ProBE: Proportioning Privacy Budget for Complex Exploratory Decision Support

Nada Lahjouji
nlahjouj@uci.edu
University of California, Irvine
Irvine, USA

Sameera Ghayyur
sghayyur@snap.com
Snap Inc.
Los Angeles, USA

Xi He
xi.he@uwaterloo.ca
University of Waterloo
Waterloo, Canada

Sharad Mehrotra
sharad@ics.uci.edu
University of California, Irvine
Irvine, USA

ABSTRACT

This paper studies privacy in the context of complex decision support queries composed of multiple conditions on different aggregate statistics combined using disjunction and conjunction operators. Utility requirements for such queries necessitate the need for private mechanisms that guarantee a bound on the false negative and false positive errors. This paper formally defines complex decision support queries and their accuracy requirements, and provides algorithms that proportion the existing budget to optimally minimize privacy loss while supporting a bounded guarantee on the accuracy. Our experimental results on multiple real-life datasets show that our algorithms successfully maintain such utility guarantees, while also minimizing privacy loss.

1 INTRODUCTION

We consider the privacy-preserving execution of complex aggregate queries over d-dimensional data. Consider, for instance, a dataset containing medical records of patients and their respective diseases with the following schema: *PATIENT_DATA(patient_name, age, gender, disease, disease_type)*. An analytical query of interest over such data, listed below in SQL, identifies prevalent viral diseases that afflict vulnerable populations such as the elderly (age over 65) or children (age below 5).

```
SELECT disease, count(*) FROM PATIENT_DATA
WHERE disease_type = 'viral'
GROUP BY disease
HAVING (count(*) > c1 AND avg(age) > 65)
OR (count(*) > c2 AND avg(age) < 5)
```

Such queries often arise in decision support (DS) applications [17, 25, 47, 50] as part of online analytical processing (OLAP) [5]. OLAP plays a crucial role in exploring data to produce valuable insight that facilitates informed decision-making. For instance, businesses and organizations utilize decision support to evaluate KPIs [9] (Key Performance Indicators), metrics which gauge progress towards an intended goal by computing metrics based on aggregate statistics. Multiple KPIs are often used in tandem to evaluate the performance of businesses, e.g., sales volume and retention rate to infer growth. Such KPIs are instances of complex aggregate queries composed of multiple conditions comparing different aggregate statistics to their respective thresholds, combined using AND/OR operators. Other use cases for such complex queries include clinical decision support applications [44, 46, 50] which use complex queries to diagnose and classify diseases (e.g. the previously defined query), building management systems [14, 53] that ensure building code compliance by comparing aggregate statistics to policy thresholds, and supply management systems [7, 35] which optimize operations by analyzing statistics as they relate to existing benchmarks or user-defined criteria.

It follows that data sources used by decision support applications often contain sensitive information about individuals, and releasing aggregated statistics from such sources can lead to severe privacy leaks [8, 20]. Differential privacy [18, 19] is a popular and effective notion that provides a formal guarantee on privacy by hiding individual records while releasing aggregate statistics, but this is done at the expense of accuracy by adding noise to the data. Traditionally, privacy-preserving query answering uses a "privacy-first" architecture that provides formally defined privacy guarantees while maximizing the utility of data given [21, 26]. Recent work [23, 24, 34, 43] has argued the advantages of a dual "utility-first" approach instead, wherein a desired level of utility is specified and privacy maximized given this requirement. This approach is far more suitable in decision support setting, as it not only provides a guarantee on the query answer and therefore confidence in the decision made based on it, but it also offers the opportunity for higher privacy provided that the utility requirements are met at a lower level of invasiveness.

In the context of decision support, utility requirements consist of more than one metric. In particular, DS queries answered by a differentially private mechanism, result in two types of errors: false positives (FP) and false negatives (FN). These errors are the basis of statistical hypothesis testing [15], a widely established method used in multiple fields [3, 42, 45] which determines the validity of a hypothesis based on sample data. The testing results in either a Type I error (FP) or Type II error (FN) which are subsequently compared to pre-set bounds to make a decision about said hypothesis. DS can be considered a direct application of hypothesis testing, as it uses queries based on statistical methods to make an informed decision. It is thus critical for a differentially-private DS framework to enforce utility bounds on both false positive and negative errors in order to guarantee the validity of its query results.

Prior work [23, 24] has studied the utility-first approach but their scope is limited, especially in the context of DS queries. Firstly, they

only focus on simple queries which do not have multiple conditions comparing different aggregate statistics to their respective thresholds, i.e. can only answer a query with a single condition in the HAVING clause (e.g. $count(*) > c1$). Secondly, they do not properly tackle the dual utility requirements of DS (i.e., FP and FN). APEx [23], which does not differentiate between the two error types (viz., FP and FNs) considering them both as errors, offers error bounds only for data point that are far from the threshold specified in queries. Errors (i.e., misclassification of points as FP and FNs) in an uncertainty region close to the threshold, remain unbounded. **MIDE** [24], overcomes this limitation of APEx and offers formal utility bounds irrespective of where data points lie, but it only provides bounds on false negatives. Given a set bound on FN, it uses a heuristic approach to explore the trade-off between privacy loss and the FP error. It does so by *weakening* the classifying threshold in such a way that a desired FN bound can be reached at a significantly lower privacy loss while incurring a small penalty on the FP. Moreover, both APEx and MIDE consider simple queries.

In this paper, we study a comprehensive approach to solving the problem of answering complex DS queries in a differentially private manner so as to offer dual utility bounds on both FP and FN while minimizing privacy loss. To address the complexity of the query, the intuition behind our approach is to decompose the original query into multiple simple queries with single aggregate functions (i.e. with known query sensitivities) which can be answered by differentially private mechanisms. Such mechanisms can then be composed by using the respective AND/OR operators used in the original query to link the aggregate functions. The main challenge, then, lies in finding a methodology to similarly decompose or proportion the overall utility bounds across the simple queries (i.e. how to assign allowed error bounds per sub-query to meet the overall accuracy requirement of the final query), as well as the privacy budget, to subsequently guarantee these bounds in a way that optimally minimizes privacy loss. One possible solution could be to divide the utility budgets equally across the multiple conditions, but such an approach, as we will see, is sub-optimal, seeing as different aggregate threshold queries may require a higher error tolerance depending on the selected data distribution.

Alternatively, we propose **ProBE** (**Pro**portioning Privacy **B**udget in Complex **E**xploratory Decision Support Queries), a framework that optimally partitions the privacy/utility budget across the individual simple queries such that the required utility bounds are guaranteed at the lowest overall privacy loss possible. We postulate this as a multi-criteria optimization problem which aims to minimize privacy loss given the constraints enforced by the desired FN/FP bounds. This approach faces two main challenges: it firstly requires quantifying the trade-off between FN and FP errors as well as the trade-off between privacy loss itself and the two errors; secondly, it requires the formulation of the utility bounds as well as the privacy loss as differentiable functions in terms of variables derived from the simple queries. We successfully address these challenges in our approach in the context of the disjunction and conjunction of such queries, and solve a multi-variate optimization problem which yields an apportionment technique for such bounds/budgets. We then propose new algorithms which implement this apportionment framework by adapting previously proposed mechanisms in a way that answers complex DS queries with the appropriate utility guarantees. We subsequently discuss the additional complexities and address our approach to solving them. Our main contributions are as follows:

- We formally define privacy-preserving complex decision support queries and their accuracy requirements.
- We postulate proportioning the privacy budget for complex decision support queries as a multi-criteria optimization problem and solve it using the method of Lagrange Multipliers.
- We propose algorithms that build upon and modify previous mechanisms to implement our budget proportioning technique to support complex decision support queries.
- We evaluate our approach against real-world datasets in different domains and show the efficacy of our approach.

The organization of this paper is as follows: Section 2 provides background on differential privacy. Section 3 defines complex decision support queries and their accuracy requirements. In Section 4, we propose our ProBE technique to optimally apportion the privacy budget for such queries. Section 5 implements ProBE through two algorithms and proposes additional optimizations. Section 6 evaluates our algorithms on multiple real datasets using complex queries. Lastly, we discuss related work in Section 7 and future work directions in Section 8.

