Answering Query Workloads with Optimal Error under Blowfish Privacy

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

November 6, 2014

Abstract

Recent work has proposed a privacy framework, called Blowfish, that generalizes differential privacy in order to generate principled relaxations. Blowfish privacy definitions take as input an additional parameter called a policy graph, which specifies which properties about individuals should be hidden from an adversary. An open question is to characterize when Blowfish privacy definitions permit mechanisms that incur significantly lower error for query answering compared to differentially private mechanisms. In this paper, we answer this question and explore error bounds for answering sets of linear counting queries under different instantiations of Blowfish privacy.

We first develop theoretical tools relating query answering under Blowfish to query answering under differential privacy. In particular, we prove a surprising equivalence between the minimum error required to answer a workload $W$ under a Blowfish policy $G$ and the minimum error required to answer a workload $W_G$ (constructed using $W$ and $G$) under differential privacy. We provide applications of these tools by finding strategies for answering multidimensional range queries under different Blowfish policy graphs. We believe the tools we develop will be useful for finding strategies to answer many other classes of queries with low error under Blowfish. Next, we generalize the matrix mechanism lower bound of Li and Miklau (called the SVD bound) for differential privacy to find an analogous lower bound for Blowfish, and illustrate our bounds using multidimensional range queries.

1 Introduction

With increasingly large datasets becoming available, it is useful to be able to release this data for research purposes without violating the privacy of individuals in the dataset. $\epsilon$-Differential privacy [2] has become the standard for private release of data due to its strong guarantee that the output of any algorithm run on the private data does not change significantly if a single individual’s record is added, removed or changed. Typical algorithms that satisfy differential privacy release noisy answers. The privacy parameter $\epsilon$ controls the amount of noise, and thus can be used to trade-off privacy for utility.

However, in certain applications (e.g., [6][14]), the differential privacy guarantee is too strict to produce private release of data that has any non-trivial utility. Tuning the parameter $\epsilon$ is not helpful here: enlarging $\epsilon$ degrades the privacy guaranteed without a commensurate improvement in utility.

Recent work [11][10] has generalized the notion of differential privacy to allow data owners specify which properties of the dataset must be protected from an adversary. In particular, Blowfish privacy [10] enumerates pairs of sensitive properties about an individual that an adversary must not be able to distinguish, using what is called a “policy graph” (see Section 2.3 for more details). Blowfish privacy was applied to several practical scenarios to achieve better utility than differential privacy [10].

In this paper, we continue this line of work and systematically analyze the privacy-utility trade-offs arising from mechanisms that satisfy Blowfish privacy. Rather than developing point solutions, we present a lower bound on the minimum error with which any workload of linear queries can be answered under a specific Blowfish privacy policy, as well as general techniques to help derive near optimal strategies for (and consequently error bounds on) answering workloads of linear queries under different instantiations of Blowfish privacy.

Overview of Results. Throughout this paper, we consider privacy algorithms that are instantiations of the extended matrix mechanism [12][13]. These are data oblivious but workload dependent algorithms which privately release the answers to a workload of linear queries $W$ using a different strategy workload $A$, such that the queries in $A$ are not very sensitive to presence or absence of one individual, and query answers in $W$ can be reconstructed using a small number of answers from $A$.

We first adapt the extended matrix mechanism to the Blowfish privacy framework. Our main result in this paper is called transformational equivalence. We show that the error incurred by answering a workload $W$ using a strategy $A$ under a Blowfish privacy policy characterized
by a policy graph $G$ is equivalent to the error incurred by answering a different workload $W_G$ using strategy $A_G$ under differential privacy. Here, $W_G$ and $A_G$ are algorithmic transformations of the original workload $W$ and strategy $A$ based on the policy graph $G$.

We present (near) optimal algorithms (or upper bounds on error) for answering multidimensional range query workloads under reasonable Blowfish policy graphs $G$. Our approach for finding good strategies works as follows (Figures 4 & 5 on page 18). Given a workload $W$, we transform it into $W_G$ and find a strategy $A_G$ that answers $W_G$ under differential privacy with low error. We then transform $A_G$ to $A$, and transformational equivalence ensures that $A$ is a good strategy for answering $W$ with low error under Blowfish policy graph $G$. This approach leverages the rich literature on near optimal strategies for answering workloads under differential privacy. When $W_G$ is not a well studied workload, we consider using a different policy graph $G'$ that is a subgraph of $G$. Our subgraph approximation result ensures that a strategy for answering $W$ under policy graph $G'$ is also a good strategy (worse by a constant factor $\ell^2$) for answering $W$ under policy graph $G$ as long as neighboring nodes in $G$ are no more than a distance $\ell$ apart in $G'$.

In particular, we use the transformational equivalence and subgraph approximation results to derive strategies for 1-dimensional range query workloads under reasonable Blowfish policies with error per query that is independent of the domain size. The best known strategy under differential privacy for 1-dimensional range queries incurs an error of $O(k/\epsilon^2)$ per query, where $k$ is the domain size. Moreover, our strategy for $d$-dimensional queries ($d \geq 2$) reduces the error by a polylog factor of $k$ over differential privacy (see Figure 3 on page 7).

Additionally, transformational equivalence allows us to directly adapt SVD based extended matrix mechanism lower bounds for answering workloads under differential privacy to the Blowfish setting. We empirically verify that the error lower bounds for answering multidimensional range query workloads under reasonable Blowfish policy graphs is much smaller than the lower bound under differential privacy. This suggests that answers to query workloads may be released with significantly lower error in the Blowfish framework.

Organization. The rest of this section is a brief survey of related work. Section 2 gives background information and gives definitions that we will use throughout the paper. Section 3 generalizes the definition of the extended matrix mechanism [13] to the Blowfish privacy framework. We describe our main result, transformational equivalence, in Section 4. Section 5 provides upper bounds on the error of matrix mechanism for multidimensional range queries under various instantiations of the Blowfish framework. We believe our techniques can be used to find efficient strategies for other classes of query workloads. Section 6 gives a lower bound on the error of the extended matrix mechanism, and illustrates the lower bounds for 1- and 2-dimensional range queries. Detailed proofs have been deferred to the appendix.

Related Work. Recent work has given error bounds under differential privacy both in general, and for specific classes of workloads and mechanisms. Dwork et al. [4] show that the amount of noise needed is related to the sensitivity of queries. Nissim et al. [16] show that it is sufficient to add noise based on the smooth sensitivity. For single counting queries, it has been shown [17] that Laplace mechanism is optimal. A sequence of results [8, 1, 15] give mechanisms independent error bounds for sets of linear counting queries using geometric arguments. Li and Miklau [13] give an error lower bound for the extended matrix mechanism based on the singular value decomposition of the workload matrix.

Some recent work has attempted to provide more flexible privacy definitions. Kifer and Machanavajjhala [11] developed the Pufferfish framework which generalizes differential privacy by specifying what information should be kept secret, and the adversary’s prior knowledge. He et al. [10] propose the Blowfish framework which also generalizes differential privacy and is inspired by Pufferfish. Both 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.

2 Background and Notation

We first define standard privacy notation in the context of differentially private query workloads. We then describe the extended matrix mechanism and Blowfish privacy.

2.1 Query Workloads

Consider some dataset $D$. Let $\mathcal{T}$ be the domain of values in the dataset, and let $|\mathcal{T}| = k$. Let $\mathcal{I}_n$ be the set of databases $D$ over $\mathcal{T}$ such that $|D| = n$. Let $\mathcal{I}$ be the set of databases with any number of entries. A workload is a set of linear counting queries. A workload can be represented as a $q \times k$ matrix $W$, where $q$ is the number of queries. Each row of this matrix corresponds to a query. The columns represent values $x \in \mathcal{T}$. The true answer to this workload will be a vector in $\mathbb{R}^\mathcal{T}$ where the $i^{th}$ entry in the vector is the answer to the query represented by the $i^{th}$ row in the matrix. Let $x \in \mathbb{R}^k$ be the true counts of all values in the domain of database values. Then $W \cdot x$ will be the true answer to this workload.

Example 2.1. Figure 7 shows examples of two well studied workloads. $I_k$ is the identity matrix representing the histogram query on $\mathcal{T} = \{x_1, x_2, \ldots, x_k\}$. $C_k$ corresponds
to the cumulative histogram workload, where each query
 corresponds to the sum of the counts of values from \( x_i \)
 through \( x_k \). Cumulative histograms have many applica-
tions in releasing cdfs, quantiles, answering range queries
\cite{12,13}, and for releasing prefix sums of a stream (see \cite{5}).

We now define variations of differential privacy. There are
two common ways of defining neighboring databases,
and each will result in a slightly different definition of
differential privacy. Additionally, there is \( \epsilon \)-differential
privacy, and its relaxation \( (\epsilon, \delta) \)-differential privacy.

\textbf{Definition 2.1} (Neighbors, bounded). Two datasets,
\( D_1 \) and \( D_2 \) are neighbors, if they differ in the value of single
entry. That is, \( \exists D, D_1 = D \cup \{ x \} \) and \( D_2 = D \cup \{ y \} \).

