databases from $\mathcal{I}_n$, i.e., databases with fixed known size.

When the privacy guarantee is relaxed, we can adjust the policy graphs so that our algorithms in Section 5 will have higher utility. Following are two examples of designing such policy graphs for real-life scenarios.

(*Line Graph*) Consider a totally ordered domain $\mathcal{T} = \{a_1, a_2, \ldots, a_k\}$, where $\forall i : \; a_i < a_{i+1}$. One such example is a database of binned salaries of individuals, where a_i corresponds to a salary between 2^{i-1} and 2^i. When only revealing rough ranges of salaries is fine, it is OK for an adversary to distinguish between values that are far apart (e.g., a_1 vs a_k) but not distinguish between values that are closer to each other. So we can use a line graph as our policy graph to express this guarantee, where only adjacent domain values a_i and a_{i+1} are connected by an edge.

(*Grid Graph*) In the scenario of location data, when revealing rough location information is fine but more precise location information is private, we can use a grid policy graph (also discussed in Section 1). Here, a 2D map uniformly divided into $k \times k$ grid points as nodes in the graph (i.e., $\mathcal{T} = \{1, \ldots, k\} \times \{1, \ldots, k\}$). Only "nearby" points are connected by edges: $E = \{(u, v) | d(u, v) \leq \theta\}$ where $d(u, v)$ denotes the distance between two points $u, v \in \mathcal{T}$ (e.g., Manhattan distance) and θ is a policy-specific parameter.

Metric on databases. In general, a policy graph introduces a metric over databases, which quantifies the privacy guarantee provided by Blowfish. Consider two databases that differ in one tuple: $D_1 = D \cup \{u\}$ and $D_2 = D \cup \{v\}$, define the distance between D_1 and D_2 be $\text{dist}_G(u, v)$, i.e., the length of the shortest path between u and v in G. For mechanism $\mathcal{M}$ satisfying (ϵ, G)-Blowfish privacy (Defs 3.2-3.3), we have,

$$\Pr[\mathcal{M}(D_1) \in S] \leq e^{\epsilon \cdot \text{dist}_G(u,v)} \cdot \Pr[\mathcal{M}(D_2) \in S]. \quad (1)$$

For two databases differing in more than one tuple, we can repeatedly apply (1) for each differing tuple.

For the grid graph policy, changing the location of a tuple from u to v results in the output probabilities of $\mathcal{M}$ differing by a factor of e^{ϵ} if $d(u, v) \leq \theta$, and differing be a factor of $e^{\epsilon \cdot \lceil d(u,v)/\theta \rceil}$ in general. So finer grained location information gets stronger protection. This privacy guarantee is identical to a recent notion called geo-indistinguishability [1].

In this paper, we assume Blowfish policy graphs are connected. We discuss Blowfish policies with disconnected components in the full version.

4. TRANSFORMATIONAL EQUIVALENCE

We now present our main result, called *transformational equivalence*, which we will use to design Blowfish private algorithms later in the paper. This result establishes a mechanism-preserving two way relationship between Blowfish privacy and differential privacy. In general, our transformation can be stated as follows: *For policy graph G, there exists a transformation of the workload and database, $(\mathbf{W}, \mathbf{x}) \to (\mathbf{W}_G, \mathbf{x}_G)$ such that $\mathbf{W}\mathbf{x} = \mathbf{W}_G\mathbf{x}_G$, and a mechanism $\mathcal{M}$ is an (ϵ, G)-Blowfish private mechanism for answering workload $\mathbf{W}$ on input $\mathbf{x}$ if and only if $\mathcal{M}$ is also an ϵ-differentially private mechanism for answering $\mathbf{W}_G$ on $\mathbf{x}_G$.* However, we can't hope to show this result in general. We prove that for certain mechanisms transformational equivalence holds only when distances on a graph can be *embedded* into points in L_1 with no distortion. It is well known [16] that not all graphs permit such embeddings.

Hence, we show different results for different restrictions on $\mathcal{M}$ and G. In Section 4.1, we show that under a class of mechanisms called the *matrix mechanism*, transformational equivalence holds for any policy graph. In Section 4.2, we show via a metric embedding-like argument that when G is a tree, transformational equivalence holds for any mechanism $\mathcal{M}$. In Section 4.3, we state the negative result for general graphs and mechanisms, and present an approximate transformational equivalence that uses spanning trees of G albeit with some loss in the utility. All of our results rely on the existence of a transformation matrix $\mathbf{P}_G$ with certain properties, whose construction is discussed in Section 4.4

4.1 Equivalence for Matrix Mechanism

Li et al [14] describe the *matrix mechanism* framework for optimally answering a workload of linear queries. The key insight is that while some workloads $\mathbf{W}$ have a high sensitivity, they can be answered with low error by answering a different *strategy* query workload $\mathbf{A}$ such that (a) $\mathbf{A}$ has a low sensitivity $\Delta_{\mathbf{A}}$, and (b) rows in $\mathbf{W}$ can be reconstructed using a small number of rows in $\mathbf{A}$.

In particular, let $\mathbf{A}$ be a $p \times k$ matrix, and $\mathbf{A}^+$ denote its Moore-Penrose pseudoinverse, such that $\mathbf{W}\mathbf{A}\mathbf{A}^+ = \mathbf{W}$. The matrix mechanism is given by the following:

$$\mathcal{M}_{\mathbf{A}}(\mathbf{W}, \mathbf{x}) = \mathbf{W}\mathbf{x} + \mathbf{W}\mathbf{A}^+ Lap(\Delta_{\mathbf{A}}/\epsilon)^p \quad (2)$$

where, $Lap(\lambda)^p$ denotes p independent random variables drawn from the Laplace distribution with scale λ. Recall

that $\Delta_{\mathbf{A}}$ is the sensitivity of workload $\mathbf{A}$. It is easy to see that all matrix mechanism algorithms are data independent (i.e., the noise is independent of the input dataset).

In order to extend matrix mechanisms to Blowfish, we define the Blowfish specific sensitivity of a workload, $\Delta_{\mathbf{W}}(G)$ analogously to Definition 2.3:

DEFINITION 4.1. *The L_1 policy specific sensitivity of a query matrix $\mathbf{W}$ with respect to policy graph G is*

$$\Delta_{,\mathbf{W}}(G) = \max_{(\mathbf{x},\mathbf{x}')\in N(G)} \|\mathbf{W}\mathbf{x} - \mathbf{W}\mathbf{x}'\|_1$$

Let P_G be a matrix that satisfies the following properties. We will describe its construction in Section 4.4.

- $\mathbf{P}_G$ has $|V|-1$ rows and $|E|$ columns.
- Let $\mathbf{W}_G = \mathbf{W}\mathbf{P}_G$. Then $\Delta_{\mathbf{W}}(G) = \Delta_{\mathbf{W}_G}$. I.e., the sensitivity of workload $\mathbf{W}$ under Blowfish policy G is the same as the sensitivity of $\mathbf{W}_G$ under differential privacy.
- $\mathbf{P}_G$ has full row rank (and therefore a right inverse $\mathbf{P}_G^{-1}$). For vector $\mathbf{x}$ we let $\mathbf{x}_G$ denote $\mathbf{P}_G^{-1}\mathbf{x}$.

Given such a $\mathbf{P}_G$, we can show our first transformational equivalence result.