2 BACKGROUND

We use existing differential privacy concepts as a basis for our work. Given an input dataset $D \in \mathcal{D}$, an algorithm satisfies differential privacy [19] if its output does not significantly change when adding or removing a single tuple in D. Formally:

Definition 2.1 (ϵ-Differential Privacy (DP)). A randomized mechanism $M : \mathcal{D} \rightarrow \mathcal{O}$ satisfies ϵ-differential privacy if

$$ln\frac{P[M(D) = O]}{P[M(D') = O]} \leq \epsilon(O) \tag{1}$$

for any set of outputs $O \subseteq \mathcal{O}$, and any pair of neighboring databases D,D' such that $|D \backslash D' \cup D' \backslash D| = 1$. The privacy metric ϵ represents the privacy budget. A higher ϵ value implies higher privacy loss, whereas a lower ϵ implies strong privacy guarantees.

Differential privacy offers important properties [19, 31] that allow for composability of multiple DP mechanisms and assessment of their privacy loss.

Theorem 2.2 (Sequential Composition). *Let $M_1, ..., M_k$ be k algorithms that satisfy ϵ_i-differential privacy. The sequence of $M_1, ..., M_k$ provides $\sum_{i=1}^{k} \epsilon_i$-differential privacy.*

When a randomized algorithm runs a ϵ-differentially private algorithm repeatedly until a stopping condition is met, it does not satisfy ϵ-differential privacy because the number of iterations is not known prior to its execution. Its overall privacy loss, however, can be determined after the output is returned. A metric used for these algorithms is *ex-post differential privacy* [33]. Formally,

Definition 2.3 (Ex-Post Differential Privacy). Let $\mathcal{E} : \mathcal{O} \rightarrow (\mathbb{R}_{\geq 0} \cup \{\infty\})$ be a function on the outcome space of mechanism $M : \mathcal{D} \rightarrow O$. Given an outcome $O = M(D)$, M satisfies $\mathcal{E}(O)$-ex-post differential privacy if for all $O \in \mathcal{O}$,

$$\max_{D,D':D\sim D'} ln\frac{P[M(D)=O]}{P[M(D')=O]} \leq \mathcal{E}(O) \quad (2)$$

for any set of outputs $O \subseteq \mathcal{O}$, and any pair of neighboring databases D,D' such that $|D\backslash D' \cup D'\backslash D| = 1$. Ex-post differentially private mechanisms also benefit from composability properties. Specifically, the Sequential Composition theorem (Def. 2.2) holds for ex-post DP mechanisms as well [49], i.e. the sequence of ex-post DP mechanisms results in a differentially private mechanism with privacy loss equal to the sum of ex-post privacy losses ϵ_i.

The Laplace mechanism [19] is a widely used differentially private algorithm that achieves ϵ-differential privacy by adding noise drawn from the Laplace distribution. This noise is also calibrated to the *sensitivity* of the query.

Definition 2.4 (Sensitivity). The sensitivity of a function $g : \mathcal{D} \rightarrow \mathbb{R}^d$, denoted Δg, is defined as the maximum L_1 distance between all pairs of neighboring databases D and D' differing in at most one element.

$$\Delta g = \max_{\forall D,D'} \parallel g(D) - g(D') \parallel_1 \quad (3)$$

The sensitivity of a function highly depends on the aggregate statistic queried. For instance, the sensitivity of a counting query is 1.

Theorem 2.5 (Laplace Mechanism (LM)). *Given a function $f : \mathcal{D} \rightarrow \mathbb{R}^d$, the Laplace Mechanism that outputs $f(D) + \eta$ is ϵ-differentially private, where η is a d-length vector of independent samples drawn from a Laplace distribution with the probability density function $p(x|\lambda) = \frac{1}{2\lambda}e^{-|x|/\lambda}$ where $\lambda = \Delta f/\epsilon$.*

3 PROBLEM DEFINITION

In this section, we first formalize the decision support queries and their utility requirements, and then present the problem which answers these queries with DP and utility guarantees.

3.1 Query and Utility Definitions

A DS query consists of a set of *aggregate threshold queries* described below. These atomic aggregate threshold queries are connected through logical operators (i.e., $\cup$ and $\cap$), which we refer to as the conjunction or disjunction of multiple such queries. Specifically, an aggregate threshold query returns the set of objects in a dataset, deferred to as predicates, whose aggregate values exceed the set thresholds. We present the formal definitions as follows.

Aggregate Threshold Query. An aggregate threshold query, denoted by $Q^{\Lambda,f}_{\mathrm{g}(.)>C}$, consists of the following: (i) an aggregate function g(.), which includes any function whose sensitivity can be computed (e.g. AVG or COUNT); (ii) the set of predicates $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_k\}$ which represent the objects that the query iterates over to check for condition satisfaction (e.g. the set of diseases in the previous query example); (iii) a set of corresponding thresholds $C = \{c_1, c_2, ..., c_k\}$ for each predicate, and (iv) an optional filter f which can be any selection condition on any column of the record. We use D^f to denote all the tuples that satisfy the filter. Each predicate λ_i takes in a tuple from filtered tuples D^f and outputs *True* or *False* based on its value. We let $D^f_{\lambda_i}$ be the set of tuples in D^f that evaluate λ_i to be *True*. This query returns all the predicates that have an aggregate over their satisfying tuples $g(D^f_{\lambda_i})$ greater than their respective threshold c_i, i.e.,

$$Q^{\Lambda,f}_{\mathrm{g}(.)>C}(D) = \{\lambda_i \in \Lambda \mid \mathrm{g}(D^f_{\lambda_i}) > c_i\} \quad (4)$$

This is similar to a group-by-having query in SQL. Given a patient dataset with schema *PATIENT_DATA*(*patient_name*, *age*, *gender*, *disease*, *disease_type* , the following is an example of an aggregate threshold query:

```
SELECT disease FROM PATIENT_DATA
WHERE disease_type = 'viral'
GROUP BY disease HAVING count(*) > c
```

The WHERE clause *disease_type = 'viral'* is an example of a filter f. The set $disease = \{d_i, i \in [1, k]\}$ is an example of a set of k predicates Λ. $count()$ is the aggregate function $g(.)$ and c is the threshold, which is the same for all the predicates. An aggregate threshold can also include $\mathrm{g}(D^f_{\lambda_i}) < c_i$ type inequalities by simply modeling such a condition as the negation of Eq. (4) and thus returning the negation of its results. In the context of complex DS queries, we refer to a single aggregate threshold query as an *atomic* query Q_{a_i}, which is the basic, irreducible form a complex DS query can take.

Complex Decision Support Queries. We consider a complex DS query $Q^{\Lambda,\mathcal{F}}$ to be a set of atomic aggregate threshold queries $Q_{a_1}, ..., Q_{a_n}$ connected by the AND/OR operators ($\cap, \cup$), where these n atomic queries share the same set of predicates Λ, but have different filters $\mathcal{F} = \{f_1, ..., f_n\}$, aggregate functions $\mathcal{G} = \{g_1, ..., g_n\}$ and thresholds $C = \{c_1, ..., c_n\}$.

Essentially, we deconstruct a query composed of multiple aggregate functions compared to their respective thresholds into separate, atomic aggregate threshold queries with a single function-threshold pair. Figure 1 shows an example of the decomposition process for the complex DS query previously introduced. The atomic queries contain single conditions comparing an aggregate function g_i to its respective threshold c_i, and their results are subsequently composed back together using the AND/OR operators defined in the original query. To formally define this concept, we make use of a Context-Free Grammar (CFG) as shown below.

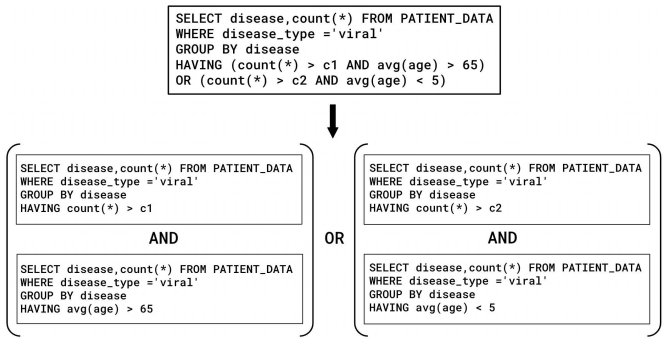


Figure 1: Complex decision support query decomposed into four atomic queries with single HAVING conditions but similar predicates connected by AND/OR operators.