Note that in the bounded case all datasets have the same
number of tuples; i.e., \( \forall D, D \in \mathcal{T}_n \).

\textbf{Definition 2.2} (Neighbors, unbounded). Two datasets,
\( D_1 \) and \( D_2 \) are neighbors if they differ in the presence of
a single entry. That is \( D_1 = D_2 \cup \{ x \} \) or \( D_2 = D_1 \cup \{ x \} \).

\textbf{Definition 2.3} (\( (\epsilon, \delta) \)-Differential Privacy). A mechanism
\( \mathcal{M} \) satisfies \( \epsilon \)-differential privacy if for all outputs \( S \subseteq \text{range}(\mathcal{M}) \),
and for all neighbors \( D_1 \) and \( D_2 \),

\[ \Pr[\mathcal{M}(D_1) \in S] \leq e^\epsilon \cdot \Pr[\mathcal{M}(D_2) \in S] \]

A mechanism satisfies \textbf{bounded} \( \epsilon \)-differential privacy
if we use Defn. 2.1 for neighbors, and \textbf{unbounded}
\( \epsilon \)-differential privacy if we use Defn. 2.2.

A common relaxation of differential privacy is \( (\epsilon, \delta) \)-differential privacy, which allows privacy leakage with a small probability \( \delta \).

\textbf{Definition 2.4} \( (\epsilon, \delta) \)-Differential Privacy. A mechanism
\( \mathcal{M} \) satisfies \( (\epsilon, \delta) \)-differential privacy if for all outputs \( S \subseteq \text{range}(\mathcal{M}) \),
and for all neighbors \( D_1 \) and \( D_2 \),

\[ \Pr[\mathcal{M}(D_1) \in S] \leq e^\epsilon \cdot \Pr[\mathcal{M}(D_2) \in S] + \delta \]

A mechanism satisfies \textbf{bounded} \( (\epsilon, \delta) \)-differential privacy
if we use Defn. 2.1 for neighbors, and \textbf{unbounded}
\( (\epsilon, \delta) \)-differential privacy if we use Defn. 2.2.

We now define the sensitivity of a workload.

\textbf{Definition 2.5}. Let \( N \) denote the set of pairs of neighboring datasets. The \( L_p \) sensitivity of a workload is:

\[ \Delta_{(p, \mathcal{W})} = \max_{(x, x') \in N} \| \mathcal{W}x - \mathcal{W}x' \|_p \]

The definition of the set \( N \) depends on whether we consider bounded or unbounded differential privacy. For unbounded differential privacy,

\[ \Delta_{(p, \mathcal{W})} = \max_{v_i \in \text{cols}(\mathcal{W})} \| v_i \|_p \]

Unless otherwise specified, henceforth we will use the term
differential privacy to mean unbounded differential privacy,
and the term sensitivity to mean sensitivity under unbounded differential privacy.

\begin{align*}
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} & & \begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} & & \begin{bmatrix}
1 & 1 & 1 & 1 \\
1 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1 \\
\end{bmatrix}
\end{align*}

\textbf{I}_k, \textbf{C}_k, \textbf{H}_k

\textbf{Figure 1}: Example workloads: histogram \textbf{I}_k, cumulative
histogram \textbf{C}_k and hierarchical \textbf{H}_k.

\textbf{Example 2.2}. The \( L_1 \) and \( L_2 \) sensitivities of \textbf{I}_k are
both 1. The \( L_1 \) and \( L_2 \) sensitivities of \textbf{C}_k are \( k \) and \( \sqrt{k} \) resp.

We can privately answer linear workloads by adding
independent noise to the true answer of each query. The
noise distribution we use depends on whether we use \( \epsilon \)-
or \( (\epsilon, \delta) \)-differential privacy.

Let \( \text{Normal}(\sigma)^m \) and \( \text{Lap}(\sigma)^m \) be \( m \)-dimensional
vectors of independent samples drawn from the Gaussian and
Laplace distributions resp., with mean 0 and scale \( \sigma \).

\textbf{Definition 2.6}. Let \( \mathcal{W} \) be a workload, and \( x \) be the vector
of true counts for the database. Let \( \epsilon \) and \( \delta \) be parameters.
The Gaussian mechanism \( \mathcal{G}(\mathcal{W}, x) \), is defined as follows:

\[ \mathcal{G}(\mathcal{W}, x) = \mathcal{W}x + \text{Normal}(\sigma)^q \]

where \( \sigma = \Delta_{(2, \mathcal{W})} \sqrt{2 \ln(2/\delta)} / \epsilon \).

\textbf{Definition 2.7}. Let \( \mathcal{W} \) be a workload, and \( x \) be the vector
of true counts for the database. Let \( \epsilon \) be a parameter. The
Laplace mechanism \( \mathcal{L}(\mathcal{W}, x) \), is defined as follows:

\[ \mathcal{L}(\mathcal{W}, x) = \mathcal{W}x + \text{Lap}(\sigma)^q \]

where \( \sigma = \Delta_{(1, \mathcal{W})} / \epsilon \).

It is known \cite{4,8,9} that the Gaussian mechanism
and the Laplace mechanism satisfy \( (\epsilon, \delta) \)-differential privacy
and \( \epsilon \)-differential privacy, respectively. We now define
the error of answering a workload using some mechanism
\( \mathcal{M} \).

\textbf{Definition 2.8}. Let \( q \) be a linear counting query (horizontal row vector), and \( \mathcal{M} \) be a mechanism. Let \( x \) be a vector
of the true counts of the dataset. The mean squared error
of answering query \( q \) on the true counts \( x \) using \( \mathcal{M} \) is

\[ \text{ERROR}_{\mathcal{M}}(q, x) = \mathbb{E}[(q \mathcal{x} - \mathcal{M}(q, x))^2] \]

where \( \mathcal{M}(q, x) \) is the noisy answer of query \( q \). The error
of the workload \( \mathcal{W} \) on the true counts \( x \) is given by

\[ \text{ERROR}_{\mathcal{M}}(\mathcal{W}, x) = \sum_{q \in \text{rows}(\mathcal{W})} \text{ERROR}_{\mathcal{M}}(q, x) \]
Theorem 2.1. \((\text{[3]} \text{[4]} \text{[13]})\) Let \(W\) be a \(q \times k\) workload.

- The mean squared error of answering \(W\) on every dataset \(x\) using the Laplace mechanism is \(2q\Delta^2_{(1,W)}/\epsilon^2\).
- The mean squared error of answering \(W\) on every dataset \(x\) using the Gaussian mechanism is \(q\Delta^2_{(2,W)}\frac{2\log(2/\delta)}{\epsilon^2}\).

Note that the errors for the Laplace and Gaussian methods do not depend on the true counts \(x\). Hence, they are referred to as data oblivious mechanisms \([1]\). In this paper, we will only consider data oblivious mechanisms, and hence we will drop the \(x\) parameter and refer to the error of a workload using \(\text{ERROR}_{\mathcal{M}}(W)\).

### 2.2 Extended Matrix Mechanism

Li et al \([12]\) describe the matrix mechanism framework for optimally answering a workload of linear queries. The key insight is that while some workloads \(W\) have a high sensitivity, they can be answered with low error by answering a different strategy query workload \(A\) such that (a) \(A\) has a low sensitivity \(\Delta_A\), and (b) rows in \(W\) can be reconstructed using a small number of rows in \(A\).

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

\[
\mathcal{M}_A(W, x) = Wx + WA^+Z(\sigma)^p
\]

where, \(Z, \sigma\) are the Laplace distribution and \(2\Delta^2_{(1,A)}/\epsilon\) for \(\epsilon\)-differential privacy, and the Gaussian distribution and \(\Delta_{(2,A)}/\epsilon \cdot \sqrt{2\ln(2/\delta)}\) for \((\epsilon, \delta)\)-differential privacy, respectively.

It is easy to see that all matrix mechanism algorithms are data oblivious. We will use \(\text{ERROR}_{\mathcal{G}, A}(W)\) to denote the error of answering \(W\) using the Gaussian version of the extended matrix mechanism under strategy \(A\). We use \(\text{ERROR}_{\mathcal{L}, A}(W)\) for the Laplace version. The error of these mechanisms can be quantified as follows:

Theorem 2.2. \((\text{[12]} \text{[13]}\) Let \(W\) be a workload. The error of answering \(W\) using the matrix mechanism defined in Equation 1 with strategy \(A\) is

\[
\text{ERROR}_{\mathcal{L}, A}(W) = P(\epsilon)\Delta^2_{(1,A)}\|WA^+\|_F^2
\]

\[
\text{ERROR}_{\mathcal{G}, A}(W) = P(\epsilon, \delta)\Delta^2_{(2,A)}\|WA^+\|_F^2
\]

where \(\|\cdot\|_F\) is the Frobenius norm, \(P(\epsilon) = 2/e^2\), and \(P(\epsilon, \delta) = \frac{2\log(2/\delta)}{\epsilon^2}\).

The Frobenius norm of matrix \(M\), denoted by \(\|M\|_F\), equals \(\sqrt{\text{trace}(M^T M)}\), where \(\text{trace}(M)\) is the sum of the entries that lie on the diagonal of \(M\).