THEOREM 4.1. *Let G be a Blowfish policy graph and $\mathbf{W}$ be a workload. Suppose $\mathbf{P}_G$ exists with the properties given above. Then the matrix mechanism given by Equation 2 is both a (ϵ, G)-Blowfish private mechanism for answering $\mathbf{W}$ on $\mathbf{x}$ and an ϵ-differentially private algorithm for answering $\mathbf{W}_G$ on $\mathbf{x}_G$. Since $\mathbf{W}\mathbf{x} = \mathbf{W}_G\mathbf{x}_G$, the mechanism has the same error in both instances.*

PROOF. We show that

$$\mathbf{W}\mathbf{x}+\mathbf{W}\mathbf{A}^+ Lap(\frac{\Delta_{\mathbf{A}}(G)}{\epsilon})^p = \mathbf{W}_G\mathbf{x}_G+\mathbf{W}_G\mathbf{A}_G^+ Lap(\frac{\Delta_{\mathbf{A}_G}}{\epsilon})^p.$$

First, $\mathbf{W}\mathbf{P}_G\mathbf{P}_G^{-1}\mathbf{x} = \mathbf{W}\mathbf{I}_k\mathbf{x} = \mathbf{W}\mathbf{x}$. Next, by assumption we have that $\Delta_{\mathbf{A}}(G) = \Delta_{\mathbf{A}_G}$. Finally,

$$\begin{aligned}\mathbf{W}_G\mathbf{A}_G^+ = \mathbf{W}\mathbf{P}_G(\mathbf{A}\mathbf{P}_G)^+ &= \mathbf{W}\mathbf{P}_G\mathbf{P}_G^+\mathbf{A}^+ \\ &= \mathbf{W}\mathbf{A}^+ \qquad (\mathbf{P}_G \text{ has full row rank})\end{aligned}$$ □

4.2 Equivalence when G is a Tree

When G is a tree, we can show something stronger, that transformational equivalence holds for any mechanism $\mathcal{M}$. More formally, suppose $\mathbf{P}_G$ has the following property.

CLAIM 4.2. *If G is a tree, any pair of $\mathbf{y}, \mathbf{z} \in \mathbb{R}^k$ are neighbors according to the Blowfish policy G if and only if $\mathbf{P}_G^{-1}\mathbf{y}$ and $\mathbf{P}_G^{-1}\mathbf{z}$ are neighbors according to unbounded differential privacy (which are vectors with L_1 distance of 1).*

We construct a $\mathbf{P}_G$ satisfying Claim 4.2 in Section 4.4. Our stronger transformational equivalence result follows.

THEOREM 4.3. *Let $\mathbf{x} \in \mathbb{R}^k$ represent a database, $\mathbf{W}$ be a workload with q linear queries, and $G = (V, E)$ be a Blowfish policy graph and a tree. We can find an invertible mapping given by $f(\mathbf{x}, \mathbf{W}, G) = (\mathbf{P}_G^{-1}\mathbf{x}, \mathbf{W}\mathbf{P}_G)$, where $\mathbf{P}_G$ is a matrix depending on G, such that $\mathcal{M}$ is a (G, ϵ)-Blowfish private mechanism for answering $(\mathbf{W}, \mathbf{x})$ with error α if and only if $\mathcal{M}$ is an ϵ-differentially private mechanism for answering $(\mathbf{W}\mathbf{P}_G, \mathbf{P}_G^{-1}\mathbf{x})$ with error α.*

PROOF. Suppose $\mathbf{P}_G$ satisfies the properties given at the beginning of the section. Then, mechanism $\mathcal{M}$ will have the same error on both instances, since the true answers to the workloads are the same in both cases:

$$\mathbf{W}\mathbf{P}_G\mathbf{P}_G^{-1}\mathbf{x} = \mathbf{W}\mathbf{I}_k\mathbf{x} = \mathbf{W}\mathbf{x}.$$

Additionally, the mapping is invertible, since $\mathbf{W}\mathbf{P}_G\mathbf{P}_G^{-1} = \mathbf{W}$ and $\mathbf{P}_G\mathbf{P}_G^{-1}\mathbf{x} = \mathbf{x}$. $\mathcal{M}$ is both (ϵ, G)-differentially private on $\mathbf{W}, \mathbf{x}$ and ϵ-differentially private on $\mathbf{W}_G, \mathbf{x}_G$, since the mapping preserves neighbors. □

4.3 General Graphs and Mechanisms

We can show that we can not hope to prove transformation equivalence for general graphs and mechanisms. First we define an *embedding* of graphs.

DEFINITION 4.2. *Let $G = (V, E)$ be a graph. Let ρ be a deterministic mapping from vertices in V to real valued vectors. Let $d_G(u, v)$ denote the shortest distance between vertices u and v, and $d(\rho(u), \rho(v)) = ||\rho(u) - \rho(v)||_1$ the L_1 distance between the mapped vectors. We define the* stretch *of mapping ρ to be $\max_{u,v\in V} d(\rho(u), \rho(v))/d_G(u, v)$, or the maximum multiplicative increase in distances due to the mapping. Similarly the* shrink *of ρ is defined as $\min_{u,v\in V} d(\rho(u), \rho(v))/d_G(u, v)$, or the smallest multiplicative decrease in distances. We call ρ an isometric embedded if stretch and shrink equal to 1.*

We now show that for graphs with no isometric embedding into points in L_1, transformational equivalence does not hold. It is well known that such graphs exist. One example is the cycle on n vertices, for which no deterministic mapping is known with stretch less than $(n-1)$ [16].

THEOREM 4.4. *Let G be a graph that does not have an isometric embedding into points in L_1. There exists a mechanism $\mathcal{M}$ and workload $\mathbf{W}$ such that for any transformation of $(\mathbf{W}, \mathbf{x}) \to (\mathbf{W}_G, \mathbf{x}_G)$ such that $\mathbf{W}\mathbf{x} = \mathbf{W}_G\mathbf{x}_G$, either $\mathcal{M}$ is not an (ϵ, G)-Blowfish private mechanism for answering $\mathbf{W}$ on $\mathbf{x}$, or $\mathcal{M}$ is not a ϵ-differentially private mechanism for answering $\mathbf{W}_G$ on $\mathbf{x}_G$.*

We refer the reader to the full paper for all proofs in this section. We would like to note that transformational equivalence holds for policy graphs that are trees, since trees can be isometrically embedded into points in L_1, and the $\mathbf{P}_G$ we construct is one such mapping. Moreover, the proof for Theorem 4.4 requires a mechanism $\mathcal{M}$ that is data-dependent; it uses the exponential mechanism that introduces noise that depends on the input. We believe data dependence is necessary for the negative result, and hence we were able to show transformational equivalence for matrix mechanism algorithms (that are data independent).

Despite the negative result, we next show an approximate transformational equivalence for general graphs and mechanisms with some loss in utility. The error in our approximate transformation is proportion to the stretch resulting from embedding G into a spanning tree of G, G'. Transformational equivalence can then be applied on G' giving us an approximate equivalence under the original graph.

LEMMA 4.5. *(Subgraph Approximation) Let $G = (V, E)$ be a policy graph. Let $G' = (V, E')$ be a spanning tree of G on the same set of vertices, such that every $(u, v) \in E$ is connected in G' by a path of length at most ℓ (G' is said to*

be an ℓ-approximate subgraph[1]). Then for any mechanism $\mathcal{M}$ which satisfies (ϵ, G')-Blowfish privacy, $\mathcal{M}$ also satisfies $(\ell \cdot \epsilon, G)$-Blowfish privacy.