Definition 3.1 (Complex DS Query CFG). Consider the grammar $G = (N, \Sigma, P, S)$ where the complex DS query is a non-terminal

symbol $N = \{Q^{\Lambda,\mathcal{F}}\}$, the atomic aggregate threshold query is a terminal symbol $\Sigma = \{Q_a\}$, and $S = Q^{\Lambda,\mathcal{F}}$ is the starting symbol. The production rules P are:

$$Q^{\Lambda,\mathcal{F}} \rightarrow Q_a \tag{5a}$$

$$Q^{\Lambda,\mathcal{F}} \rightarrow Q^{\Lambda,\mathcal{F}_{i_1}} \cap Q^{\Lambda,\mathcal{F}_{i_2}} \tag{5b}$$

$$Q^{\Lambda,\mathcal{F}} \rightarrow Q^{\Lambda,\mathcal{F}_{i_1}} \cup Q^{\Lambda,\mathcal{F}_{i_2}} \tag{5c}$$

Using this grammar, we can recursively compose complex decision queries with any combination of AND/OR operators. Specifically, by using production rule 5a and 5b we produce a complex query composed of atomic queries connected only by the AND operator like $Q^{\Lambda,\mathcal{F}} = Q_1 \cap Q_2 \cap, \ldots, \cap Q_n$, which we refer to as a **conjunction query**. Similarly, by using production rules 5a and 5c we derive a query composed of OR operators only $Q^{\Lambda,\mathcal{F}} = Q_1 \cup Q_2 \cup, \ldots, \cup Q_n$, which we refer to as a **disjunction query**.

Utility Measures. Decision support applications, as mentioned previously, require setting a bound on the false negative and false positive errors while minimizing privacy loss. We define these bounds formally below.

Definition 3.2 (Bound on the False Negative Rate (FNR)/ False Positive Rate(FPR)). Let $M : \mathcal{D} \rightarrow O$ be a randomized mechanism that answers a complex decision support query $Q^{\Lambda,\mathcal{F}}$ composed of n atomic aggregate threshold queries $Q_1, Q_2, \ldots, Q_n$ with the same predicates Λ and different filters F. We say M satisfies (i) a β-bound on the FNR if for any database $D \in \mathcal{D}$, for all predicates $\lambda_i \in \Lambda$, the following holds:

$$P[\lambda_i \notin M(D) \wedge \lambda_i \in Q^{\Lambda,\mathcal{F}}(\mathcal{D} = D)] \leq \beta \tag{6}$$

(ii) a α-bound on the FPR if for any database $D \in \mathcal{D}$, for all predicates $\lambda_i \in \Lambda$, the following holds:

$$P[\lambda_i \in M(D) \wedge \lambda_i \notin Q^{\Lambda,\mathcal{F}}(\mathcal{D} = D)] \leq \alpha \tag{7}$$

In other words, Eq. (6) represents a bound β on the FNR, i.e. the probability that a predicate λ_i is not in the result of mechanism M given it is in the result of query $Q^{\Lambda,\mathcal{F}}$ for all predicates. Similarly, Eq. (7) represents a bound α on the FPR, i.e. the probability that a predicate λ_i given it is in the result of M but not in the result of $Q^{\Lambda,\mathcal{F}}$ for all predicates. We specify $\mathcal{D} = D$ as the probability of a predicate being in the result of the query considers the data distribution, but for the rest of the paper we use D for simplicity.

Note that in our framework, the β and α accuracy bounds are user-set parameters and depend largely on the nature of the overarching decision support application; for instance, a medical diagnosis application may choose to emphasize the FNR over the FPR if the absence of disease detection is more crucial than an erroneous detection. This model is motivated by similar models used in approximate query processing [6] where users specify confidence intervals, or statistical inference which is used to control false negative and/or positive errors [29]. Additionally, we choose to focus on FNR/FPR over other accuracy measures due to the nature of the queries we tackle, which return a set of objects rather than aggregate values. For instance, a different metric such as variance error may yield a high error value for a specific query whereas the returned set remains unchanged, thus not precisely reflecting the accuracy of the results.

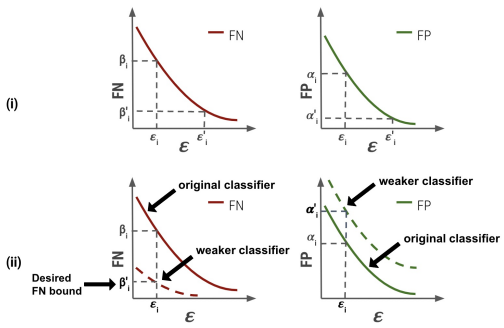


Figure 2: Trade-off between false negatives FN, false positives FP and the privacy budget ϵ in (i) with APEx and (ii) with MIDE.

3.2 ProBE Problem Definition

Given a complex decision support query $Q^{\Lambda,\mathcal{F}}$ on a dataset D composed of atomic queries $Q_{a_1}, ..., Q_{a_n}$ in the structure of query tree T, we want to develop a differentially private mechanism $M_{\text{PROBE}}(T)$ that answers the overall query such that privacy loss ϵ is minimized subject to a β-bound on FNR and α-bound on FPR.

Primitives for Atomic Queries and Limitations. We show two mechanisms from prior work to illustrate their utility guarantees for a simple atomic aggregate threshold query $Q^{\Lambda,f}_{g(\cdot)>C}$ and then present the problem formulation that builds on top of these mechanisms for complex decision support queries.

The first mechanism is the Laplace mechanism used by APEx [23] to answer atomic queries, but *it fails to offer either bounded FPR or bounded FNR.* It adds noise to the aggregate value per predicate and compares the result to its corresponding threshold c_i given a β bound on the overall error rate and a region u around the threshold in which the error is unbounded as inputs (i.e. error tolerance region). However, this mechanism does not offer the β-bound on FNR guarantee for predicates which have aggregates too close to their thresholds, i.e. their aggregates are in the region $[c_i - u, c_i + u]$. We refer to u as the *uncertain region*, which factors into determining the privacy budget needed to achieve the β guarantee outside of said region by using $\epsilon = \frac{\Delta g \ln(1/(2\beta)}{u}$. It follows that a larger uncertain region u implies smaller privacy loss, at the expense of increased false negatives and positives alike. Conversely, a smaller u would lead to less unbounded errors, but would increase privacy loss as a result. Figure 2(i) shows this relationship between FP/FN and ϵ, where increasing ϵ leads to an equal decrease in FP/FN and vice versa.

The second mechanism, known as Threshold Shift Laplace Mechanism (TSLM) designed by MIDE [24] offers a *one-sided guarantee on either bounded FNR or bounded FPR.* TSLM generalizes the threshold c_i by shifting c_i to $c_i - u$ so that the entire uncertain region (i.e. the unbounded error) resides on the left side of the old threshold, and compares this new threshold to the noisy aggregate. Figure 2 (ii) shows the effect of this approach on the trade-off between FN/FP and ϵ, where shifting the threshold by u results in a weaker classifier which achieves a lower bound on FN at the same ϵ but at a higher FP. By setting the privacy budget $\epsilon = \frac{\Delta g \ln(1/(2\beta)}{u}$ as

defined in [23], this solution guarantees a β-bound on FNR for all predicates.

Threshold Shift Laplace Mechanism (TSLM): Given an atomic aggregate threshold query $Q^{\Lambda,f}_{g(\cdot)>C}$, by setting the privacy budget to $\epsilon = \frac{\Delta g \ln(1/(2\beta)}{u}$ where u is the generalized parameter used to shift threshold C to $C-u$, the Threshold Shift mechanism achieves a β-bound on FNR.

Through the threshold shift mechanism, the uncertain region is shifted to $[c_i - 2u, c_i]$, thus providing a formal bound β on the FNR without incurring additional privacy cost, but doing so at the expense of the FPR due to the unbounded error being entirely made of false positives. Due to the nature of this algorithm, it does not provide any sort of bound on the FPR but rather increases them due to the trade-off between the FNR and FPR resulting from the shift. This is due to the fact that the increase in FPR resulting from the shift depends entirely on the data distribution, hence the FPR cannot be pre-determined before running the mechanism.