Example 2.3. Answering workload \(C_k\) using Laplace mechanism results in a total error of \(O(k^3/e^2)\). The hierarchical strategy workload \(H_k\) (Figure 1) corresponds to releasing counts on a binary tree over the domain. Using \(H_k\) as the strategy for \(C_k\) can be shown to result in

\[
\text{ERROR}_{\mathcal{L}, H_k}(C_k) = O(k \log^3 k/e^2) \quad \text{[3]} \text{[4]}
\]

We define the minimum error that any strategy \(A\) can achieve for workload \(W\).

Definition 2.9. Let \(W\) be a workload.

\[
\text{MINERROR}_L(W) = \min_{A: WA^+ = W} \text{ERROR}_{\mathcal{L}, A}(W)
\]

\[
\text{MINERROR}_G(W) = \min_{A: WA^+ = W} \text{ERROR}_{\mathcal{G}, A}(W)
\]

### 2.3 Blowfish Privacy

We give definitions for the Blowfish framework \([10]\). An instantiation of the Blowfish framework is a policy graph, which generalizes the notion of neighboring databases from differential privacy. Note that in \([10]\), a policy is slightly more complex. They also define constraints on the set of possible databases, which defines the adversary’s prior knowledge about the database. In this paper, we assume no constraints on the set of possible databases.

Definition 2.10 (Policy Graph). A policy graph is a graph \(G = (V, E)\) with \(V \subseteq T \cup \{\perp\}\), where \(\perp\) is the name of a special vertex.

This graph defines pairs of domain values that an adversary should not be able to distinguish between. If \(\perp \in V\), we add a column to \(W\) to correspond to this “new” domain value, with all values in the column being 0 to ensure that every node in \(V\) is associated with a column in \(W\).

Definition 2.11 (Neighbors, Blowfish). Consider a policy graph \(G = (V, E)\). Let \(D_1\) and \(D_2\) be datasets. \(D_1\) and \(D_2\) are neighbors, denoted \((D_1, D_2) \in N(G)\), if exactly one of the following is true:

- \(D_1\) and \(D_2\) differ in the value of exactly one entry such that \((u, v) \in E\), where \(u\) is the value of the entry in \(D_1\) and \(v\) is the value of the entry in \(D_2\).
• $D_1$ differs from $D_2$ in the presence of exactly one entry, $u$, such that $(u, \bot) \in E$.

$(\epsilon, G)$-Blowfish privacy and $(\epsilon, \delta, G)$-Blowfish privacy are defined by applying the new definition of neighbors from Definition 2.11 to Definitions 2.3 and 2.4 respectively. More formally,

**Definition 2.12** ($(\epsilon, G)$-Blowfish Privacy). *Let $G$ be a policy graph. A mechanism $M$ satisfies $(\epsilon, G)$-Blowfish privacy if for all outputs $S \subseteq \text{range}(M)$, and for all neighboring datasets $(D_1, D_2) \in N(G)$,\n\[
\Pr[M(D_1) \in S] \leq e^\epsilon \cdot \Pr[M(D_2) \in S]
\]

**Definition 2.13** ($(\epsilon, \delta, G)$-Blowfish Privacy). *Let $G$ be a policy graph. A mechanism $M$ satisfies $(\epsilon, \delta, G)$-Blowfish privacy if for all outputs $S \subseteq \text{range}(M)$, and for all neighboring datasets $(D_1, D_2) \in N(G)$,\n\[
\Pr[M(D_1) \in S] \leq e^\epsilon \cdot \Pr[M(D_2) \in S] + \delta
\]

Let $u$ and $v$ be in the same connected component and consider $D_1 = D \cup \{u\}$ and $D_2 = D \cup \{v\}$. Then under $(\epsilon, G)$-Blowfish privacy,
\[
\Pr[M(D_1) \in S] \leq e^{-d(u, v)} \cdot \Pr[M(D_2) \in S]
\]

where $d(u, v)$ is the shortest path between $u$ and $v$ in $G$. However, if $u$ and $v$ are not connected, there is no bound on probabilities; i.e., an adversary is allowed to distinguish between $D_1$ and $D_2$ based on some output. In particular, if $G$ has $c$ connected components $C_1, \ldots, C_c$, $C_i = (V_i, E_i)$, we are allowed to disclose (without any noise) which $V_i$ every tuple in the dataset belongs to.

Therefore, we can split any workload $W$ into smaller workloads $W_1, \ldots, W_c$ that are column projections of the original workload, where $W_i$ only has columns corresponding to $V_i$ (and for workload $W_i$ we consider the policy graph $C_i$). We can answer each of these workloads independently (using the same $\epsilon$, since they pertain to disjoint subsets of the domain), and then add the resulting vectors together to compute the final noisy answer for $W$. Therefore, without loss of generality we assume for the rest of the paper that $G$ is connected.

The above definitions generalize both the bounded and unbounded versions of differential privacy. We have the bounded version of differential privacy with policy graph

$G = (V, E)$ such that $E = \{(u, v) \mid u, v \in T\}$.

We have unbounded differential privacy with policy graph

$G = (V, E)$ such that $E = \{(u, \bot) \mid u \in T\}$.

More generally, any graph that includes $\bot$ will result in databases from $T$ (like unbounded differential privacy), while a graph that does not include $\bot$ results in databases from $T_n$, where $n$ is the size of the domain (like bounded differential privacy).

### 3 Blowfish Matrix Mechanism

Given a Blowfish policy graph $G$ and a workload $W$, the sensitivity of the workload $W$ under policy $G$ can be computed as follows.

**Definition 3.1.** *The $L_p$ policy specific sensitivity of a query matrix $W$ with respect to policy graph $G$ is*

\[
\Delta_{(p, W)}(G) = \max_{(x, x') \in \text{range}(G)} \|Wx - Wx'\|_p
\]

Let $G = (V, E)$ be a policy graph, $k = |V|$ and $n_G = |E|$. We define a $(k \times n_G)$ matrix $P_G$ as follows. We begin with $|V|$ rows, one for each value in the domain, and one for $\bot$ if appropriate; i.e., the rows of $G$ correspond to columns of $W$. For each edge $(u, v) \in E$ add a column to $P_G$ with a 1 in the row corresponding to vertex $u$, and a $-1$ in the row corresponding to vertex $v$ (the order of the 1 and $-1$ is not important) and zeros in the rest of the rows. Since we assume $G$ is connected, every $v \in V$ participates in at least one edge. Hence, no row of $P_G$ will contain all zeros.

For workload $W$ we denote $WP_G$ as $W_G$.

**Lemma 3.1.** *Let $W$ be a workload, and $G$ a policy graph.\n\[
\Delta_{(p, W)}(G) = \max_{v_i \in \text{cols}(W_G)} \|v_i\|_p
\]

All proofs in this section are deferred to the appendix. Notice that we defined $W_G$ in such a way that $\Delta_W = \Delta_{W_G}$. That is, the policy specific sensitivity of $W$ is the same as the standard sensitivity of a new workload $W_G$. We can now define the Blowfish matrix mechanism almost identically to Equation (7) but change the sensitivity to what was specified in Definition 3.1. Analogous to Theorem 2.2 we have:

**Theorem 3.2.** *Consider a workload $W$, and Blowfish policy graph $G$. The error of answering $W$ using the matrix mechanism with strategy $A$ with respect to discriminative secret graph $G$ is:
\[
\text{ERROR}^G_{(\epsilon, A)}(W) = P(\epsilon)\Delta_{(1, A_G)}^2\|WA^+\|_F^2
\]

\[
\text{ERROR}^G_{(\epsilon, \delta, A)}(W) = P(\epsilon, \delta)\Delta_{(2, A_G)}^2\|WA^+\|_F^2
\]

where $P(\epsilon)$ is $2/\epsilon$ and $P(\epsilon, \delta)$ is $\frac{2\log(2/\delta)}{\epsilon^2}$.

We can view the multiplication by $P_G$ as a transformation of the domain. Columns in $W$ correspond to domain values and to vertices of $G$. Columns in $P_G$ correspond to edges in $G$. While a query $q \in W$ associates weights on (a subset of) vertices in $G$, the same query $q$ transformed by $P_G$ associates weights on (a subset of) edges in $G$. Lemma 3.3 describes the relationship between this set of vertices and edges for counting queries.
Lemma 3.3. Let \( q \) be a linear counting query (that is, all entries in \( q \) are either 1 or 0), and \( G = (V, E) \) be a policy graph. Let \( \{v_1, \ldots, v_t\} \subseteq V \) be the vertices corresponding to the nonzero entries of \( q \). Then, the nonzero columns of \( q \cdot P_G = q_G \) correspond to the set of edges \((u, v)\) with exactly one end point in \( \{v_1, \ldots, v_t\} \). That is,

\[
\{(u, v) : |\{u, v\} \cap \{v_1, \ldots, v_t\}| = 1\}.
\]

On this transformed domain, we can directly answer the workload using a transformed database \( x_G = P_{G}^{-1}x \), where \( P_{G}^{-1} \) is the right inverse of \( P_G \). As we will see in Section 3, \( P_G \) has a right inverse for all connected graphs \( G \). Now, the workload \( W_G \) can be answered using this new database, since

\[
W_G \cdot x_G = W \cdot P_G \cdot P_{G}^{-1} \cdot x = Wx,
\]

which is the answer to the original workload. Viewing multiplication by \( P_G \) as a transformation in this way will be helpful in understanding the strategies in Section 5.