COROLLARY 4.6. *Let G be a graph and let G' be an ℓ-approximate spanning tree. Suppose $\mathcal{M}$ is an ϵ differentially private mechanism for $\mathbf{W}_{G'}, \mathbf{x}_{G'}$. Then, $\mathcal{M}$ is an $(\ell \cdot \epsilon, G)$-Blowfish private mechanism for $\mathbf{W}, \mathbf{x}$. Since $\mathbf{W}\mathbf{x} = \mathbf{W}_G\mathbf{x}_G$, the mechanism has the same error in both instances.*

A well-known result of Fakcharoenphol et al [7] (Theorem 2) shows that any metric can be embedded into a distribution of trees with $O(\log n)$ expected stretch. It would be desirable to use this result to give a $O(\log(n))$-approximate subgraph for any graph G. However, because the bound on stretch only holds in expectation, our privacy guarantee would only hold in expectation! A deterministic embedding with low stretch does not always exist. To see this, consider an n-vertex cycle. Any spanning tree consists of all but one edge (u, v) from the cycle. While u and v were distance 1 apart in the cycle, they are distance $n-1$ apart in the spanning tree! If we picked a spanning tree at random by randomly choosing the edge that was dropped the expected stretch is only 2. Using a union bound over all pairs in this example, we can also see that it is impossible to guarantee a low stretch (and therefore a privacy guarantee) with high probability. Therefore, we cannot apply Lemma 4.5 in a general way to find a suitable spanner for any policy graph G. However, the lemma is still useful in many cases and we will use it throughout the rest of the paper.

4.4 Construction of $\mathbf{P}_G$

Our construction of $\mathbf{P}_G$ from the policy graph G is related to the vertex-edge incidence matrix, where every row corresponds to a vertex in G, every column corresponds to an edge in G. A column has two non-zero entries (1 and -1) in the rows corresponding vertices connected by the corresponding edge. We can view $\mathbf{P}_G$ and $\mathbf{P}_G^{-1}$ as linear transformations from the vertex domain V to the edge domain E. While $\mathbf{x}$ corresponds to counts on vertices of G, the transformed database $\mathbf{x}_G = \mathbf{P}_G^{-1}\mathbf{x}$ would assign weights to edges in G. Similarly, while an original linear query $q \in \mathbf{W}$ associates weights on (a subset of) vertices in G, a query $q_G \in \mathbf{W}_G = \mathbf{W}\mathbf{P}_G$ associates weights on (a subset of) edges in G. This intuition will be very useful when using the equivalence result to design (ϵ, G)-Blowfish algorithms.

We cannot just use the vertex-edge incidence matrix as $\mathbf{P}_G$ since it may either have $k+1$ rows (when G contains $\perp$), or since it does not have an inverse (when G does not contain $\perp$). We will describe our construction of $\mathbf{P}_G$ that satisfies all our constraints in the rest of the section. The details are quite technical, and an uninterested reader can skip over them and still understand the rest of the paper. As mentioned before, we assume G is connected.

Case I: Unbounded (with $\perp$)

We start our construction with a simple case. Let $G = (V, E)$ be a *connected* undirected graph, with $V = \mathcal{T} \cup \{\perp\}$, $|\mathcal{T}| = k$. We define $\mathbf{P}_G$ to be the following $k \times |E|$-matrix: let each row of $\mathbf{P}_G$ correspond to a value in $\mathcal{T}$; for each edge $(u, v) \in E$ $(u, v \neq \perp)$, add a column to $\mathbf{P}_G$ with a 1 in the row corresponding to value u, a -1 in the row corresponding to value v (order of 1 and -1 is not important), and zeros in the rest of the rows; and for each edge $(u, \perp) \in E$ $(u \neq \perp)$, add a column with a 1 in the row corresponding to u and zeros in the rest. Figure 2 gives an example.

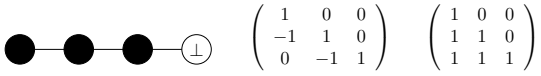


Figure 2: Example policy graph G and their $\mathbf{P}_G, \mathbf{P}_G^{-1}$

It is easy to see that $\mathbf{P}_G$ has all the properties required for our transformational equivalence results to hold.

LEMMA 4.7. *Let $\mathbf{W}$ be a workload, and G be a policy graph. Then $\Delta_{\mathbf{W}}(G) = \Delta_{\mathbf{W}_G}$.*

LEMMA 4.8. *$\mathbf{P}_G$ constructed above has rank k.*

As $\mathbf{P}_G$ has rank k, and $k \leq |E|$, it has a right inverse: $\mathbf{P}_G^{-1} = \mathbf{P}_G^\top(\mathbf{P}_G\mathbf{P}_G^\top)^{-1}$. Finally, we prove Claim 4.2: When G is a tree $\mathbf{P}_G$ isometrically maps neighbors under policy graph G to neighbors under differential privacy.

LEMMA 4.9. *Suppose $\mathbf{P}_G$ is constructed for a Blowfish policy graph G as above, and G is a tree. Any pair of databases $\mathbf{y}, \mathbf{z} \in \mathbb{R}^k$ are neighbors according to the Blowfish policy G if and only if $\mathbf{P}_G^{-1}\mathbf{y}$ and $\mathbf{P}_G^{-1}\mathbf{z}$ are neighboring databases according to unbounded differential privacy.*

We refer the reader to the full paper for all the proofs.

Case II: Bounded (without $\perp$)

We next consider a slightly more involved case: let $G = (V, E)$ be a *connected* undirected graph, with $V = \mathcal{T}$, where $|\mathcal{T}| = k$. If we follow the same construction as in Case I, rows in the resulting $\mathbf{P}_G$ are not linearly independent any more, and thus $\mathbf{P}_G^{-1}$ is not well-defined (no right inverse can be defined for $\mathbf{P}_G$). Fortunately, for every such G, we can replace one vertex in V with $\perp$, denoting the resulting graph as G', and correspondingly modify $\mathbf{W}$ and $\mathbf{x}$ to $\mathbf{W}', \mathbf{x}'$ resp., such that (a) $\mathbf{P}_{G'}$ is full rank, (b) answering $\mathbf{W}'$ on $\mathbf{x}'$ under policy G' has the same error as answering $\mathbf{W}$ on $\mathbf{x}$ under G, and (c) $\mathbf{W}\mathbf{x}$ can be reconstructed from the answer to $\mathbf{W}'\mathbf{x}'$.

Pick any value $v \in V$; in $G' = (V', E')$, let $V' = V - \{v\} + \{\perp\}$ and $E' = E - \{(v, u) \mid u \in V\} + \{(\perp, u) \mid (v, u) \in E\}$. Then G' falls into Case I, so we can construct $\mathbf{P}_{G'}$ and $\mathbf{P}_{G'}^{-1}$ as in Case I.

We transform $\mathbf{x}$ by removing the entry $\mathbf{x}[v]$ (denoted as $\mathbf{x}_{-v}$). We then transform $\mathbf{W}$ to $\mathbf{W}$ by removing the column v and rewriting all queries that depend on $\mathbf{x}[v]$ to use $n - \sum_{j \neq v} \mathbf{x}[j]$, where $n = \sum_{i \in \mathcal{T}} \mathbf{x}[i]$ is the size of the input database, without any loss in our ability to answer the original queries. We can do this because when $\perp$ is not in G, neighboring databases have the same number of tuples. We can show that our construction satisfies all three requirements (a), (b), and (c) discussed above.

LEMMA 4.10. *Consider G', $\mathbf{W}'$, and $\mathbf{x}_{-v}$ constructed above. We have: i) $\mathbf{W}\mathbf{x} = \mathbf{W}'\mathbf{x}_{-v} + \mathbf{c}(\mathbf{W}, n)$, where $\mathbf{c}(\mathbf{W}, n)$ is a constant vector depending only on $\mathbf{W}$ and the size of the database; and ii) any two databases $\mathbf{y}$ and $\mathbf{z}$ are neighbors under G if and only if $\mathbf{y}_{-v}$ and $\mathbf{z}_{-v}$ are neighbors under G'.*

[1]While we that require $V(G) = V(G')$, the proof does not require G' to be a subgraph of G (i.e., $E' \subseteq E$). But it suffices for the applications of this technique in this paper.