Optimization Problem for Complex Queries. Minimizing the privacy loss while bounding both FPR and FNR before running a DP mechanism is difficult to achieve unless data distribution is known ahead. Hence, we formulate a hybrid approach instead, wherein one of the two constraints is fixed in our optimization problem and chosen to generate possible solutions, whereas the second constraint is used post-optimization to algorithmically relax the solution in such a way that ensures its bound is upheld. For the rest of our paper, we consider the constraint on FNR as our optimization problem constraint, and FPR to be our post-optimization constraint, but a mirrored problem (i.e. flipping the constraints) is supported by our approach as well. The choice of constraints relies on the nature of the DS application and its intended use cases.

As each atomic query Q_{a_i} can be individually answered using a randomized mechanism $M_a(\beta_i) : \mathcal{D} \rightarrow O_i \times \mathbb{R}^+$ that takes an FNR bound β_i and outputs a query answer and ex-post privacy loss ϵ_i (e.g. mechanisms presented in [24]), we want to use the outputs of such mechanisms O_I, and privacy budgets ϵ_i as inputs for our mechanism $M_{\text{PROBE}}(T)$. Thus, the aim of ProBE is to generate functions that apportion the overall FNR bound in terms of each atomic query's FNR bound to formulate a minimization problem for privacy loss $\epsilon_{\text{PROBE}} = f_\epsilon(\epsilon_1, .., \epsilon_n)$. Specifically, we want to generate a function $f_\beta(\beta_1, ..., \beta_n)$ which apportions the β bound into each β_i such that $FNR \leq \beta$, and $\epsilon(\epsilon_1, .., \epsilon_n)$ is minimized. We therefore obtain a constrained optimization problem defined as:

$$\begin{aligned} \underset{\epsilon_1..\epsilon_n}{\text{minimize}} \quad & f_\epsilon(\epsilon_1, .., \epsilon_n) \\ \text{subject to} \quad & f_\beta(\beta_1, ..., \beta_n) \leq \beta \end{aligned} \tag{8}$$

We first solve this optimization problem given the single constraint on FNR in Section 4, where we first develop ProBE instantiated with the β-bound on FNR. We then relax our solution by enforcing the post-optimization constraint on FPR in Section 5.

4 PROBE FRAMEWORK

In this section, we first present optimization techniques for complex decision support queries that consist of a single connection operator (conjunction or disjunction) with the purpose of guaranteeing a β-bound on FNR. We then generalize our approach to queries combining both conjunction/disjunction operators for n atomic queries.

4.1 Query Conjunction Mechanism

Consider a conjunction query composed of a conjunction operator that links two atomic aggregate threshold queries $Q = Q_1 \cap Q_2$. Our optimization problem has two differentially private mechanisms, M_1 and M_2 of the same type, that answer Q_1 and Q_2, respectively. One way to combine the independent outputs of M_1 and M_2 for the final answer of Q is to find their intersection $M(D) = M_1(D) \cap M_2(D)$. We define this step as *query conjunction mechanism.*

Definition 4.1 (Query Conjunction Mechanism). Let randomized mechanism $M_i : \mathcal{D} \rightarrow O_i$ with a differential privacy guarantee satisfy a β_i-bound on FNR for aggregate threshold query Q_i. We can answer query Q which is a conjunction of 2 aggregate threshold queries $Q_1 \cap Q_2$ using mechanism M where $M(D) = M_1(D) \cap M_2(D)$.

β-Bound on False Negative Rate. As per Sec. 3.2, we know that a mechanism M_i that answers an individual aggregate threshold query has an associated β_i-bound on FNR. To derive the apportioning function f_β for the FNR for Q in terms of β_1 and β_2, we generate a confusion matrix (Figure 3) by running M_1 and M_2 on Q_1 and Q_2 respectively and classifying their outcomes, as well as classifying their conjunction and disjunction. As seen in Figure 3, the conjunction mechanism $M = M_1 \cap M_2$ results in a false negative in any of the three cases (A,B,C). As M_1 and M_2 are mechanisms of independent randomness, their outcomes are independent from one another given the true query answer, though the atomic queries themselves Q_1 and Q_2 may not be independent. Note that because the three cases are mutually exclusive, we can simply add their probabilities to obtain the overall probability of M resulting in a false negative. Thus, we can deduce the overall FNR as the three lines below. For any predicate $\lambda_j \in \Lambda$,

$$\begin{aligned} & P[\lambda_j \notin M(D) | \lambda_i \in Q^{\Lambda,F}(D)] \\ = \; & P[\lambda_j \in M_1(D) | \lambda_j \in Q_1(D)] \cdot P[\lambda_j \notin M_2(D) | \lambda_j \in Q_2(D)] \\ + \; & P[\lambda_j \notin M_1(D) | \lambda_j \in Q_1(D)] \cdot P[\lambda_j \in M_2(D) | \lambda_j \in Q_2(D)] \\ + \; & P[\lambda_j \notin M_1(D) | \lambda_j \in Q_1(D)] \cdot P[\lambda_j \notin M_2(D) | \lambda_j \in Q_2(D)] \\ = \; & (1 - FNR_1) \cdot FNR_2 + (1 - FNR_2) \cdot FNR_1 + FNR_1 \cdot FNR_2 \quad (9) \\ = \; & FNR_1 + FNR_2 - FNR_1 \cdot FNR_2 \quad (10) \\ \leq \; & FNR_1 + FNR_2 \leq \beta_1 + \beta_2 \end{aligned}$$

The last inequality holds due to the β_i-bound guarantee on FNR_i provided by M_i. Hence, we obtain the function:

$$f_\beta(\beta_1, \beta_2) = \beta_1 + \beta_2 \tag{11}$$

The detailed analysis can be found in Appendix A.1.

Privacy Loss. We apply the sequential composition theorem of differential privacy (Def. 2.2) to compose the privacy loss of the two sub-mechanisms of M. Hence, the budget apportioning function of M is

$$\epsilon = f_\epsilon(\epsilon_1, \epsilon_2) = \epsilon_1 + \epsilon_2 \tag{12}$$

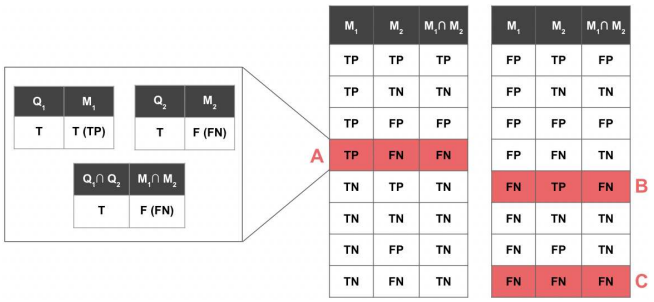


Figure 3: Classification of conjunction of outputs M_1 and M_2 resulting from running the mechanisms on Q_1, Q_2. False negative (FN) outcomes are highlighted in red.

This means that our optimization problem is now as follows.

$$\begin{aligned} &\text{minimize} && \epsilon_1 + \epsilon_2 \\ &\text{subject to} && \beta_1 + \beta_2 \leq \beta \end{aligned} \tag{13}$$

We consider the primitive mechanism for atomic queries, TSLM [24], described in Sec. 3.2 to illustrate the optimization. In this case, we can use the privacy budget $\epsilon_i = \frac{\Delta g_i \ln(1/(2\beta)}{u_i}$ which guarantees a β_i-bound on FNR. Thus, we rewrite the ϵ apportioning function as follows:

$$\begin{aligned} f_\epsilon(\epsilon_1, \epsilon_2) &= \epsilon_1 + \epsilon_2 \\ &= \frac{\Delta g_1 \ln(1/(2\beta_1))}{u_1} + \frac{\Delta g_2 \ln(1/(2\beta_2))}{u_2} \\ &= \ln(1/2\beta_1)^{\frac{\Delta g_1}{u_1}} + \ln(1/2\beta_2)^{\frac{\Delta g_2}{u_2}} \\ &= -\ln((2\beta_1)^{\frac{\Delta g_1}{u_1}} (2\beta_2)^{\frac{\Delta g_2}{u_2}}) \end{aligned} \tag{14}$$

To minimize ϵ, we thus need to maximize

$$(\beta_1)^{\frac{\Delta g_1}{u_1}} (\beta_2)^{\frac{\Delta g_2}{u_2}} \tag{15}$$

Our optimization problem is now as follows. We want to minimize ϵ by maximizing the expression defined in Eq. (15), subject to the β-bound constraint from problem (13). In other words, we aim to find the local maxima of such a multivariate function given the inequality constraint on its variables. This type of constrained optimization problem can thus be solved with the Lagrange Multipliers method [4], as this method aims to determine the extrema of a function composed of multiple variables given an equality or inequality constraint. The Lagrange method achieves this by reformulating the optimization problem into a set function called the Lagrangian function. Solving this function yields the proportioning technique below. A complete proof can be found in Appendix A.1.