Example 3.1. Consider a domain \( T = \{x_1, x_2, \ldots, x_k\} \) and a policy \( G_k^{\text{line}} = (T, E) \), where \( E = \{(x_i, x_{i+1}) \mid \forall i < k\} \) (see Figure 2). This is the line graph policy. Notice that under the line graph policy, the sensitivity of the cumulative histogram workload \( C_k \) is exactly 1 – changing an individual record from \( x_i \) to \( x_{i+1} \) changes exactly one query (namely the count of elements from \( x_{i+1} \) to \( x_k \)) by 1. We can also derive this mathematically. \( M = C_k \times P_{G_k^{\text{line}}} \) is a \((k \times (k - 1))\) matrix, where the first row has all zeros, and the remaining \( k - 1 \) rows form the identity matrix. The standard sensitivity of \( M \) is 1, and thus the policy specific sensitivity of \( C_k \) under \( G_k^{\text{line}} \) is also 1. It is also easy to verify that the policy specific sensitivity of \( C_k \) under \( G_k^{\theta} \) (Fig 2) is \( \theta \).

We will use this to show the following:

Lemma 4.2. Let \( G \) be a Blowfish policy graph and \( W \) be a workload. If \( P_G \) has a right inverse, then \( BA = W \) if and only if \( BA_G = W_G \), where \( A_G = AP_G \). Additionally, both \( WA^+ \) and \( WA_GA^+_G \) are solutions to both \( BA = W \) and \( BA_G = W_G \).

This brings us to our crucial theorem. We use the fact that the solution spaces of \( BA = W \) and \( BA_G = W_G \) are the same in order to show that the error achieved by using strategy \( A \) for workload \( W \) with respect to a policy graph \( G \) is the same as the error achieved by using strategy \( A_G \) for \( W_G \) under differential privacy. This will allow us to find upper bounds under both \((\epsilon, \delta, G)\)-Blowfish and \((\epsilon, G)\)-Blowfish privacy. It will also allow us to directly develop lower bounds for Blowfish analogous to the SVDBound for differential privacy [13]. First we give some notation:

Definition 4.1. Let \( W \) be a workload, and \( G \) be a policy graph.

\[
\text{MINERROR}_G(W) = \min_{A:WA^+A=W} \text{ERROR}_G(W)
\]

\[
\text{MINERROR}_G(W) = \min_{A:WA^+A=W} \text{ERROR}_G(W)
\]

Theorem 4.3. Let \( G \) be a Blowfish policy graph. If \( P_G \) has a right inverse, then we have \( \|WA^+\|_F = \|W_GA_G^+\|_F \). Therefore,

\[
\text{ERROR}_G(W) = \text{ERROR}_G(W_G)
\]

Additionally, minimum errors are equivalent. That is,

\[
\text{MINERROR}_G(W_G) = \text{MINERROR}_G(W)
\]

\[
\text{MINERROR}_G(W_G) = \text{MINERROR}_G(W)
\]

The right inverse requirement seems quite restrictive at first:

Lemma 4.4. Let \( M \) be an \( m \times n \) matrix. \( M \) has a right inverse if and only if its rows are linearly independent.

In other words, \( P_G \) must have at least as many columns as it has rows, and must be full rank. It is easy to check that this is not true of \( P_G \) for most graphs \( G \). For instance, \( P_G^{\text{line}} \) (Fig 2) has only \( k - 1 \) columns and \( k \) rows. Fortunately, for every connected \( G \), we can slightly modify the workload \( W \) to \( W' \) and \( P_G \) to \( P'_G \) such that (i) the minimum error for answering \( W' \) under \( P'_G \) is the same as the minimum error for \( W \) under \( P_G \), and (ii) \( P'_G \) is full rank and thus has a right inverse.

To begin, suppose \( W \) has at least one column with all zeros. We can safely eliminate those columns from \( W \) and the corresponding rows from \( P_G \) (recall that columns in
W and rows in P_G correspond to values in T). These changes do not affect the sensitivity of W_G, since these changes only change W by removing an all zeros column, and any good strategy for answering W_G will also have zeros in those columns. Thus we can consider these modified matrices without affecting any of our results. We next show: (a) the resulting P_G' is full rank for every connected graph, and (b) every workload W can be converted to an equivalent workload W' when considering databases in T. We state the former as a lemma, and explain the latter thus showing that our results apply to all connected graphs.

**Lemma 4.5.** Let G = (V,E) be a Blowfish policy graph and assume G is connected. Removing any row of P_G results in a full rank matrix.

Recall that we assume G is a connected graph. Let W be a workload, and assume that W has at least one column with all zeros. Then we can delete that column and the corresponding row of P_G without affecting W_G (we are simply removing a zero column of W_G, and these can be ignored anyways). The modified version of P_G is full rank, and therefore has a right inverse. To show that the workload has at least one column with all zeros, first consider the case where ⊥ is in the graph. We must add an all zeros column of to W that corresponds with ⊥, so W already has a zero column.

If ⊥ is not in G, recall that the databases must come from T; that is the size of the database n is known. The size of the database can be cast as a linear query Q_n = (1,1, . . . ,1). Any linear query Q = (q_1,q_2, . . . ,q_k) can be answered if we know the answer to Q = Q - q_1 · Q_n = (0,q_2 - q_1, . . . ,q_k - q_1). Moreover, the error in answering Q is the same as the error in answering Q since they differ in a scalar (q_1 · n).

Thus, given a query workload W, pick some v ∈ T. Let V be the workload W[v;] × Q_n, where W[v;] is the column in the workload corresponding to v and Q_n is a column vector of all ones. It is easy to verify that W' = W - V has all zeros in the column corresponding to v.

**Example 4.1.** In C_k, the first row is Q_n. Since we already know n, we don’t need to answer that query privately. We can equivalently consider a workload C'_k with all zeros in the first row and removing the first column (since it would have all zeros). Consider the line graph G_{k, ecc}^{'}. Removing the first row from P_{G_{k, ecc}} would result in a (k-1) × (k-1) matrix that is full rank (and actually the inverse of C'_k).

Thus, by Theorem 4.3, the minimum error for answering C_k under Blowfish policy G_{k, ecc}^{' linear} is equal to the minimum error for answering C'_k · P_{G_{k, ecc}}^{' linear} = I_{k-1} under differential privacy. Since I_{k-1} is the identity workload, the optimal strategy is to add Laplace or Gaussian noise to each query to yield a total error of Θ(k/ε²).

| Workload | Error per query | ε-Diff. [18] |
|----------|-----------------|--------------|
| R_k | G_k^1 | Θ(1/ε²) | O(1/ε²) |
| R_k | G_k^θ | O(log³d/ε²) | O(log³k/ε²) |

Figure 3: Summary of results. This work answers R_k under G_k^1 and G_k^θ using a new, extendable framework with the same error as [10]. Additionally, we give efficient strategies to answer multidimensional range queries.

5 Upper Bounds

In this section, we derive near optimal strategies under the extended matrix mechanism framework for answering workloads under (ε, G)-Blowfish privacy for different policies. In Section 5.1 we define the types of queries and graphs we will be focusing on. In Section 5.2 we describe our general approach to finding strategies, and the tools and techniques that we use. In Section 5.3 we present strategies for answering one dimensional range queries under various graphs. In Section 5.4 we present strategies for answering multidimensional range queries under various graphs. Figure 3 summarizes our upper bounds.

5.1 Workloads and Policy Graphs

Consider a multidimensional domain 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 x ∈ R^k^d with each entry x_i denoting the true count of a value i ∈ T. It is important to note that our results in this paper can be easily extended to the case when dimensions have different sizes.

We focus on range queries. A multidimensional range query can be represented as a d-dimensional hypercube with the bottom left corner l and the top right corner r. In particular, when d = 1, a range query q(l, r) is a linear counting query which counts the values within l and r in the database x, i.e., q(l, r) · x = ∑_{l ≤ i ≤ r} x_i. Let R_k denote the workload of all such one dimensional range queries, i.e., R_k = {q(l, r) | l, r ∈ [k] ∧ l ≤ r}. Similarly, let R_k^d = {q(l, r) | l, r ∈ [k]^d ∧ l ≤ 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 R_k^d can be represented as a q × k^d matrix, where q = (k(k-1)/2)^d is the total number of range queries.

The class of policy graphs G_{k, ecc}^d we consider here are based on the L_1 distance in the domain. Consider two vertices u = (u_1, . . . , u_d) and v = (v_1, . . . , v_d) ∈ [k]^d, the L_1 distance between is |u - v| = |u_1 - v_1| + · · · + |u_d - v_d|. In general, G_{k, ecc}^d is a graph with vertex set [k]^d, and (u, v)
is an edge in \( G^\theta_k \) if and only if \(|u - v| \leq \theta\). We will sometimes refer to \( G^1_k \), or \( G^1_k \), as a line graph.