Workload	Error per query		
	Blowfish		ϵ-Diff. [18]
$\mathbf{R}_k$	G^1_k	$\Theta(1/\epsilon^2)$	$O(\log^3 k/\epsilon^2)$
	G^θ_k	$O(\frac{\log^3 \theta}{\epsilon^2})$	
$\mathbf{R}_{k^d}$	$G^1_{k^d}$	$O(d\frac{\log^{3(d-1)} k}{\epsilon^2})$	$O(\log^{3d} k/\epsilon^2)$
	$G^\theta_{k^d}$	$O(d^3\frac{\log^{3(d-1)} k \log^3 \theta}{\epsilon^2})$	

Figure 3: Summary of data independent error bounds.

Require:
$\mathbf{W}$ is a workload of range queries, $\mathbf{x}$ is a database.
function 1DRANGE($\mathbf{W}, \mathbf{x}$)
  $\mathbf{x}_G \leftarrow \mathbf{P}^{-1}_{G^1_k}\mathbf{x}$ // *prefix sums from* $\mathbf{x}$
  $\tilde{\mathbf{x}}_G \leftarrow$ Differentially private estimate for $\mathbf{x}_G$
  $\mathbf{W}_G \leftarrow \mathbf{W}\mathbf{P}_{G^1_k}$ // *differences between prefix sum pairs*
  return $\mathbf{W}_G\tilde{\mathbf{x}}_{G^1_k}$

Algorithm 1: 1D range queries.

Technical details of the construction and proof of correctness are presented in the full paper.

EXAMPLE 4.1. *Recall the $\mathbf{C}_k$ workload from Figure 1. In $\mathbf{C}_k$, the last row computes n, the size of the database. Since we already know n, we do not need to answer that query privately. We can equivalently consider a workload $\mathbf{C'}_k$ with all zeros in the last row and removing the last column (since it would have all zeros). We can also remove the all zero row that remains resulting in a $(k-1)$ by $(k-1)$ matrix. Consider the line graph with k nodes connected in a path. We can replace the rightmost node with $\perp$ (Figure 2) to get G'. $\mathbf{P}_{G'}$ is a $(k-1)\times(k-1)$ matrix that is full rank, and $\mathbf{P}^{-1}_{G'}$ is equal to $\mathbf{C'}_k$.*

Thus, by Theorem 4.3 and Lemma 4.10, the minimum error for answering $\mathbf{C}_k$ under Blowfish policy G^1_k is equal to the minimum error for answering $\mathbf{C'}_k \cdot \mathbf{P}_{G'} = \mathbf{I}_{k-1}$ under ϵ-differential privacy. Since $\mathbf{I}_{k-1}$ is the identity workload, an optimal data independent strategy would be to add Laplace noise to yield a total error of $\Theta(k/\epsilon^2)$.

5. BLOWFISH PRIVATE MECHANISMS

In this section, we derive mechanisms (with near optimal data independent error) for answering range queries and histograms under Blowfish policies to illustrate the power of the transformational equivalence theorem. In Section 5.1 we define the types of queries and graphs we will be focusing on. In Sections 5.2 and 5.3 we present strategies for answering multi-dimensional range queries under the grid graph policy. These algorithms are data independent and incur the same error on all datasets. In Section 5.4, we extend our strategies to get data dependent algorithms for Blowfish. Figure 3 summarizes our data independent error bounds. The error incurred by data dependent versions of our algorithms will be evaluated in Section 6.

5.1 Workloads and Policy Graphs

Consider a multidimensional domain $\mathcal{T} = [k]^d$, where $[k]$ denotes the set of integers between 1 and k (inclusive). The size of each dimension is k and thus the domain size is k^d. A database in this domain can be represented as a (column) vector $\mathbf{x} \in \mathbb{R}^{k^d}$ with each entry $\mathbf{x}_i$ denoting the true count of a value $i \in \mathcal{T}$. It is important to note that our results in this paper can be easily extended to the case when dimensions have different sizes.

A multidimensional range query can be represented as a d-dimensional hypercube with the bottom left corner $\mathbf{l}$ and the top right corner $\mathbf{r}$. In particular, when $d = 1$, a range query $\mathbf{q}(\mathbf{l}, \mathbf{r})$ is a linear counting query which count the values within $\mathbf{l}$ and $\mathbf{r}$ in the database $\mathbf{x}$, i.e., $\mathbf{q}(\mathbf{l}, \mathbf{r})\mathbf{x} = \sum_{\mathbf{l}\le i\le \mathbf{r}} \mathbf{x}_i$. Let $\mathbf{R}_k$ denote the workload of all such one dimensional range queries, i.e., $\mathbf{R}_k = \{\mathbf{q}(\mathbf{l}, \mathbf{r}) \mid \mathbf{l}, \mathbf{r} \in [k] \wedge \mathbf{l} \le \mathbf{r}\}$. Similarly, let $\mathbf{R}_{k^d} = \{\mathbf{q}(\mathbf{l}, \mathbf{r}) \mid \mathbf{l}, \mathbf{r} \in [k]^d \wedge \mathbf{l} \le \mathbf{r}\}$ denote the workload of all d-dimensional range queries. Note that each range query can be represented as a k^d-dimensional row vector, and $\mathbf{R}_{k^d}$ can be represented as a $q \times k^d$ matrix, where $q = (k(k-1)/2)^d$ is the total number of range queries.

The class of policy graphs $G^\theta_{k^d} = (V, E)$ we consider here are called *distance-threshold* policy graphs. They are defined based on the L_1 distance in the domain $\mathcal{T} = [k]^d$. Consider two vertices $u = (u_1, \ldots, u_d)$ and $v = (v_1, \ldots, v_d) \in V \subseteq [k]^d$, the L_1 distance between is $|u - v| = |u_1 - v_1| + \cdots + |u_d - v_d|$. There is an edge (u, v) in E if and only if $|u - v| \le \theta$. Two special cases of $G^\theta_{k^d}$ and their semantics were discussed in Section 3 as line graph (G^1_k) and grid graph ($G^\theta_{k^2}$).

We note that the data independent mechanisms we present for one dimensional range queries under G^1_k and G^θ_k (Sections 5.2.1 and 5.3.1) are similar to the ones presented in the original Blowfish paper [11]. We present them here to illustrate our transformational equivalence and subgraph approximation results, and to help the reader understand our novel mechanisms for multi-dimensional range queries.

5.2 Range Queries under $G^1_{k^d}$

In this section we first describe the easy case of 1D range queries before considering multi-dimensional range queries. We will heavily utilize the structure of the transformed query workload in this section. The following lemma helps relate the queries in $\mathbf{W}$ to the queries in $\mathbf{W}_G$.

LEMMA 5.1. *Let $\mathbf{q}$ be a linear counting query (that is, all entries in $\mathbf{q}$ are either 1 or 0), and $G = (V, E)$ be a policy graph. Let $\{v_1, \ldots, v_\ell\} \subseteq V$ be the vertices corresponding to the nonzero entries of $\mathbf{q}$. Then, the nonzero columns of $\mathbf{q} \cdot \mathbf{P}_G = \mathbf{q}_G$ correspond to the set of edges (u, v) with exactly one end point in $\{v_1, \ldots, v_\ell\}$. That is,*

$$\{(u, v) : |\{u, v\} \cap \{v_1, \ldots, v_\ell\}| = 1\}.$$

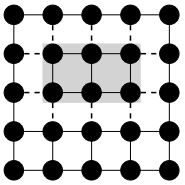
PROOF. Each entry c of $\mathbf{q}_G$ satisfies $c = u - v$ where u, v are entries in $\mathbf{q}$ and $(u, v) \in E$. c is nonzero exactly when $u \ne v$, or equivalently, when

$$|\{u, v\} \cap \{v_1, \ldots, v_k\}| = 1. \quad \square$$

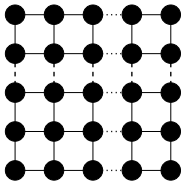
5.2.1 $\mathbf{R}_k$ under G^1_k

We begin with a simple case: one-dimensional range queries under a one-dimensional line graph. We outline the application of Theorem 4.3 in Algorithm 1. Recall from Example 4.1 that the inverse of $\mathbf{P}_{G^1_k}$ is the cumulative histogram workload. Therefore, the transformed database $\mathbf{x}_G = \mathbf{P}^{-1}_{G^1_k}\mathbf{x}$ corresponds to the set of prefix sums in $\mathbf{x}$. Algorithm 1 computes a differentially private estimate of $\mathbf{x}_G$ (say using the Laplace mechanism).