Theorem 4.2. *Given a conjunction query $Q = Q_1 \cap Q_2$ answered by a conjunction mechanism $M(D) = M_1(D) \cap M_2(D)$ where M_i implements TSLM and $\epsilon = \epsilon_1 + \epsilon_2$, we achieve minimum privacy loss ϵ by budgeting the β-bound on FNR as:*

$$\beta_1 = \frac{u_2 \Delta g_1 \beta}{u_1 \Delta g_2 + u_2 \Delta g_1}, \beta_2 = \frac{u_1 \Delta g_2 \beta}{u_1 \Delta g_2 + u_2 \Delta g_1} \tag{16}$$

Generalized n-Query Conjunction. We extend the previous approach to generalize it over the conjunction of n-aggregate threshold queries. Consider a query $Q = Q_1 \cap ... \cap Q_n$, where each atomic query Q_i is answered with TSLM M_i that has an associated FNR bound β_i. By decomposing the conjunctions into $(n-1)$ 2-way conjunctions, we can generalize our previous f_ϵ and f_β functions into:

$$\begin{aligned} &\text{minimize} && \Sigma_{i=1}^{n} \epsilon_i \\ &\text{subject to} && \Sigma_{i=1}^{n} \beta_i \leq \beta \end{aligned} \tag{17}$$

Using the Lagrange Multipliers method again, we obtain the apportionment technique below:

Theorem 4.3. *Given a complex DS query $Q^{\Lambda,F}$ composed of n aggregate threshold queries connected by $n-1$ conjunctions, we achieve minimum privacy loss ϵ by budgeting the β-bound on FNR as:*

$$\beta_i = \frac{\Delta g_i \beta \prod_{x=1}^{n, x \neq i} (u_x)}{\sum_{y=1}^{n} \prod_{x=1}^{n, x \neq y} (u_x \Delta g_y)}, \forall i = \{1, 2, ..., n\} \tag{18}$$

Thus, by setting the individual β_i bounds for sub-queries Q_i according to Eq. (18), we provide a mathematical guarantee that privacy loss ϵ is optimally minimized while also maintaining the β-bound guarantee on FNR.

Note that other DP composition theorems could be used to express ϵ as a function of the individual ϵ_i, including the Advanced Composition Theorem [19], but this would change the formulation of our optimization problem. Similarly, other mechanisms may be used to provide DP guarantees presuming that they also offer a β-bound on FNR, but will require different derivations to solve the optimization problem. Using relaxations of DP such as Rényi differential privacy [41] or f-differential privacy [10] is, however, non-trivial, as the expression of privacy loss ϵ in terms of the β or α error rates used (i.e. as defined in [24]) may not necessarily hold. Thus, they require extensive analysis to formulate a baseline expression of privacy loss before solving the optimization problem.

4.2 Query Disjunction Mechanism

Similarly to the previous section, we model a privacy-preserving mechanism on a query composed of two atomic aggregate threshold queries linked by the disjunction operator $Q^{\Lambda,F} = Q_1 \cup Q_2$, where Q_1 and Q_2 are answered by two differentially private mechanisms M_1 and M_2 respectively. The overall mechanism is the union of the two sub-mechanisms $M(D) = M_1(D) \cup M_2(D)$.

Definition 4.4 (Query Disjunction Mechanism). Let mechanism $M_i : \mathcal{D} \rightarrow O_i$ with a differential privacy guarantee satisfy a β_i-bound on FNR for aggregate threshold query Q_i. We can answer query Q which is a disjunction of 2 aggregate threshold queries $Q_1 \cup Q_2$ using mechanism M where $M(D) = M_1(D) \cup M_2(D)$.

β-Bound on False Negative Rate. Similarly to the conjunction mechanism, each randomized mechanism M_i ran on an individual aggregate threshold query in a disjunction has an associated β_i-bound on FNR. We thus derive the apportioning function f_β for the overall false negative rate FNR for Q in terms of β_i by deriving and using a confusion matrix similar to Figure 3 for disjunction. Thus, by similar analysis (shown in Appendix A.1), the overall FNR can be upper bounded by the three lines below. For any predicate

$\lambda_i \in \Lambda$,

$$\begin{aligned} & P[\lambda_j \notin M(D) | \lambda_i \in Q^{\Lambda,F}(D)] \\ \leq\ & P[\lambda_j \notin M_1(D) | \lambda_j \notin Q_1(D)] \cdot P[\lambda_j \notin M_2(D) | \lambda_j \in Q_2(D)] \\ +\ & P[\lambda_j \notin M_1(D) | \lambda_j \in Q_1(D)] \cdot P[\lambda_j \notin M_2(D) | \lambda_j \notin Q_2(D)] \\ +\ & P[\lambda_j \notin M_1(D) | \lambda_j \in Q_1(D)] \cdot P[\lambda_j \notin M_2(D) | \lambda_j \in Q_2(D)] \\ =\ & TNR_1 \cdot FNR_2 + FNR_1 \cdot TNR_2 + FNR_1 FNR_2 \\ =\ & TNR_1 \cdot FNR_2 + FNR_1 (TNR_2 + FNR_2) \\ \leq\ & FNR_2 + FNR_1 \leq \beta_1 + \beta_2 \end{aligned}$$

Again, we simplify this expression by providing an upper bound on the FNR by removing negative clauses. We thus obtain the apportioning function:

$$FNR \leq f_\beta(\beta, \beta) = \beta_1 + \beta_2 \leq \beta \tag{19}$$

Privacy Loss. Similar to conjunction, we use the sequential composition theorem of differential privacy (Def. 2.2) to compose the privacy loss of the two sub-mechanisms of M. Assuming that each sub-mechanism uses TSLM, we can again use the budget $\epsilon_i = \frac{\Delta g_i \ln(1/(2\beta_i))}{u_i}$, which guarantees a β_i-bound on FNR. Thus, overall privacy loss can be minimized by maximizing

$$(\beta_1)^{\frac{\Delta g_1}{u_1}} (\beta_2)^{\frac{\Delta g_2}{u_2}} \tag{20}$$

Since our f_ϵ and f_β functions in the case of disjunctions are identical to that of conjunctions, our optimization problem for both are the same. Using the Lagrange method therefore yields the same apportionment technique. Thus, Theorems 4.2 and 4.3 hold for disjunction as well, which are formally shown in Appendix A.1 and A.2 respectively.

4.3 Combined Conjunctions/Disjunctions

Consider a complex decision support query $Q^{\Lambda,F}$ comprised of a set of n atomic aggregate threshold queries $Q_{a_1}, ..., Q_{a_n}$ connected by disjunctions or conjunctions. Such a query corresponds to a binary operator tree T where each operator is either a conjunction ($\cap$) or a disjunction ($\cup$). To execute $Q^{\Lambda,F}$, we first evaluate each of the atomic queries, then recursively combine their outcomes based on the connecting operators in the operator tree T to determine the results.

Our challenge, therefore, lies in generalizing our apportionment technique such that, given a query tree T, we determine the β budget distribution across all atomic queries such that the β-bound is guaranteed while minimizing privacy loss ϵ. We show through the example below how to express the β as a function of the β_i given a tree structure, then formalize the problem of optimal apportionment given any tree.