- Find some strategy \( A_G \) to answer \( W_G \) with low error under differential privacy.
- Use \( A = A_G P_G^{-1} \) to answer \( W \).

Based on Theorem 4.3 we can show that if \( A_G \) can answer \( W_G \) with near optimal error under differential privacy, then \( A = A_G P_G^{-1} \) is a near optimal strategy for \( W \) under Blowfish policy \( G \).

**Corollary 5.1.** Let \( c \geq 1 \) be some real number. Let \( G \) be a Blowfish policy graph, \( W \) be a linear workload and \( A \) be a strategy for answering the workload. Let \( W_G = W P_G \) and \( A_G = A P_G \). Then, \( \text{ERROR}^G_{(Z,A)}(W) \leq c \cdot \text{MINERROR}^G_{(Z)}(W) \) if and only if \( \text{ERROR}^G_{(Z,A)}(W_G) \leq c \cdot \text{MINERROR}^G_{(Z)}(W_G) \), for both \( Z = G \) and \( Z = L \).

An important special case of the above corollary (which we will use later) is that if we know an optimal strategy \( A_G \) (or \( c = 1 \) in the above Corollary) for answering \( W_G \) under differential privacy, the strategy \( A_G P_G^{-1} \) is an optimal strategy for answering \( W \) under the Blowfish policy graph \( G \). Theorem 4.3 and Corollary 5.1 allow us to leverage the rich literature on the matrix mechanism for differential privacy to design efficient mechanisms for answering workloads under Blowfish.

We would like to point out that the error equivalence in Theorem 4.3 and Corollary 5.1 applies both to the total error as well as the error per query, since the number of queries in \( W \) and \( W_G \) are the same.

### 5.2 Techniques

To find strategies for workloads \( R_k \) and \( R_{k+1} \), we will use two main techniques. The first (Section 5.2.1) applies and extends the idea of transformation equivalence, developed in Section 4. The basic idea is that a workload \( W \) under Blowfish privacy policy \( G \) can be transformed into a workload \( W_G \) under differential privacy. Then the existing matrix mechanism for answering \( W_G \) under differential privacy can be applied, and the strategy can be converted back to answer \( W \) under Blowfish privacy. However, the matrix mechanism is inefficient and \( W_G \) will potentially be much larger than \( W \) (especially when \( G \) is dense). So we need the second technique (Section 5.2.2), which says that, instead of working with \( G \), we can find a sparser (sub)graph \( G' \) of the policy graph \( G \) if it preserves the distances well and work with \( G' \) to design mechanisms. These two techniques are orthogonal and can be applied together to design efficient mechanisms. These ideas are depicted in Figures 4 and 5.

#### 5.2.1 Transformational Equivalence

Theorem 4.3 shows that the error for workload \( W \) using strategy \( A \) under policy graph \( G \) is equal to the error for \( W_G = W P_G \) using strategy \( A_G \) under both bounded and unbounded differential privacy. Hence, we can adopt the following general method:

- Given \( W \), convert it to \( W_G = W P_G \).

- Find some strategy \( A_G \) to answer \( W_G \) with low error under differential privacy.

- Use \( A = A_G P_G^{-1} \) to answer \( W \).

**Lemma 5.2.** (Subgraph Approximation) Let \( G = (V, E) \) be a policy graph. Let \( G' = (V', E') \) be a subgraph 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 \( \ell \) \((G') \) is said to be an \( \ell \)-approximation subgraph \( G' \). Then for any mechanism \( M \) which satisfies \((\epsilon, G')\)-Blowfish privacy, \( M \) also satisfies \((\ell \cdot \epsilon, G)\)-Blowfish privacy.

We illustrate all these tools in the following sections. We focus on \((\epsilon, G)\)-Blowfish privacy under policies \( G^\theta_k \) (unless otherwise specified). Analogous upper bounds can be derived for \((\epsilon, \delta, G)\)-Blowfish by using Gaussian noise; and we defer details to a full version of the paper.

---

1While 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 the rest of this paper.
5.3 One dimensional range queries

In this section we present strategies for answering one-dimensional range queries under $G_k^d$. The material in this section and the next are aided by figures that appear in the Appendix [3].

5.3.1 $R_k$ under $G_k^1$

We begin with a simple case: one-dimensional range queries under a one-dimensional line graph. We can answer these queries with constant error under Blowfish.

Theorem 5.3. Workload $R_k$ can be answered with $\Theta(1/\epsilon^2)$ error per query under $(\epsilon, G_k^1)$-Blowfish privacy.

Proof. Consider any range query $q(l, r)$. This is a vector of 0s and 1s, with 1s appearing in columns corresponding to values in the range $[l, r]$. Recall from Lemma 3.3 that the transformed query $q_G(l, r) = q(l, r) \cdot P_G$ associates 1s to only those edges $(u, v)$ in $G$ such that only one of $u$ or $v$ have a 1 in $q(l, r)$. When $G$ is the line graph, this corresponds to the edges at the ends of the range, namely $(l - 1, l)$ and $(r, r + 1)$. This is illustrated in Figure 7 [13]. Therefore, any $q_{G_k^1}(l, r)$ consists of at most two 1s in any row (and the rest are 0). We can answer this workload of queries $(R_{G_k^1})$ using the identity matrix $I_{k-1}$ as our strategy. Every $q_{G_k^1}(l, r) \in R_{G_k^1}$ can be reconstructed by summing at most two queries in the strategy matrix. Each query in $I_{k-1}$ can be answered with $\Theta(1/\epsilon^2)$ error using the Laplace mechanism. So each $q_{G_k^1}(l, r)$ incurs only $\Theta(1/\epsilon^2)$ error. That is, we can answer $R_{G_k^1} = R_k \cdot P_{G_k^1}$ with $\Theta(1/\epsilon^2)$ error per query using $I_{k-1}$ as a strategy under $\epsilon$-differential privacy. By Theorem 4.3, we can answer $R_k$ under $(\epsilon, G_k^1)$-Blowfish privacy with $\Theta(1/\epsilon^2)$ error per query using $I_{k-1} \cdot \mathbf{P}^{-1}_{G_k^1}$ as the strategy.

The best known strategy (with minimum error) for answering $R_k$ under $\epsilon$-differential privacy is the Privelet strategy [13] with a much larger asymptotic error of $O(\log^3 k/\epsilon^2)$ per query.

5.3.2 $R_k$ under $G_k^d$

We next explore one dimensional range queries under a more complex policy, $G_k^d$. These results generalize the results from the previous section. In this section, we rely heavily on subgraph approximation (Lemma 5.2).

We first describe how to obtain a subgraph $H_k^d$ from $G_k^d$. We designate $k/\theta$ vertices at intervals of $\theta$; call these “red” vertices. In $H_k^d$, 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; i.e., vertices $\{1, 2, \ldots, \theta - 1\}$ are connected only to vertex $\theta$, vertices $\{\theta + 1, \theta + 2, \ldots, 2 \theta - 1\}$ are connected only to vertex $2 \theta$, and so on. Figure 8a shows $G_k^1$, and Figure 8b shows $H_k^3$. Note that like $G_k^1$, $H_k^d$ is also a tree with $k - 1$ edges. We order the edges in $H_k^d$ according to their left endpoints.

Theorem 5.4. Workload $R_k$ can be answered with $O(\log^3 \theta/\epsilon^2)$ error per query under $(\epsilon, G_k^1)$-Blowfish privacy.

Proof. Note that any pair of adjacent vertices in $G_k^1$ are connected in $H_k^d$ by a constant length path (of $\leq 3$). Therefore, by Lemma 5.2, it is enough to show that each query in $R_k$ can be answered with $O(\log^3 \theta/\epsilon^2)$ error under $(\epsilon, H_k^d \cdot \text{Blowfish})$-Blowfish privacy (and the error under $G_k^1$ will only be off by a constant factor).

Consider some query in $R_k$, say $q(l, r)$. The corresponding query in $R_{H_k^d}$ consists of all edges which satisfy Lemma 3.3. 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 the edges that satisfy Lemma 3.3 correspond to $\{(i, x\theta) \mid (x-1)\theta \leq i < l\}$ and $\{(j, y\theta) \mid (y-1)\theta \leq j \leq r\}$. Figures S8a-S8c illustrates the proof up to this point.

Note that the transformed query $q_{H_k^d}(l, r)$ corresponds to the union of two range queries (according to the ordering of edges in $H_k^d$). Moreover, each range query is of length at most $\theta$ – within $[(x-1)\theta, x\theta)$ for some $x$. Thus we can answer all the queries in $R_{H_k^d} = R_k \cdot P_{H_k^d}$ by (a) using $k/\theta$ instantiations of Privelet [13] to answer all range queries of length at most $\theta$ within $[(x-1)\theta, x\theta)$ for all $x$, and (b) reconstructing queries $q_{H_k^d}(l, r) \in R_{H_k^d}$ by adding up the corresponding range queries output by Step (a). Since the $k/\theta$ instantiations of Privelet are on disjoint subsets of the domain, they all can use the same $\epsilon$ privacy budget. Thus, each range query within $[(x-1)\theta, x\theta)$ incurs only $O(\log^3 \theta/\epsilon^2)$ error. Therefore, each query in $R_{H_k^d}$ incurs at most $O(\log^3 \theta/\epsilon^2)$ error. By Theorem 4.3, this is also the error of $R_k$ under $(\epsilon, H_k^d \cdot \text{Blowfish})$.