Figure 4: A one dimensional range query on vertices is transformed into a query on edges (represented by dashed lines).



(a) $G^1_{5^2}$ with a two dimensional range query, represented by a grey box. The edges in the transformed query (satisfying Lemma 5.1), are shown with dashed lines. These edges form four ranges.



(b) For each row of vertical edges, we answer all ranges over the row. One such row is shown with dashed lines. We must do the same for columns, and one such column is shown with dotted lines.

Figure 5: Answering $\mathbf{R}_{k^2}$ under $G^1_{k^2}$.

Next, we transform the queries. Note that for any range query $\mathbf{q} = [l, r]$, by Lemma 5.1 $\mathbf{q}_G$ contains at most two nonzero elements (this is illustrated in Figure 4) corresponding to the edges $(l-1, l)$ and $(r, r+1)$. The values of $\mathbf{x}_G$ at these edges are the prefix sums $\sum_{i=1}^{l-1} x_i$ and $\sum_{i=1}^{r} x_i$, and their difference is indeed the answer to the original range query. We can show the following bound on the data independent error of Algorithm 1.

THEOREM 5.2. *Algorithm 1 with $\tilde{\mathbf{x}}_{G^1_k} = \mathbf{x}_{G^1_k} + Lap(1/\epsilon)$ answers workload $\mathbf{R}_k$ with $\Theta(1/\epsilon^2)$ error per query under (ϵ, G^1_k)-Blowfish privacy.*

PROOF. Every $\mathbf{q}_{G^1_k}(l, r) \in \mathbf{R}_{G^1_k}$ can be reconstructed by summing at most two queries in $\mathbf{x}_{\tilde{G^1_k}}$. Each entry in $\mathbf{x}_{\tilde{G^1_k}}$ has $\Theta(1/\epsilon^2)$ error from the Laplace mechanism. So each $\mathbf{q}_{G^1_k}(l, r)$ incurs only $\Theta(1/\epsilon^2)$ error. □

In fact we show that Algorithm 1 is the optimal data independent algorithm for answering range queries in 1D under G^1_k. The proof appears in the full version of the paper [8].

LEMMA 5.3. *Any (ϵ, G^1_k)-Blowfish private mechanism answers $\mathbf{R}_k$ with $\Omega(1/\epsilon^2)$ error per query.*

The best known data independent strategy (with minimum error) for answering $\mathbf{R}_k$ under ϵ-differential privacy is the Privelet strategy [18] with a much larger asymptotic error of $O(\log^3 k/\epsilon^2)$ per query.

5.2.2 $\mathbf{R}_{k^d}$ under $G^1_{k^d}$

$G^1_{k^d}$ is a grid with k^d vertices and $d \cdot (k-1) \cdot k^{d-1}$ edges. Let us consider the problem in two dimensions first. (see Figure 5a). The transformed domain (after using Theorem 4.3) would be the set of edges in the graph. Consider a 2D range query $\mathbf{q}([x, y], [x', y'])$ (grey box in figure). The transformed query $\mathbf{q}_G$ has non-zero entries corresponding to edges on the boundary of the original range query (dashed lines in the figure). Note that these edges can be divided into 4 contiguous ranges of edges; i.e., $\mathbf{q}_G$ is the sum of 4 disjoint range queries in the transformed domain.

Thus, a strategy for answering the transformed query workload in two dimensions would be to answer all one dimensional range queries along the rows (dashed vertical edges in Fig 5b) and columns (dotted horizontal edges in Fig 5b) under differential privacy. There are $2(k-1)$ such sets of range queries. Note that these sets of range queries are disjoint, and can each be answered using ϵ-differential privacy (under parallel composition). Any query $\mathbf{q}_G$ can be computed by adding up the answers to 2 row range queries and 2 column range queries.

In d dimensions, $\mathbf{q}_G$ will be the sum of $2d$ $(d-1)$-dimensional range queries on the transformed dataset $\mathbf{x}_G$, each corresponding to a face of the d-dimensional range query. Our strategy would be to answer $d(k-1)$ sets of $(d-1)$-dimensional range queries under ϵ-differential privacy.

We bound the data independent error of our algorithm.

THEOREM 5.4. *Workload $\mathbf{R}_{k^d}$ can be answered with*

$$O(d \log^{3(d-1)} k/\epsilon^2)$$

error per query under $(\epsilon, G^1_{k^d})$-Blowfish privacy.

PROOF. For each dimension, we must answer $k-1$ sets of $(d-1)$-dimensional range queries, for a total of $(k-1) \cdot d$ sets of $(d-1)$-dimensional ranges. As we have shown, all of these sets are disjoint and can be answered in parallel. Therefore, the total error is just the error of answering one of these sets of ranges. We can answer these ranges using Privelet [18] with $O(\frac{\log^{3(d-1)} k}{\epsilon^2})$ error. To answer our query, we must sum $2d$ of these ranges for a total error of $O(d\frac{\log^{3(d-1)} k}{\epsilon^2})$.

By Theorem 4.3, we can answer $\mathbf{R}_{k^d}$ under $G^1_{k^d}$ with the same error per query. □

We get a $\Omega(\log^3 k)$ factor better error than differential privacy using Privelet [18] under a fixed dimensionality d.

5.3 Range Queries under $G^\theta_{k^d}$

We next consider answering range queries under a more complex graph. Unlike in the case of $G^1_{k^d}$, the workloads resulting from the use of Theorem 4.3 to the $G^\theta_{k^d}$ policy are not well studied under differential privacy. We will use subgraph approximation to design Blowfish private mechanisms.

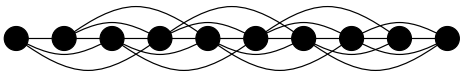
5.3.1 $\mathbf{R}_k$ under G^θ_k

We next present an algorithm for answering one dimensional range queries under, G^θ_k. These results generalize the results from Section 5.2.1, and will leverage subgraph approximation (Lemma 4.5).

We first describe how to obtain a subgraph H^θ_k from G^θ_k. We designate k/θ vertices at intervals of θ; call these "red" vertices. In H^θ_k, consecutive red vertices are connected to form a path (like the line graph). All non-red vertices are only connected to the next red vertex (to its right); i.e., vertices $\{1, 2, \ldots, \theta-1\}$ are connected only to vertex θ, vertices $\{\theta+1, \theta+2, \ldots, 2\theta-1\}$ are connected only to vertex 2θ, and so on. We order the edges in H^θ_k by their left endpoints.

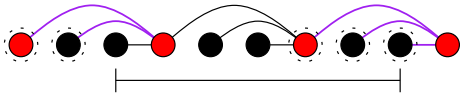
Like G^1_k, H^θ_k is also a tree with $k-1$ edges. Figure 6a shows G^3_{10} and Figure 6b shows H^3_{10}. Note that for all θ, a pair of adjacent vertices in G^θ_k are connected by a path of length $\ell \leq 3$ in H^θ_k. So we can use subgraph approximation.

Consider some query in $\mathbf{R}_k$, say $\mathbf{q}(l, r)$. The corresponding query $\mathbf{q}_{H^\theta_k}$ in $\mathbf{R}_{H^\theta_k}$ consists of all edges with one of



(a) G^3_{10}, each vertex is connected to other vertices within distance 3 along the line.