Example 4.1. Consider an example query $Q_{T1} = Q_1 \cup (Q_2 \cap Q_3)$ shown in Figure 4(a). We refer to the sub-query associated with a node by its node-id (e.g., in Figure 4(a) node n_1 corresponds to sub-query Q_1.). For Q_{T2}, we can derive the FNR by first deriving individual FNRs for the sub-tree $Q_{n_3} = Q_2 \cup Q_3$ using the 2-conjunction mechanism, where the two sub-trees Q_{n_3} and Q_{n_1} can in turn be executed using the 2-disjunction mechanism. We thus obtain $\beta_{n_3} = \beta_2 + \beta_3$, $\beta_{n_1} = \beta_1$ and $\beta_{n_1} + \beta_{n_3} \leq \beta$. By substitution,

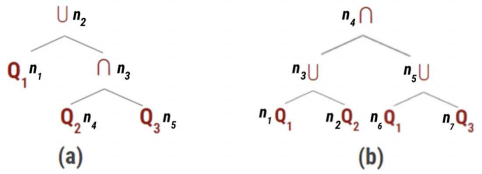


Figure 4: The figure shows the query trees for (a) $Q_{T1} = Q_1 \cup (Q_2 \cap Q_3)$**, and (b)** $Q_{T2} = (Q_1 \cup Q_2) \cap (Q_1 \cup Q_3)$

we obtain the false negative rate constraint

$$\beta_1 + \beta_2 + \beta_3 \leq \beta \tag{21}$$

Let us further consider query $Q_{T2} = (Q_1 \cup Q_2) \cap (Q_1 \cup Q_3)$ shown in Figure 4(b). Executing the same recursive steps for Q_{T2} yields

$$2\beta_1 + \beta_2 + \beta_3 \leq \beta \tag{22}$$

In the example above, note that for Q_{T1}, leaf nodes in $T1$ correspond to unique atomic queries while in Q_{T2}, the atomic query Q_1 appears twice in $T2$. Let us denote the number of occurrences of a atomic query Q_i in the leaf nodes of a tree by o_i. Thus, in $T1$, the values of o_1, o_2, and o_3 are all 1, whereas in $T2$ o_2, and o_3 are 1, while the value of o_1 is 2, leading to the difference in the FNR constraint for $T1$ and $T2$. More specifically, given a tree T with $Q_{a_1}, ..., Q_{a_n}$ sub-queries and corresponding $o_1, ..., o_n$ occurrences, the overall FNR for query $Q^{\Lambda,F}$ can be expressed as

$$f_\beta(\beta_1, \beta_2, ..., \beta_n) = o_1\beta_1 + o_2\beta_2 + ... + o_n\beta_n \leq \beta \tag{23}$$

A formal proof of the above is provided in Appendix A.2. As for privacy loss, we must run the atomic mechanisms M_i on each leaf node in order to maintain their independent randomness. Therefore, the ϵ_i budget allocated for each sub-query must be divided across its occurrences as well, i.e. $o_i\epsilon_i$. Thus, our apportionment problem for a given tree is now

$$\begin{aligned} \text{minimize} \quad & \Sigma_{i=1}^{n} o_i \epsilon_i \\ \text{subject to} \quad & \Sigma_{i=1}^{n} o_i \beta_i \leq \beta \end{aligned} \tag{24}$$

We use the Lagrange Multipliers method to yield the theorem below.

Theorem 4.5. *Given a complex decision support query $Q^{\Lambda,F}$ with query tree T composed of n aggregate threshold queries with an associated o_i number of occurrences within the tree, we achieve minimum privacy loss ϵ by budgeting the β-bound on FNR as:*

$$\beta_i = \frac{\Delta g_i \beta \prod_{x=1}^{n, x \neq i} (u_x)}{\sum_{y=1}^{n} \prod_{x=1}^{n, x \neq y} (u_x o_y \Delta g_y)}, \forall i = \{1, 2, ..., n\} \tag{25}$$

Note that a query $Q^{\Lambda,F}$ may be represented by multiple equivalent trees, due to the distributive property of logical operators. In Example 4.1 for instance, Q_{T1} and Q_{T2} have equivalent query trees $T1$ and $T2$, where $T2$ is obtained from distributing the OR operator over the AND operator. From Eq. (21) and (22), however, we can infer that a distributed query tree, i.e. a query with a higher number of nodes, may cause the overall privacy loss of the query to increase, because it allocates a lower β_i to one or more given sub-queries Q_i.

Thus, the ProBE mechanism aims to generate the optimal query tree with the least number of leaf nodes to minimize any additional distribution of the β budget. We can use Boolean function

Algorithm 1 ProBE Mechanism Overview. DS query $Q^{\Lambda,\mathcal{F}} = [T, \{Q_1(g_1, c_1), \ldots, Q_n(g_n, c_n)\}]$, maximum privacy budget ϵ_{max}, dataset D, accuracy requirements (β-FNR, α-FPR)

1: **procedure** ProBE($Q^{\Lambda,\mathcal{F}}, \epsilon_{max}, D, \beta, \alpha$)
2: Tree minimization $(Q_c, o) \leftarrow minimize(Q)$ [37]
3: $(O_f, \epsilon_f, Q_c) \leftarrow$ PhaseOne($Q_c, o, \epsilon_{max}, D, \beta$)
4: $(O_f, \epsilon_f) \leftarrow$ PhaseTwo($Q_c, o, \epsilon_{max}, D, \beta, \alpha, \epsilon_f, O_f$)
5: **return** O_f, ϵ_f

minimization algorithms such as [37] to return the compact tree representation of the query T_c. Our mechanism subsequently extracts the number of occurrences o_i from T_c for each atomic query which will be used to compute the β_i budget as in Eq. (25).

5 IMPLEMENTING PROBE

Now that we have determined our β budget apportionment technique, we can implement ProBE by individually running mechanisms M_i on the atomic queries Q_i based on apportionment discussed in the previous section, then combining the results following the conjunction/disjunction operators as depicted in the query tree. However, simply implementing this framework incurs an arbitrary cost on the FPR when bounding the FNR as previously explained in §3.2. To address this, we develop an additional optimization which aims to provide an algorithmic bound on false positives.

In this section, we propose a two-phase algorithm that implements the ProBE apportionment framework to answer complex decision support queries with minimal privacy loss and bounds on utility. The overall algorithm is depicted in Algorithm 1. We first use a boolean minimization algorithm (Quine-McCluskey) [37] to minimize our query tree in order to obtain the occurrences of each sub-query within the query tree o and the tokenized version of this minimal tree Q_c (line 2). We subsequently run the two phases (lines 3-4). The first phase of the algorithm traverses the query tree and implements the apportionment framework from Section 4, which solves the optimization problem to guarantee the β-bound on FNR. The second phase then relaxes this solution by providing a post-optimization bound on the FPR. We first discuss Phase One, wherein we set the initial uncertain region parameter u to a large value in order to potentially obtain minimal privacy loss. Phase Two subsequently checks if the resulting false positives in Phase One exceed the FPR bound. If not, it uses intermediate results from Phase One to determine the next optimal uncertain region u_{opt} such that the α bound on FPR is met. Finally, we propose an iterative, entropy-based variant of the ProBE algorithm which further optimizes privacy loss in terms of Min-Entropy [24].

5.1 Phase One of ProBE

The first phase of the ProBE algorithm is detailed in Algorithm 2. This algorithm takes the minimized query Q_c, the set of occurrences of each sub-query o as well as the maximum privacy loss allowed ϵ_{max} and the FNR bound β. For each query node in the query tree, we first compute the initial uncertain region, which we set to a large percentage (30%) of its range of values. We also compute its corresponding FNR bound β_i according to Eq. (25) (lines 3-5). Note that because we are using a two-phase algorithm, we allocate half

Algorithm 2 First Phase of ProBE Mechanism.