5.4 Multidimensional range queries

We now give strategies for answering range queries in higher dimensions under Blowfish.

5.4.1 $R_k$ under $G_k^d$

In this case, $G_k^d$ is a grid with $k^d$ vertices and $2d \cdot k^d$ edges.

Theorem 5.5. Workload $R_k$ can be answered with $O(d \log^{3(d-1)} k/\epsilon^2)$ error per query under $(\epsilon, G_k^d)$-Blowfish privacy.

Proof. For some range query, the corresponding query in $R_{G_k^d} = R_k \cdot P_{G_k^d}$ will essentially be the bounding box of the $d$ dimensional query hyperrectangle. This is illustrated in two dimensions in Figure 9a. Each face of the hyperrectangle will produce a range of edges in the transformed query. The transformed query is therefore the sum
2d ranges, each of the ranges in d − 1 dimensions. We can see in Figure 9b that the transformed query will be made up of four one-dimensional ranges. If our original range query were in three dimensions, the transformed query would be made up of six two-dimensional ranges.

Our goal is to answer all (d − 1)-dimensional ranges of edges under differential privacy. For each face of the hyperrectangle, the corresponding range of edges consists only of edges orthogonal to the face. In each range, all edges are parallel. Fix one dimension and consider all edges parallel to this dimension. There will be \( k - 1 \) (d − 1)-dimensional layers of these edges. Our strategy is to answer all range queries over each of these layers. Because the layers are disjoint, each set of range queries can be answered in parallel without dividing the \( \epsilon \)-budget. Moreover, the sets of edges for each fixed dimension we consider are disjoint, since each set contains edges orthogonal to all edges in every other set. We illustrate this strategy in two dimensions in Figure 9c.

How much error will we incur answering all these range queries? 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 the Privelet framework \[\text{IS}\] with \( O\left(\frac{\log^{3(d-1)} k}{\epsilon^2}\right) \) error. To answer our query, we must sum 2d of these ranges for a total error of \( O(d \log^{3(d-1)} k / \epsilon^2) \).

By Theorem 4.3 we can answer \( R_{k,d} \) under \( G_{k,d}^0 \) with the same error per query.

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

### 5.4.2 \( R_{k,d} \) under \( G_{k,d}^0 \)

We now turn our attention to multidimensional range queries under \( G_{k,d}^0 \). Our strategy will be similar to the one in Section 5.3.2. We find a subgraph and map edges to vertices. We show the queries of the transformed workload can be decomposed into range queries of bounded size, and our strategy matrix consists of these range queries. The results of this section apply to general dimension \( d \), but throughout the section we will use \( d = 2 \) as an example in proofs and figures.

We first describe how to obtain subgraph \( H_{k,d}^0 \) from \( G_{k,d}^0 \). Although we provide a short explanation here, this is most easily understood by studying Figure 10a and Figure 10b. We divide \( G_{k,d}^0 \) into d-dimensional hypercubes with edge length \( \theta / d \). 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. For each non-red vertex, we remove all edges except the one connecting to this selected red vertex (for vertices that are on the boundary of cubes, and therefore fall in multiple cubes, we pick a consistent way of mapping them). The red vertices are then connected in a grid so that each red vertex is connected to the other 2d nearest red vertices.

We divide up edges into two categories. The first category of edges, which we call external edges, are edges whose endpoints are both red (and form a grid like \( G_{k,d}^1 \)). Internal edges are edges with only one red endpoint.

#### Theorem 5.6

**Workload** \( R_{k,d} \) can be answered with

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

error per query under \((\epsilon, G_{k,d}^0)\)-Blowfish privacy.

**Proof.** We first decompose our query into two pieces: all internal edges, and all external edges. We find strategies to answer each of these queries, then sum the two to find the answer to the desired query. Figure 10c shows the set of external edges in the transformed query. External edges always form a lattice, so we can answer this part of the query using the strategy from Section 5.4.1, and this will contribute \( O(d \log^{3(d-1)} \cdot k/d \theta) \) error.

We also need a way to answer all the internal edges. We order these edges by their black endpoint. Consider the set of vertices, \( V \) corresponding to the set of internal edges which satisfy Lemma 3.3 \( V \) can be divided into \( 2d \) \( d\)-dimensional range queries, one for each face of the original range query. These \( d\)-dimensional range queries are bounded by \( \theta \) in the dimension orthogonal to the corresponding face of the original range query. This is illustrated in two dimensions in Figure 10d. Our strategy to answer these bounded ranges is the following: For each dimension, divide the domain (which is a hypercube of size \( k^d \)) into \( \frac{k}{\theta} \) layers, each with thickness \( \theta / d \). We then answer all range queries on each layer. For a given dimension, all layers are independent. Therefore, we can answer these sets of range queries in parallel. However, the sets of range queries for different dimensions are not independent. An edge used in some horizontal layer will also be used in some vertical layer. We can answer each set of range queries using the Privelet framework with error

\[
O\left(\frac{\log^{3(d-1)} k \log^3 \theta / d}{\epsilon^2}\right).
\]

However, because range queries in different dimensions are dependent, we must divide up our \( \epsilon \)-budget \( d \) ways. Additionally, each query is made up of \( 2d \) of these range queries. The total error of this strategy is therefore

\[
O(d^3 \cdot \frac{\log^{3(d-1)} k \log^3 \theta / d}{\epsilon^2}).
\]

The total error is the sum of the errors from the strategies of answering the internal edges and the external edges. This sum is just Equation 5.4.2.
6 Error Lower Bounds under Blowfish

In this section we present a lower bound on the minimum error needed to answer a workload under the extended matrix mechanism framework with respect to \((\epsilon, \delta, G)\)-Blowfish privacy (Section 6.1), then compare this lower bound to the \((\epsilon, \delta)\)-differential privacy lower bound for 1- and 2-dimensional range queries under different Blowfish policy graphs (Sections 6.2 and 6.3).

6.1 General Lower Bound

The main result of Li and Miklau [13] is that the minimum error is related to the singular value decomposition of the workload matrix.

**Theorem 6.1** ([13]). Let \(W\) be an \(m \times n\) workload.

\[
\text{MINERROR}_G(W) \geq P(\epsilon, \delta) \frac{1}{n} (\lambda_1 + \ldots + \lambda_s)^2
\]

where \(P(\epsilon, \delta) = \frac{2 \log(2/\delta)}{\epsilon^2}\) and \(\lambda_1, \ldots, \lambda_s\) are the singular values of \(W\).

Our lower bound for Blowfish privacy follows immediately by combining Theorems 6.1 and 4.3.

**Corollary 6.2.** Let \(G\) be a Blowfish policy graph, and let \(W\) be a workload. If \(P_G\) has a right inverse,

\[
\text{MINERROR}_G(W) \geq P(\epsilon, \delta) \frac{1}{n_G} (\lambda_1 + \ldots + \lambda_s)^2
\]

where \(P(\epsilon, \delta) = \frac{2 \log(2/\delta)}{\epsilon^2}\), \(\lambda_1, \ldots, \lambda_s\) are the singular values of \(W_G\), and \(n_G\) is the number of columns of \(W_G\) (same as the number of edges in \(G\)).

**Remarks:** Note that \(s\) is the number of singular values of \(W_G\), and therefore if \(W_G\) is \(s \times n_G\), \(s = \min(q, n_G)\). This lower bound applies for all connected graphs since, by Lemma 4.3 \(P_G\) has a right inverse for all connected graphs. Moreover, since Blowfish with the complete graph results in bounded differential privacy, Corollary 6.2 also gives us a lower bound for bounded differential privacy, whereas Theorem 6.1 applies only to unbounded differential privacy. Finally, since \((\epsilon, \delta, G)\)-Blowfish privacy is a relaxation of \((\epsilon, G)\)-Blowfish privacy, these results provide lower bounds for \(\epsilon\)-Blowfish privacy as well.

6.2 Lower Bounds for \(R_k\)

We analytically derive an asymptotic lower bound for 1-dimensional range queries \(R_k\) under the line graph \(G_{k}^1\).

**Theorem 6.3.**

\[
\text{MINERROR}_{G_{k}^1}(R_k) = \Theta(k^2/\epsilon^2)
\]

Proof. (sketch) Recall from Example ?? that the workload \(C_k\) on domain \([1,k]\) is defined as the set of range queries \(\{q(i,j) \mid 1 \leq i \leq j \leq k\}\). We show a lower bound of \(\Omega(k^2/\epsilon^2)\) for \(R_k\) under Blowfish policy \(G_{k}^1\) in 3 steps,

- Partition the set of range queries \(R_k\) into a set of cumulative histogram queries \(C_k \cup C_{k-1} \cup \ldots \cup C_1\), each operating on a subset of the domain.
- Any strategy for answering \(R_k\) under \(G_{k}^1\) incurs no less error than the sum of the errors incurred for the optimal Blowfish strategy for answering each \(C_i\), \(i \in [k]\) under \(G_{i}^1\) on the appropriate domain.
- \(\text{MINERROR}_{G_{k}^1}(C_k) = \Omega(k^2/\epsilon^2)\).