(b) H^3_{10}, each vertex is connected to the nearest red vertex to its right.



(c) A range query shown on H^3_{10}. The transformed query consists of the edges highlighted in purple and their left end points always form two contiguous ranges.

(d) Our strategy will answer all range queries on 3 sets of edges, each set shown in a different color. These sets of edges are disjoint.

Figure 6: A summary of a strategy for answering $\mathbf{R}^k$ under G^θ_k for $\theta = 3, k = 10$. Our results hold in general.

the end points within the range (l, r) (Lemma 5.1). If $l \leq x\theta \leq r \leq y\theta$, where $x\theta$ and $y\theta$ are the smallest red nodes greater than l and r, then these edges correspond to $\{(i, x\theta) \mid (x-1)\theta \leq i < l\}$ and $\{(j, y\theta) \mid (y-1)\theta \leq j < r\}$ (edges connected to dotted nodes in Figure 6c). That is, the transformed query $\mathbf{q}_{H^\theta_k}(l, r)$ corresponds to the difference of two range queries (according to the ordering of edges in H^θ_k). Moreover, each range query is of length at most θ – within $[(x-1)\theta, x\theta]$ for some x.

Thus, our strategy for answering all the queries in $R_{H^\theta_k} = R_k \cdot P_{H^\theta_k}$ is as follows. Partition the transformed domain (or edges in H^θ_k) into disjoint groups of θ – all edges connecting a red node to nodes on its left form a group (see Figure 6d). Next, answer all range queries of length at most θ within each of these groups under ϵ-differential privacy (say using Privelet). Finally, reconstructing queries $\mathbf{q}_{H^\theta_k}(l, r) \in R_{H^\theta_k}$ using the computed range queries. Since these sets of range queries form disjoint subsets of the domain, they all can use the same ϵ privacy budget (by parallel composition).

THEOREM 5.5. *There exists a mechanism that answers workload $\mathbf{R}_k$ with*

$$O\left(\log^3 \theta/\epsilon^2\right)$$

error per query under (ϵ, G^θ_k)-Blowfish privacy.

PROOF. Our strategy partitions the transformed domain (or edges in H^θ_k) into groups of size θ, and answers range queries over them. Using Privelet to compute range queries within each partition results in $O(\log^3 \theta/\epsilon^2)$ error per query. Queries in $R_{H^\theta_k}$ are differences of at most 2 range queries, and thus also incur at most $O(\log^3 \theta/\epsilon^2)$ error. By Theorem 4.3, the same mechanism answers $\mathbf{R}_k$ and satisfies (ϵ, H^θ_k)-Blowfish privacy with $O(\log^3 \theta/\epsilon^2)$ error per query.

For every edge $(u, v) \in G^\theta_k$, u and v are connected by a path of length at most 3, this strategy also ensure $(3\epsilon, G^\theta_k)$-Blowfish privacy. Thus, using the above strategy with privacy budget $\epsilon/3$ gives us the required result. □

5.3.2 $\mathbf{R}_{k^d}$ under $G^\theta_{k^d}$

We now turn our attention to multidimensional range queries under $G^\theta_{k^d}$. Our strategy will be similar to the one in Section 5.3.1. We find a subgraph to approximate $G^\theta_{k^d}$. We show the queries of the transformed workload can be decomposed into range queries on edges of bounded size, and our strategy is to answer these range queries.

To get a subgraph H^θ_k, we divide $G^\theta_{k^d}$ into d-dimensional hypercubes with edge length $\frac{\theta}{d}$ (see Figures 7a and 7b). We designate the vertices at the corners of the cubes as "red" vertices. We pick a mapping of hypercubes to red vertices. For example, in the 2-dimensional case, we may map each square to its upper right red vertex. Each non-red vertex within a cube is connected to this selected red vertex (we pick a consistent mapping for vertices that are on the boundary of cubes). We call these *internal* edges. The red vertices are then connected in a grid (like $G^1_{k^d}$) with *external* edges.

Due to space constraints we give a brief sketch of our strategy. Given a range query $\mathbf{q}$, the transformed query will correspond to a set of internal and a set of external edges (as per Lemma 5.1). Since external and internal edges are disjoint, our strategy answers the transformed query restricted to the external edges and internal edges independently, each under ϵ-differential privacy.

Since the external edges form a grid graph $G^1_{(k/\theta)^d}$ (Figure 7c), we can use our strategy from Section 5.2.2 to answer this part of the transformed query with error at most $O(d^{\frac{\log^{3(d-1)} d \cdot k/\theta}{\epsilon^2}})$. We can show that the non-red end points of the internal edges featuring in the transformed query for $2d$ d-dimensional range queries. However, each of these range queries has a width at most θ in one of the dimensions (see Figure 7d). Thus like in Section 5.2.2, it is sufficient to answer all d-dimensional range queries having width at most θ in one dimension. However, the edges in these range queries are not disjoint, and hence we can only use θ/d privacy budget for answering these range queries, resulting in the following error bound:

THEOREM 5.6. *Workload $\mathbf{R}_{k^d}$ can be answered with*

$$O(d^3 \cdot \frac{\log^{3(d-1)} k \log^3 \theta}{\epsilon^2})$$

error per query under $(\epsilon, G^\theta_{k^d})$-Blowfish privacy.

Discussion. Our Blowfish mechanisms under $G^1_{k^d}$ and $G^\theta_{k^d}$ policy graphs improve upon Privelet by a factor of $\log^3 k$, but incur additional error by a factor of d and $d^3 \log^3 \theta$ respectively. Thus the proposed mechanisms are better than using Privelet when $d \log \theta$ is small compared to $\log k$. This is true in the case of location privacy where $d = 2$ and θ (10s of km) is usually much smaller than k (1000s of km).

5.4 Data Dependent Algorithms

Till now we considered data independent Blowfish mechanisms whose error is independent of the input database. Recent work has investigated a new class of data dependent algorithms for answering histogram and range queries

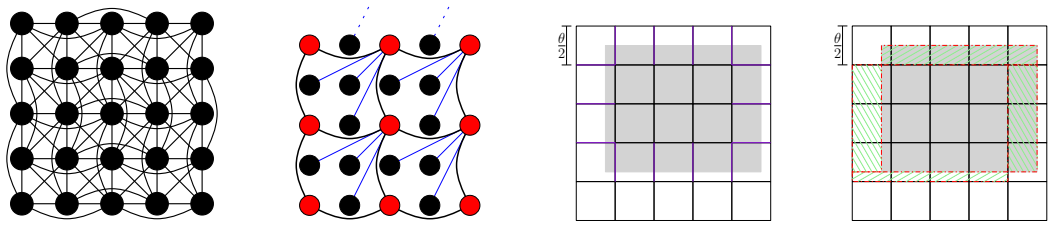


Figure 7: Transforming queries in $\mathbf{R}_{k^2}$ under $G^{\theta}_{k^2}$. (a) $G^2_{5^2}$, edges are vertices with L_1 distance ≤ 2. (b) A section of $H^2_{k^2}$. Internal edges connect red nodes and external edges connect black nodes to red nodes. (c) A 2D range query superimposed on $H^2_{k^2}$. We only show the divisions in $\theta/2$ blocks of vertices. Within each block, all vertices would be connected to the upper right corner. The grid of lines shows all the external edges. Highlighted in purple are the external edges which satisfy Lemma 5.1 and therefore appear in the transformed query. (d) The green patterned rectangles show the sets of vertices corresponding to the internal edges in the transformed query. There are 4 such rectangles, and for each one either the height or length is bounded by θ. Note that there are other ways in which we could divide the green patterned region into 4 rectangles, we arbitrarily chose one. Our strategy is to answer all range queries over each row of squares, and each column of squares. We illustrate using specific values for θ and k, but our results hold in general.

that exploit the properties of the data and incur much lower error on some (typically sparse) datasets. In this section, we present two methods for adapting the previously given mechanisms to get data dependent mechanisms for Blowfish.