1: **procedure** PhaseOne($Q_c, o, \epsilon_{max}, D, \beta$)
2: Initialize global budget variable $\epsilon_f \leftarrow 0$
3: **for** query node $Q_i \in Q_c$ **do**
4: $u_i \leftarrow 0.3 * range(Q_i)$
5: $\beta_i \leftarrow \frac{\Delta g_i (\beta/2) \prod_{x=1}^{n, x \neq i} (u_x)}{\sum_{y=1}^{n} \prod_{x=1}^{n, x \neq y} (u_x o_y \Delta g_y)}$
6: $flag_i = True$
7: $O_f \leftarrow$ Traverse($Q_c.root$)
8: **return** O_f, ϵ_f, Q_c
9: **function** Traverse(node)
10: **if** node is conjunction **then**
11: $O_l \leftarrow$ Traverse(node.left)
12: if O_l is empty, then skip node.right and **return** O_l
13: else, $O_r \leftarrow$ Traverse(node.right)
14: **return** $O_l \cap O_r$
15: **else if** node is disjunction **then**
16: $O_l \leftarrow$ Traverse(node.left)
17: $O_r \leftarrow$ Traverse(node.right)
18: **return** $O_l \cup O_r$
19: **else if** node is a query Q_i **then**
20: **if** $flag = True$ **then**
21: $O_i, \epsilon_i, G_i \leftarrow$ Tslm (Q_i, u_i, β_i, D)
22: Update global budget $\epsilon_f \leftarrow \epsilon_f + \epsilon_i$
23: $flag_i = False$
24: **if** $\epsilon_f > \epsilon_{max}$ **then**
25: Terminate program and **return** 'Query Denied'.
return O_i

of the β budget to each phase. However, we explore different β allocation strategies across the two phases in Appendix F (Additional Experiments). We also store a general flag parameter $flag_i$, which indicates if the DP mechanism (in this case, TSLM) should be run if the sub-query is encountered in the traversal. These parameters are stored within the node itself Q_i.

The algorithm can now begin traversing the tree in a pre-order traversal (i.e. starting from the root and executing the leaf nodes left to right). We first check if a leaf node is either a query or an operator. If it is an operator, we recursively call the traversal function in order to reach the leftmost leaf query node and then the rightmost leaf query node (lines 10-13/15-17). Depending on the nature of the operator (conjunction or disjunction), we either intersect the results of the left and right traversal (line 14) or union the results (line 18).

If the current node is a query Q_i, then we execute the TSLM mechanism provided that the $flag_i$ is True. Thus, TSLM is run with the appropriate β_i and u_i, which returns predicates whose noisy values are greater than their respective shifted thresholds $c_i - u_i$ into the result set O_i, the noisy values set G_i, as well as the resulting privacy loss ϵ_i computed with the equation $\epsilon_i = \frac{\Delta g_i ln(1/2\beta_i)}{u_i}$ (line 21). The resulting ϵ_i is subsequently accumulated into the global privacy budget variable ϵ_f, and the execution flag $flag_i$ is changed to False (lines 22-23). If the current privacy loss ϵ_f is above our maximum tolerated privacy loss ϵ_{max}, the query is denied. Otherwise, the output of the current node O_i is returned (line 25).

Skipping for conjunction queries. For the conjunction of n aggregate threshold queries, we note that the result of running a privacy-preserving mechanism like $M(D) = M_1(D) \cap M_2(D) \ldots \cap M_n(D)$ on such a query can be determined as false if one sub-mechanism $M_i(D)$ is evaluated as false due to the nature of the intersection operator. We exploit this fact by adding an optimization which skips the evaluation of sub-queries in conjunctions if any of the previous sub-queries return an empty set, thus further minimizing overall privacy loss. We implement this in line 12, where, upon returning the traversal results of the left node, if the output O_l is an empty set, we can automatically skip the execution of the right node and return the O_l result itself. Note that due to the recursive nature of this traversal, any subsequent conjunctions will also be skipped until a disjunction operator is met or the query terminates.

5.2 Phase Two of ProBE

The second phase of the ProBE Mechanism is depicted in Algorithm 3. This algorithm takes the β, α bounds, as well as the maximum budget ϵ_{max}. It also takes the resulting output O_{one} and privacy loss ϵ_{one} from Phase One. The query Q_i also includes the previously derived parameters (e.g. the noisy aggregates G_i and the β_i bound) resulting from Phase One. The algorithm starts by estimating the number of false positives as a result of running Phase One for each query node Q_i (line 4). We first describe the approach to derive the FPR estimate below.

Estimating the Bound on FPR. At the beginning of Phase Two, we first determine an upper bound on the FPR resulting from Phase One. If the latter is within the user-specified α bound, we can skip the second phase of the algorithm, otherwise we rerun some of the query nodes with an additional privacy budget. We empirically measure the FPR of each mechanism M_i for sub-query Q_i by the ratio between the number of false positives $|FP|$ and the number of negatives $|N|$. However, we cannot compute the truthful number of false positives and negatives without looking at the data, which consumes an additional privacy budget. Hence, we derive (i) an upper bound for the number of false positives and (ii) a lower bound for the number of negatives, to get an upper bound for FPR.

We first obtain the following observed results based on the noisy aggregates from Phase One:

$$O_{pp} \leftarrow \{\lambda_j \in \Lambda \mid G_i[j] > c_i\}$$

$$O_p \leftarrow \{\lambda_j \in \Lambda \mid G_i[j] > c_i - u_i\}$$

$$O_n \leftarrow \{\lambda_j \in \Lambda \mid G_i[j] < c_i - u_i\}$$

where $G_i[j]$ is the noisy aggregate for predicate j in the ith sub-query Q_i. In particular, O_p and O_n are the reported positives and negatives for Q_i by Phase One. O_{pp} are the predicates with large noisy counts which are "definitely positive".

A naive upper bound for the number of false positives is $|O_p|$, which includes all the reported positives. However, among them, the predicates in O_{pp} have noisy aggregates much larger than the testing threshold $c_i - u_i$ in TSLM, which are unlikely to be false positives. If all the definitely positive predicates in O_{pp} had true counts $< c - 2u_i$ (at worst case), they would have a noisy aggregate $> c - u_i$ and thus become a false positive with a probability $\leq \beta_i$ by the property of Laplace noise. Therefore, we can have an upper bound for the number of false positives:

$$|FP| \leq |O_p - O_{pp}| + |O_{pp}| \cdot \beta_i \tag{26}$$

The number of negatives $|N|$ is greater than $|O_n| - |FN|$, where $|FN|$ is the truthful number of false negatives. By the β_i-FNR property of TSLM, we have $|FN| \leq \beta_i(|G_i| - |N|)$, where $|G_i|$ is the total number of predicates in the input to query Q_i. Hence, we have this inequality

$$|N| \geq |O_n| - |FN| \geq |O_n| - \beta_i(|G_i| - |N|).$$

Solving this inequality by moving all the terms involving the unknown $|N|$, we have a lower bound to $|N|$,

$$|N| \geq \frac{|O_n| - \beta_i |G_i|}{1 - \beta_i}. \tag{27}$$

Algorithm 3 Second Phase of ProBE Mechanism.

1: **procedure** PhaseTwo($Q_c, o, \epsilon_{max}, \beta, \alpha, \epsilon_{one}, O_{one}$)
2: Let $O_f \leftarrow \{\}, \epsilon_f \leftarrow \epsilon_{one}$
3: **for** query node $Q_i(G_i, c_i, u_i, \beta_i, flag_i) \in Q_c$ **do**
4: $f_{est}, r_{est} \leftarrow$ EstimateFPs(Q_i, O_{one})
5: Compute allowed false positives $f_{max} \leftarrow \frac{\alpha}{n} r_i$
6: **if** $f_{est} > f_{max}$ **then**
7: Search $u_{opt} \leftarrow$ the largest u such that running EstimateFPs(Q_i, O_{one}) returns $f_{est} \leq f_{max}$
8: Set $flag_i \leftarrow True$ to rerun TSLM
9: $O_f \leftarrow$ Traverse($Q_c.root$)
10: **if** updated $f_{est} > f_{max}$ for any Q_i **then** Terminate program and **return** 'Query Denied'
11: **return** O_f, ϵ_f

Algorithm 4 Estimating FPs.

function EstimateFPs($Q_i(G_i, c_i, u_i, \beta_i, flag_i), O_{one}$)
$O_{pp} \leftarrow \{\lambda_j \in \Lambda \mid G_i[j] > c_i\}$
$O_p \leftarrow \{\lambda_j \in \Lambda \mid G_i[j] > c_i - u_i\}$
$O_n \leftarrow \{\lambda_j \in \Lambda \mid G_i[j] < c_i - u_i\}$
$O_{pp} \leftarrow O_{pp} \cap O_{one}, O_p \leftarrow O_p \cap O_{one}, O_n \leftarrow O_n - O_{one}$
Upper bound for FPs $f_{est} = |O_p - O_{pp}| + |O_{pp}| * \beta_i$
Lower bound for Negatives $r_{est} = \frac{|O_n| - \beta_i |G_i|}{1 - \beta_i}$
return f_{est}, r_{est}

The process of deriving an upper bound on the FPR is formalized in the function EstimateFPs. We retrieve the O_{pp}, O_p O_n. We then compute the upper bound on FP and lower bound on N according to Eq. (26) and (27) (lines 17-18).