For the first step, if the domain \(T = [k]\), then \(R_k\) is the set of queries \(\{q(i,j) \mid 1 \leq i \leq j \leq k\}\). This can be partitioned into disjoint sets of queries \(S_i = \{q(i,j) \mid \forall j \text{ s.t. } i \leq j \leq k\}\). \(S_i\) is identical to the \(C_{k-i+1}\) workload on the domain \(i, i+1, \ldots, k\) by Corollary 6.2.

Next, note that \(G_{k}^1\) restricted to the subdomain \(i, i+1, \ldots, k\) corresponds also to the line graph \(G_{k-i+1}^1\). Thus, it is enough to lower bound the sum of the minimum errors for each \(C_i\) under \(G_{i}^1\), for \(i \in [k]\).

By Corollary 6.2, the lower bound on the minimum error for \(C_k\) depends on the singular values of \(C_k \cdot P_{G_{k}^1}\), which is equal to the identity matrix \(I_{k-1}\) (see Example 4.1 on page 7). Thus, we have

\[
\text{MINERROR}_{G_{k}^1}(C_k) = \Omega(k^2/\epsilon^2)
\]

which completes the lower bound proof.

The lower bound is asymptotically tight since we know from Theorem 5.3 that \(R_k\) can be answered with error \(O(k^2/\epsilon^2)\) under \((\epsilon, G_k^1)\)-Blowfish privacy.

6.3 Lower Bounds for \(R_{k^4}\)

We now illustrate the lower bounds (from Corollary 6.2) for 1-dimensional \((R_k)\) and 2-dimensional \((R_{k^2})\) range queries satisfying \((\epsilon, \delta, G_{k}^0)\)-Blowfish privacy and \((\epsilon, \delta, G_{k^2}^0)\)-Blowfish privacy respectively.

Figures 6a and 6b illustrate the relationship between the lower bound on error and size of the domain for \(R_k\) and \(R_{k^2}\) respectively. We plot the original lower bound for unbounded differential privacy (from [13]) and the new lower bounds we derived for Blowfish policies \(G_{k}^0\) and \(G_{k^2}^0\) for various values of \(\theta\). Additionally, we plot a lower bound for bounded differential privacy, which is obtained by using the complete graph (on \(T\)) as the policy graph.

For the one dimensional range query workload we see that minimum error under unbounded differential privacy increases faster than the minimum error under \(G_{k}^0\) for sufficiently large domain sizes. For two dimensional ranges,
error under Blowfish policy $G_{\theta}^{k,2}$ is only better than unbounded differential privacy for $\theta = 1$. However, all values of $\theta$ perform better than bounded differential privacy. Note that for sets of linear queries, it is possible for the sensitivity of a workload under bounded differential privacy to be twice the sensitivity of the workload under unbounded differential privacy, and thus have up to 4 times more error. Characterizing analytical lower bounds for these workloads and policies is an interesting avenue for future work.

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, where the transformation only depends on the policy graph. This, in conjunction with a subgraph approximation result, helped us derive lower and upper bounds for 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.

References

[1] A. Bhaskara, D. Dadush, R. Krishnaswamy, and K. Talwar. Unconditional differentially private mechanisms for linear queries. In Proceedings of the 44th symposium on Theory of Computing, pages 1269–1284. ACM, 2012.

[2] C. Dwork. Differential privacy. In ICALP, 2006.

[3] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In EUROCRYPT, pages 486–503, 2006.

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

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

[6] M. Fredrikson, E. Lantz, S. Jha, S. Lin, D. Page, and T. Ristenpart. Privacy in pharmacogenetics: An end-to-end case study of personalized warfarin dosing. SEC’14, pages 17–32, Berkeley, CA, USA, 2014. USENIX Association.

[7] A. Ghosh, T. Roughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. In STOC, pages 351–360, 2009.

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

[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. In SIGMOD, 2014.

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

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

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

[14] A. Machanavajjhala, A. Korolova, and A. D. Sarma. Personalized social recommendations - accurate or private? In PVLDB, volume 4, pages 440–450, 2011.
[15] A. Nikolov, K. Talwar, and L. Zhang. The geometry of differential privacy: the sparse and approximate cases. In Proceedings of the 45th annual ACM symposium on Symposium on theory of computing, pages 351–360. ACM, 2013.

[16] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In STOC, pages 75–84, 2007.

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

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

Lemma 3.1. Let $W$ be a workload, and $G$ a policy graph.

$$
\Delta_{(p,W)}(G) = \max_{v_i \in \text{cols}(W_G)} \|v_i\|_p
$$

Proof. 

$$
\Delta_{(p,W)}(G) = \max_{(x,x') \in N(G)} \|Wx - Wx'\|_p
$$

By the definition of neighbors, $x$ and $x'$ differ in two counts (an entry has been switched from one domain value to another). Let these be domain values be $i$ and $j$, and let $x_i, x_j, x'_i, x'_j$ be their counts. So, $x_i = x'_i + 1$ and $x_j = x'_j - 1$. So, $Wx - Wx' = x - x'$. Additionally, by the definition of neighbors, $(i, j)$ must be an edge in $G$. Therefore, column there is a column in $P_G$ which has a 1 and -1 at $i$ and $j$. So, $x - x'$ is a column of $WP_G = W_G$. So,

$$
\Delta_{(p,W)}(G) = \max_{v_i \in \text{cols}(W_G)} \|v_i\|_p
$$

Theorem 3.2. Consider a workload $W$, and Blowfish policy graph $G$. The error of answering $W$ using the matrix mechanism with strategy $A$ with respect to discriminative secret graph $G$ is:

$$
\text{ERROR}_{(G,A)}^G(W) = P(\epsilon)\Delta_A^2(1, A_G)\|WA^+\|^2_F \\
\text{ERROR}_{(G,A)}^G(W) = P(\epsilon, \delta)\Delta_A^2(2, A_G)\|WA^+\|^2_F
$$

Lemma 3.3. Let $q$ be a linear counting query (that is, all entries in $q$ 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 $q$. Then, the nonzero columns of $q \cdot P_G = qG$ 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 $qG$ satisfies $c = u - v$ where $u, v$ are entries in $q$ and $(u,v) \in E$. $c$ is nonzero exactly when $u \neq v$, or equivalently, when

$$
\{u, v\} \cap \{v_1, \ldots, v_\ell\} = 1
$$

Lemma 4.2. Let $G$ be a Blowfish policy graph and $W$ be a workload. If $P_G$ has a right inverse, then $BA = W$ if and only if $BA = W_G$. Where $A_G = AP_G$. Additionally, both $WA^+$ and $W_GA^+_G$ are solutions to both $BA = W$ and $BA = W_G$.

Proof. First, assume $BA = W$. Then

$$
BA = W \implies BA = W_G
$$

Next, assume $BA = W_G$. Then

$$
BA = W_G \implies BA = W
$$

Additionally, from Lemma 4.1 we have that $WA^+$ is a solution to $BA = W$, and $W_GA^+_G$ is a solution to $BA = W_G$. Because these equations have the same solution space, $WA^+$ is a solution to $BA = W_G$ and $W_GA^+_G$ is a solution to $BA = W_G$.

Theorem 4.3. Let $G$ be a Blowfish policy graph. If $P_G$ has a right inverse, then we have $\|WA^+\|_F = \|W_GA^+_G\|_F$. Therefore,

$$
\text{ERROR}_{(G,A)}^G(W) = \text{ERROR}_{(\epsilon, \delta)}(G, A_G)(W)
$$

Additionally, minimum errors are equivalent. That is,

$$
\text{MINERROR}_{\epsilon, \delta}(G, A_G)(W) = \text{MINERROR}_{\epsilon, \delta}(G, A_G)(W)
$$

Proof. We show the proof for the error under the Gaussian noise (i.e., $G$). The proof for the errors under the Laplace noise (i.e., $L$) is similar.

By Lemma 4.2 and we have that both $WA^+$ and $W_GA^+_G$ are solutions to the system $BA = W$. Additionally, by Lemma 4.1 $\|WA^+\|_F \leq \|B\|_F$ and $\|W_GA^+_G\|_F \leq \|B\|_F$ for all solutions $B$. Therefore, $\|WA^+\|_F = \|W_GA^+_G\|_F$. Then by definition of error we have

$$
\text{ERROR}_{(G,A)}^G(W) = \text{ERROR}_{(G,A_G)}(W_G)
$$

Additionally, assume that $A = A_*$ minimizes $\text{ERROR}_{(G,A)}^G(W)$ subject to $WA^+ = W$. Then by Lemma 4.2 $W_GA^+_G = A_G = W_G$, and so

$$
\text{MINERROR}_{\epsilon, \delta}(G,A_G)(W) = \text{MINERROR}_{\epsilon, \delta}(G,A_G)(W)
$$

We can prove the converse similarly, that is

$$
\text{MINERROR}_{\epsilon, \delta}(G,A_G)(W) \geq \text{MINERROR}_{\epsilon, \delta}(W)
$$

And therefore we have

$$
\text{MINERROR}_{\epsilon, \delta}(G,A_G)(W) = \text{MINERROR}_{\epsilon, \delta}(W)
$$
Lemma 4.4. Let $M$ be an $m \times n$ matrix. $M$ has a right inverse if and only if its rows are linearly independent.