5.4.1 *Using data dependent DP algorithms*

In all of our algorithms we employed data independent differentially private algorithms in our strategies. We used the Laplace mechanism for computing noisy histograms in our strategy for answering $\mathbf{R}_k$ under G^1_k, and used Privelet for answering noisy range queries in all the other cases. Instead, when the policy graph is a tree, we could use a state-of-the-art data dependent technique like DAWA [13] for answering histograms and range queries under differential privacy. For instance, DAWA computes a noisy histogram as follows: (a) partition the domain such that domain values within a group have roughly the same counts, (b) estimates the total counts for each of these groups using the Laplace mechanism, and (c) uniformly divides the noisy group totals amongst its constituents. When many counts are similar (especially when $\mathbf{x}$ is sparse), DAWA incurs lower error than Laplace mechanism since it adds noise to fewer counts (see Section 6).

5.4.2 *Using properties of the transformed database*

All the example workload/policy pairs discussed in this section are such that the transformed workload $\mathbf{W}_G$ is "easier" to answer under differential privacy than $\mathbf{W}$. Thus, the data independent Blowfish algorithms outperform the data independent differentially private algorithms for answering $\mathbf{W}$ on $\mathbf{x}$ by more than a constant factor. However, this is not true for all workloads.

Consider, for instance, the identity workload $\mathbf{I}_k$ that computes the histogram of counts under the line graph G^1_k. The transformed workload $\mathbf{I}_{G^1_k}$ is the set of differences between adjacent elements in the transformed database; i.e., $\mathbf{x}_G[i] - \mathbf{x}_G[i-1]$, for all i. The transformed workload seems no easier than the original workload.

However, we can utilize the fact that $\mathbf{x}_G$ has special structure. Recall that $\mathbf{P}^{-1}_{G^1_k}$ is precisely equal to the cumulative histogram workload $\mathbf{C}_k$. Thus, the counts in the transformed database $\mathbf{x}_G$ are prefix sums of the counts in $\mathbf{x}$, and are non-decreasing. We can use this property of $\mathbf{x}_G$ to reduce error by enforcing the non-decreasing constraint on the noisy counts $\tilde{\mathbf{x}}_G$. Hay et al [10] present a simple algorithm to postprocess the counts so that the error depends on the number of distinct counts in $\mathbf{x}_G$. Note that whenever a count in $\mathbf{x}$ is 0, a pair of consecutive prefix sums in $\mathbf{x}_G$ are the same. Therefore, the number of distinct values in $\mathbf{x}_G$ is precisely the number of non-zero entries in $\mathbf{x}$. This suggests that postprocessing $\tilde{\mathbf{x}}_G$ to ensure that the counts are non-decreasing will lead to a significant reduction of error for sparse datasets. We can use this strategy whenever $\mathbf{P}^{-1}_G$ creates constraints in $\mathbf{x}_G$.

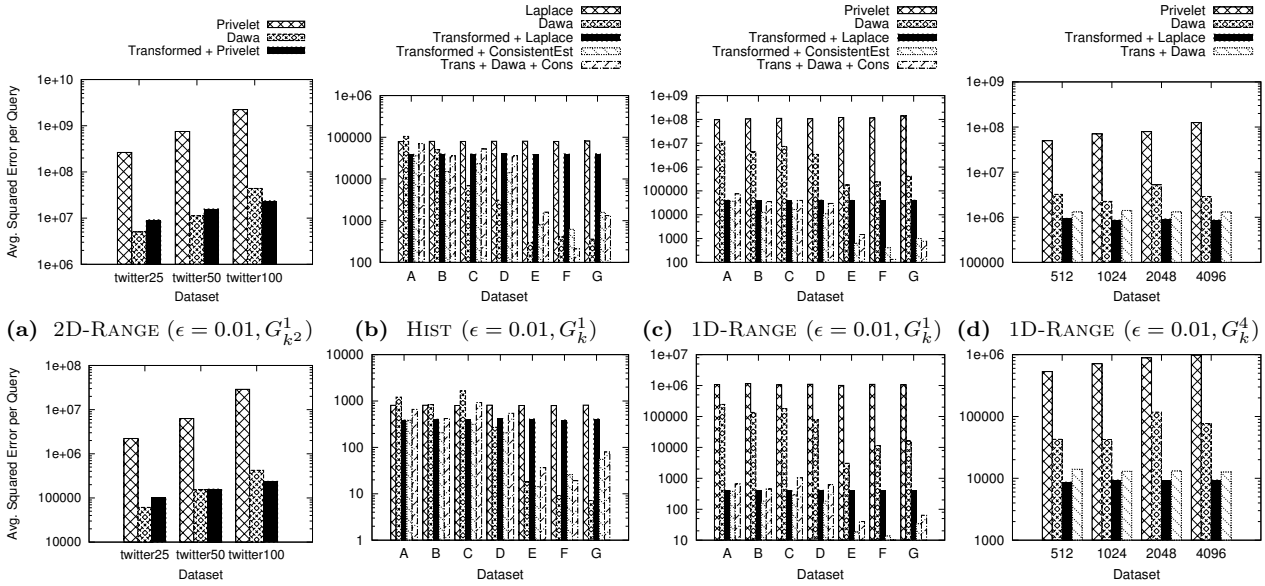
6. EXPERIMENTS

In Section 5, we outlined a number of algorithms for answering range query and marginal workloads under the distance threshold policies, and derived data independent error bounds for them. In this section, we implement both data independent and data dependent versions of these algorithms and empirically evaluate their error on a number of real one and two dimensional datasets. In particular, we compare the error attained by $\epsilon/2$-differentially private algorithm for a task to that of (ϵ, G)-Blowfish mechanisms for the same task (since a $\epsilon/2$-differentially private algorithm also satisfies (ϵ, G)-Blowfish privacy for all G). The highlights of this section are:

- For 1-D range queries, since the policy graphs are "tree-like", we can design data dependent algorithms for Blowfish by utilizing state-of-the-art data dependent algorithms for differential privacy, thanks to Theorem 4.3. The transformed workload is simpler and permits an order of magnitude improvement in error for both the data independent and data dependent implementations.
- For 2-D, we are not aware of a low stretch embedding of the grid policy graph to a tree. Though we are restricted to the use of matrix mechanism algorithms, our new Blowfish private algorithms outperform the best data dependent differentially private algorithms on sparse datasets.
- For histograms, while the transformed workload is not easier than the original, we can exploit constraints in the

	Description	Domain Size	Scale	% Zero Counts
A	Histogram of new links by time added to a subset of the US patent citation network	4096	2.8×10^7	6.20
B	Histogram of personal income from 2001-2011 American community survey	4096	2.0×10^7	44.97
C	Histogram of new links by time added to HepPH citation network	4096	3.5×10^5	21.17
D	Frequency of search term "Obama" over time (2004-2010)	4096	3.4×10^5	51.03
E	Number of external connections made by each internal host in an IP-level network trace collected at the gateway router of a major university.	4096	2.6×10^4	96.61
F	Histogram on "capital loss" attribute of Adult US Census dataset	4096	1.8×10^4	97.08
G	Histogram of personal medical expenses based on a national home and hospice care survey from 2007	4096	9.4×10^3	74.80
T100	Aggregated counts of number of tweets by geo location	100×100	1.9×10^5	84.93
T50	collected over 24 hours restricted to a bounding box of	50×50	1.9×10^5	69.24
T25	50N,125W and 30N,110W (western USA)	25×25	1.9×10^5	43.20

Table 1: Description of datasets



(a) 2D-RANGE ($\epsilon = 0.01, G^1_{k^2}$) **(b)** HIST ($\epsilon = 0.01, G^1_k$) **(c)** 1D-RANGE ($\epsilon = 0.01, G^1_k$) **(d)** 1D-RANGE ($\epsilon = 0.01, G^4_k$)

(e) 2D-RANGE ($\epsilon = 0.1, G^1_{k^2}$) **(f)** HIST ($\epsilon = 0.1, G^1_k$) **(g)** 1D-RANGE ($\epsilon = 0.1, G^1_k$) **(h)** 1D-RANGE ($\epsilon = 0.1, G^4_k$)

Figure 8: Comparison of $\epsilon/2$-Differentially private and (ϵ, G)-Blowfish algorithms for four workloads.

transformed database to obtain improvements in error.