Proof. This is just a concise way of stating the following facts:
- If $m > n$, then $M$ cannot have a right inverse.
- If $m = n$, then $M$ has both a left and right inverse if and only if its determinant is nonzero, which is true if and only if the matrix is full rank.
- If $m < n$, then $M$ has a right inverse if and only if it is full rank.

Lemma 4.5. Let $G = (V, E)$ be a Blowfish policy graph and assume $G$ is connected. Removing any row of $P_G$ results in a full rank matrix.

Proof. We first show that any connected graph $G$ produces an $P_G$ of rank $m - 1$ where $m$ is the number of rows of $P_G$. We then show that our modification of $P_G$ does not change the rank. We are then left with an $P_G$ with $m - 1$ rows and rank $m - 1$. So, the modified $P_G$ will be full rank.

Every connected graph $G$ has a spanning tree $T$. Note that $P_T$ is a column projection of $P_G$, so to show $P_G$ has rank $m - 1$, it is sufficient to show that $P_T$ has rank $m - 1$. Every tree has some node $v$ of degree 1. The corresponding row of $P_T$ has all zeros except for a single 1 or −1. Let $e_v$ be the row corresponding to $v$. Let the $k$th column be the one corresponding to the single nonzero element of $e_v$. Other than row $v$, there is unique row $u$ which has a nonzero value in column $k$. Let $r_u$ be the row $u$ with a zero in column $v$ and identical in all other values.

Consider $P_{T-v}$. This matrix will be different from $P_T$ in the following ways: $P_{T-v}$ will be missing column $k$ and row $v$. Additionally, the new value of row $u$ will be $r_u$. Note that row $u$ from $P_T$ can be written as $r_u - e_v$. All rows in $P_T$ other than row $u$ are linearly independent of $e_v$. Therefore, every row in $P_T$ can be written as a linear combination of a row in $P_{T-v}$, and possibly $e_u$. This means that $\text{rank}(P_T) = 1 + \text{rank}(P_{T-v})$. $P_{T-v}$ is also a tree, so we proceed inductively. The base case of this induction is a tree with only one edge, and this corresponds to a matrix of rank 1. Therefore, $P_G$ for a connected graph $G$ has rank $m - 1$, where $m$ is the number of vertices in $G$ or equivalently the number of rows of $P_G$.

Note that the $m - 1$ rows we remove during the induction are all linearly independent, and the row corresponding the final vertex left over can be written as a linear combination of these $m - 1$ rows. However, when we initially picked vertex $v$ of degree 1, we had two choices for $v$, since any tree has two vertices of degree 1. Since we have two choices at each inductive step, it is possible to pick any vertex we wish to end up with on the last step of the induction. That is, not only is does $P_G$ have rank $m - 1$, every set of $m - 1$ rows of $P_G$ is linearly independent.

Next, we have assumed that some column of $W$ is all zeros. This means that we may remove the corresponding row of $P_G$ without changing $W_G$. Because every set of $m - 1$ rows of $P_G$ is linearly independent, removing one row leaves us with $m - 1$ rows, all of which are linearly independent. Therefore, the modified $P_G$ has full rank, as desired.

Corollary 6.2. Let $G$ be a Blowfish policy graph, and let $W$ be a workload. If $P_G$ has a right inverse,

$$\text{MINERROR}_G(W_G) \geq P(\epsilon, \delta) \frac{1}{n_G}(\lambda_1 + \ldots + \lambda_s)^2$$

where $P(\epsilon, \delta) = \frac{2 \log(2/\delta)}{e^2}$, $\lambda_1, \ldots, \lambda_s$ are the singular values of $W_G$, and $n_G$ is the number of columns of $W_G$ (same as the number of edges in $G$).

Proof. From the results of [13] we have

$$\text{MINERROR}_G(W_G) \geq P(\epsilon, \delta) \frac{1}{n_G}(\lambda_1 + \ldots + \lambda_s)^2$$

where $P(\epsilon, \delta) = \frac{2 \log(2/\delta)}{e^2}$, $\lambda_1, \ldots, \lambda_s$ are the singular values of $W_G$, and $n_G$ is the number of columns of $W_G$. But then by Theorem 4.3 we know that $\text{MINERROR}_G(W_G) = \text{MINERROR}_G(Z)$ which completes the proof.

Corollary 5.1. Let $c \geq 1$ be some real number. Let $G$ be a Blowfish policy graph, $W$ be a linear workload and $A$ be a strategy for answering the workload. Let $W_G = WP_G$ and $A_G = AP_G$. Then, $\text{ERROR}(W_G, A_G) \leq c \cdot \text{MINERROR}_Z(W_G)$ if and only if $\text{ERROR}(Z, A_G)(W_G) \leq c \cdot \text{MINERROR}_Z(W_G)$, for both $Z = G$ and $Z = L$.

Proof. This follows immediately from the following two facts from Theorem 4.3 for both $Z = L$ and $Z = G$: $\text{ERROR}(Z, A)(W_G) = \text{ERROR}(Z, A_G)(W_G)$ $\text{MINERROR}_Z(W_G) = \text{MINERROR}_Z(W)$

Lemma 5.2. (Subgraph Approximation) Let $G = (V, E)$ be a policy graph. Let $G' = (V, E')$ be a subgraph 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 $\ell$. $G'$ is said to be an $\ell$-approximation subgraph of $G$. Then for any mechanism $M$ which satisfies $(\epsilon, G')$-Blowfish privacy, $M$ also satisfies $(\ell \cdot \epsilon, G)$-Blowfish privacy.

\footnote{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 the rest of this paper.}


Proof. Assume \(D\) and \(D'\) are neighboring databases under policy graph \(G\). Then \(D = A \cup \{x\}\) and \(D' = A \cup \{y\}\) for some database \(A\), and \((x, y) \in E\). From our assumption, \(x\) and \(y\) are connected by a path in \(G'\) of length at most \(\ell\). Therefore, there exist a sequence of vertices \(x = v_1, \ldots, v_j = y\) such that \((v_i, v_{i+1}) \in E\) and \(j < \ell\). Further, \(A \cup \{v_i\}\) and \(A \cup \{v_{i+1}\}\) are neighbors under policy graph \(G'\). Therefore, we have

\[
\Pr[M(A \cup \{v_i\}) \in S] \leq e^{-\ell} \cdot \Pr[M(A \cup \{v_{i+1}\}) \in S].
\]

Composing over all \(1 \leq i \leq j\) gives

\[
\Pr[M(A \cup \{x\}) \in S] \leq e^{-\ell} \cdot \Pr[M(A \cup \{y\}) \in S],
\]

as desired. \(\square\)
B  Figures

Figure 7: A one dimensional range query on vertices is transformed into a query on edges. The only edges present in the transformed query are the ones at the end of the range. These edges are highlighted in purple.

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

(b) $H^3_{10}$, for each vertex, we remove all adjacent edges except the one connecting to nearest red vertex to the right. Note that for all $\theta$, a pair of adjacent vertices in $G^\theta_k$ are connected by a path of length at most 3 in $H^\theta_k$.

(c) A range query shown on $H^3_{10}$. The transformed query consists of the edges satisfying Lemma 3.3 and these are highlighted in purple. These edges are ordered by their left endpoints, highlighted with dotted outlines, which 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, and therefore the range queries on each set can be answered in parallel.

Figure 8: A summary of a strategy for answering $R^k$ under $G^\theta_k$. 
$G_{k^2}^1$, a two dimensional line graph.

$G_{k^2}^1$ with a two dimensional range query, represented by a grey box. The edges in the new query (those that satisfy Lemma 3.3), are highlighted in purple. These edges form four ranges: two horizontal ranges of vertical edges, and two vertical ranges of horizontal edges.

For each row of vertical edges, we answer all ranges over the row. One such row is highlighted in purple. We must do the same for columns, and one such column is highlighted in green.

Figure 9: Answering $R_{k^2}$ under $G_{k^2}^1$. 
(a) $G_{2}^{2}$, each vertex is connected to other vertices within $L_{1}$ distance 2 on the grid.

(b) A section of $H_{2}^{k}$. Internal edges are light blue and external edges are black.

(c) A 2D range query superimposed on the graph. Instead of showing all vertices, we show the divisions in $\theta/2$ blocks. 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 3.3 and therefore appear in the transformed query.

(d) The shaded rectangles show the sets of vertices corresponding to the internal edges which satisfy Lemma 3.3. There are 4 such rectangles, and for each one either the height or length is bounded by $\theta$. Note that there are other ways in which we could divide the shaded region into 4 rectangles, we arbitrarily chose one. Our strategy is to answer all range queries over each row squares, and each column of squares.

Figure 10: Transforming queries in $R_{k^{2}}$ under $G_{k^{2}}^{\theta}$.