Datasets: We evaluate the error on 7 different one dimensional datasets and 3 two dimensional dataset (see Table 1). All the one dimensional datasets A-G have the same domain size (4096) but vary in their scale (total number of records), and were used in prior work (notably [13]). We aggregate our two dimensional dataset to a domain size of 100×100 (T100), 50×50 (T50), and 25×25 (T25), by imposing uniform grids of appropriate size on the space. Finally, we aggregate over dataset "D" to domain sizes 4096 (no aggregation), 2048, 1024, and 512. Note that most datasets are sparse (low scale and high % of zero counts).

Policies: We use G^1_k and G^4_k for one dimensional and $G^1_{k^2}$ for two dimensional datasets.

Workloads: We consider four workloads. 1D-RANGE is 10,000 random one-dimensional range queries. 2D-RANGE is 10,000 random two dimensional range queries. HIST is the histogram workload. We report the average mean square error over 5 independent runs, and use $\epsilon \in \{0.001, 0.01, 0.1, 1\}$. Results for $\epsilon \in \{0.001, 1\}$ are deferred to the full version.

6.1 Results

HIST: We compare 5 algorithms on our 1-D datasets under policy graph G^1_k. We use Laplace mechanism and DAWA [13], the best data independent and state-of-the-art data dependent differentially private algorithms for the workload, resp. For Blowfish, we use Laplace mechanism on the transformed database to get a data independent strategy (Section 5.4). We called this "Transformed + Laplace". Since G^1_k is a tree, we can construct two data dependent Blowfish algorithms as follows: (1) We use the consistency postprocessing algorithm to the output of Transformed + Laplace to ensure that the noisy counts $\tilde{\mathbf{x}}_G$ are non-decreasing. We call this "Transformed + ConsistentEst". (2) We compute a noisy histogram on $\mathbf{x}_G$ using DAWA and then apply consistency (called "Transformed + Dawa + Cons").

We see that the (ϵ, G)-Blowfish data independent technique (Transformed + Laplace) is only a factor of 2 better than the data independent $(\epsilon/2)$-differentially private algorithm (Laplace Mechanism). Significant gains in error over the data independent methods are seen in sparse datasets E, F & G by using DAWA (for differential privacy) and the

two data dependent Blowfish algorithms (note log scale on y-axis). When $\epsilon = 0.1$ (and 1) one of the Blowfish data dependent algorithms outperform the two differentially private mechanism on all but two datasets F and G. These are very sparse, and DAWA achieves lower error. On the other hand, the transformed database (which is the set of prefix counts) is not as sparse, and thus Blowfish algorithms achieve higher error. At smaller ϵ values $(0.01, 0.001)$, DAWA outperforms one or both the data dependent Blowfish algorithms on all datasets except A and B. Designing data dependent Blowfish mechanism for HIST under G^1_k with "optimal" error is an interesting open question.

1D-RANGE: First, we consider the G^1_k policy graph. We consider Privelet and DAWA as the data independent and dependent algorithms under differential privacy, resp. The Blowfish data independent strategy is to use Laplace mechanism on the transformed database. We again implement two data dependent algorithms for Blowfish – enforce the non-decreasing constraint on the noisy $\tilde{\mathbf{x}}_G$ computed using the Laplace mechanism and DAWA, resp. In this case, we see 2-3 orders of magnitude difference in the error of all the Blowfish algorithms from their differentially private counterparts! This is because both the transformed workload and the transformed database are "easier" that the original workload and database. We observe that while the Blowfish data dependent strategy using DAWA is better than the one using Laplace mechanism on all datasets when $\epsilon = 1$, the reverse is true for $\epsilon = 0.1, 0.01$. We conjecture this is true because the lower privacy budget results in poorer data dependent clustering for DAWA.

We also study the error incurred under the G^4_k policy graph under datasets of varying domain sizes ($k = 4096, 2048, 1024, 512$). Since G^4_k is not a tree, we use a spanning tree H^4_k as described in Section 5.3.1 and Figure 6. Since G^4_k can be embedded into H^4_k with a stretch of 3, by Corollary 4.6 an $\epsilon/3$-differentially private mechanism for answering $\mathbf{W}_{H^4_k}$ on $\mathbf{x}_{H^4_k}$ also is a (ϵ, G^4_k)-Blowfish private mechanism for answering $\mathbf{W}$ on $\mathbf{x}$. The Blowfish data independent algorithm is Laplace mechanism on the transformed database. The data dependent algorithm is DAWA on the transformed workload and database. Both use privacy budget $\epsilon/3$. Again, the blowfish mechanisms have at least an order of magnitude smaller error than their differentially private counterparts. While the error for the differentially private algorithms increases as the domain size increases, the error for the Blowfish mechanisms does not change with domain size. This is because the transformed workload is like the identity matrix. While the Blowfish data dependent algorithm is better than using Laplace mechanism for $\epsilon = 1$, it is worse for smaller ϵ.

2D-RANGE: We consider three algorithms for the grid graph policy $G^1_{k^2}$. Privelet and DAWA are the data independent and data dependent differentially private algorithms. The blowfish data independent strategy is to use Privelet for the one dimensional range queries in the transformed workload. We do not know of a data dependent algorithm under Blowfish for $G^1_{k^2}$, since it is not "tree-like". Though each transformed query requires 4 one dimensional range queries (over the transformed database), we still see that the Blowfish algorithm (a) significantly outperforms Privelet, and (b) improves over DAWA when the domain size is large.

7. CONCLUSIONS

We systematically analyzed error bounds on linear query workloads under the Blowfish privacy framework. We showed that the error incurred when answering a workload under Blowfish is identical to the error incurred when answering a transformed workload under differential privacy for a large class of privacy mechanisms and graphs. This, in conjunction with a subgraph approximation result, helped us derive strategies for answering linear counting queries under the Blowfish privacy framework. We showed that workloads can be answered with significantly smaller amounts of error per query under Blowfish privacy compared to differential privacy, suggesting the applicability of Blowfish privacy policies in practical utility driven applications.

Acknowledgements: This work was supported by the National Science Foundation under Grants 1253327, 1408982, 1443014 and a gift from Google.

8. REFERENCES

[1] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi. Geo-indistinguishability: Differential privacy for location-based systems. In *ACM CCS*, pages 901–914. ACM, 2013.

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

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

[4] C. Dwork, M. Hardt, T. Pitassi, O. Reingold, and R. Zemel. Fairness through awareness. In *TCS*, pages 214–226. ACM, 2012.

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

[6] C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In *STOC*, pages 715–724, 2010.

[7] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 448–455. ACM, 2003.

[8] S. Haney, A. Machanavajjhala, and B. Ding. Design of policy-aware differentially private algorithms. *CoRR*, abs/1404.3722, 2014.

[9] M. Hardt and K. Talwar. On the geometry of differential privacy. In *STOC*, pages 705–714, 2010.

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

[11] X. He, A. Machanavajjhala, and B. Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *SIGMOD*, 2014.

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

[13] C. Li, M. Hay, and G. Miklau. A data- and workload-aware algorithm for range queries under differential privacy. *To appear Proc. VLDB Endow.*, 2014.

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

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

[16] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

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

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