This algorithm also offers an additional optimization based on the nature of the operator (i.e. conjunction/disjunction), through which certain elements in the uncertain region can be eliminated in the first step. We illustrate this through the example below.

Example 5.1. Consider an example query $Q = Q_1 \cap Q_2$. Consider a predicate λ_1 s.t. its noisy value falls within the uncertain region $[c_1 - u_1, c_1]$ of Q_1 i.e. is undecided, but falls within $[-\infty, c_2 - 2u_2]$ of Q_2 i.e. decidedly negative. We know that due to the nature of the $\cap$ operator, the result of intersecting the values for λ_1 from Q_1 and Q_2 will be negative without classifying λ_1 in Q_1. We can therefore eliminate λ_1 from the set of potential false positives within

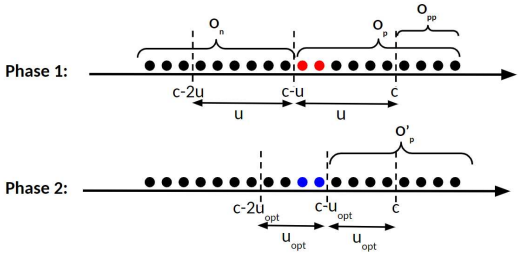


Figure 5: Illustration of sets O_p, O_{pp}, O_n on noisy values of predicates for Phase One. In Phase Two, we identify a new u_{opt} such that the newly observed positive set O'_p based on $> c - u_{opt}$ is reduced from O_p by a size of $f_{est} - f_{max}$ (indicated by dots changing from red to blue) if the number of estimated false positives f_{est} in Phase One is greater than the allowed number of false positives f_{max}.

the uncertain region to be used for the second phase. Conversely, consider the query $Q = Q_1 \cup Q_2$ and a predicate λ_1 s.t. λ_1 is reported as negative, i.e. falling into $[-\infty, c_1 - u_1]$ (it is in O_n), but its noisy value for Q_2 falls within $[c_2, \infty]$ of Q_2 i.e. decidedly positive. We know that the $\cup$ operator only requires one element to be positive for the result to be positive as well, so λ_1 can be classified as positive and thus removed from O_n.

The above optimization is illustrated in line 16, where we only keep predicates in O_p and O_{pp} which have appeared in the results of Phase One O_{one}, and we similarly only keep the negative predicates which are not in O_{one} . The estimated upper bound on false positives is stored in f_{est} and the lower bound on negatives in r_{est}. With these estimates for the FPR, the rest of the second step can be executed.

Resetting the Uncertain Region u_{opt}. After retrieving the upper bound on the current FPR, the second step algorithm first checks if the upper bound on FPs f_{est} is higher than the allowed number of FPs f_{max}. The latter value is derived from the bound α, which is divided equally amongst sub-queries i.e. α/n. We prove that this allocation ensures overall α in Appendix A.3. The bound is then multiplied by the estimated number of negatives stored in r_{est}.

If the bound is exceeded, we compute the next u_{opt} at which the "extra" false positives i.e. $f_{est} - f_{max}$ would reside outside the new threshold $c - u_{opt}$ as shown in Figure 5. In this way, the newly observed positives become O'_p, and its size is $f_{est} - f_{max}$ smaller than O_p. Hence, the newly estimated false positive negatives based on Eq. (26) are also reduced by that if $|O_p - O_{pp}| \geq f_{est} - f_{max}$. We do so by retrieving the noisy value of the cutoff predicate where the number of extra FPs would reside to the left of. This noisy value becomes our new lower bound for the uncertain region, i.e. $c_i - u_{opt}$. We thus solve for u_{opt} from this equality. Upon obtaining the new uncertain region, we can now rerun the TSLM algorithm with u_{opt} and β_i as parameters and accumulate the privacy loss resulting from this second phase. We do so by setting each execution flag $flag_i$ for the appropriate query nodes, then running the TRAVERSE function again (lines 8-9). We recompute the estimated FPs f_{est} after execution (line 10) by calling the estimation function again, and if the f_{max} FP bound is not met, then the query is denied and the algorithm exits, otherwise the resulting output and privacy loss O_f, ϵ_f are returned.

THEOREM 5.1. *Algorithm 1 satisfies $\epsilon_{\max}$-DP, a β-bound on the False Negative Rate and an α-bound on the False Positive Rate if the query is not denied.*

Proof of the theorem above can be found in Appendix A.4.

5.3 Multi-Step Entropy-based Algorithm

ProBE assigns different privacy levels to different predicates due to the early elimination of data points at the first step, meaning predicates that go on to the second step have a higher privacy loss ϵ. This concept is captured through the definition of Predicate-wise DP (PWDP)[24], a fine-grained extension of differential privacy which quantifies the different levels of privacy loss data points may have in multi-step algorithms.

Naturally, a measure to quantify this new definition of privacy is needed. Previous work tackles this problem by proposing a new privacy metric for PWDP entitled *Min-Entropy* [24] $\mathcal{H}_{min}$ which measures a lower bound on the level of uncertainty given the set of predicates and their respective privacy levels $\Theta = \{(\lambda_1, \epsilon_1), ..., (\lambda_k, \epsilon_k)\}$. the Data Dependent Predicate-Wise Laplace Mechanism (DDPWLM) also introduced in [24] maximizes min-entropy (i.e. maximizing the lower bound on uncertainty) in an iterative algorithm by also using the noisy aggregate values obtained from previous iterations to compute the best privacy level ϵ at which Min-Entropy is maximized for the set of elements in the uncertain region. This algorithm similarly guarantees a β-bound on FNR for a single aggregate threshold query by setting the privacy loss to $\epsilon = \frac{\Delta g \ln(1/(2\beta))}{u}$, but it does so by setting a starting privacy level ϵ_s as well as a maximum level ϵ_m prior to execution which is spent across a fixed number of iterations m. In each iteration, the privacy budget is further distributed across fine-grained steps m_f and the privacy level with the highest min-entropy is chosen as the next iteration's budget. It follows that the uncertain region parameter u is, again, chosen statically at the beginning, and therefore does not provide a bound on false positives.

We propose ProBE-Ent, a multi-step entropy-based algorithm which integrates DDPWLM with ProBE in order to not only answer complex DS queries with minimized entropy, but also to provide a post-facto bound on FPs. Instead of having a fixed starting ϵ_s, we internally choose a starting u_0 and compute the initial privacy level. Within the first step of DDPWLM, we run the FP estimation algorithm (Algorithm 4) in order to compute the optimal uncertain region parameter u_{opt}. We use this new uncertain region as an upper bound for the algorithm, i.e. we compute the ϵ_m upper bound that represents the exit condition for the algorithm. Additionally, we provide a β budget optimization which exploits the multi-step nature of the algorithm in a way that provides a potentially higher budget if previous sub-queries exit early. This is done by redistributing any remaining β_i which was not used in the current sub-query (due to early exit) to subsequent sub-queries, thus fully utilizing the β bound given and consequently minimizing the overall privacy loss. We provide a more detailed version of this algorithm with complete definitions in Appendix C.

6 EXPERIMENTS

In this section, we evaluate the ProBE algorithms previously explained on different real-life datasets based on various decision

support application types. We assess their performance (in terms of resulting privacy and utility) over a varying number of complex queries, and over examples of queries modeled after frequently used KPIs. Our experiments prove that all ProBE achieves its utility guarantees, while also successfully minimizing privacy loss for different levels of query complexity.

6.1 Experimental Setup

Datasets. To evaluate our approach, we use three real-world datasets from different domains. The first dataset, NYCTaxi, is comprised of New York City yellow taxi trip records in 2020 [1], where the data consists of 17 attributes and approximately 3 million records. The second data set, UCIDataset, is comprised of occupancy data in 24 buildings at the University of California, Irvine campus collected from April to May 2019 [40]. This data consists of 18 attributes and 5 million records. The final dataset, TurkishMarketSales, stores records of sale transactions at a chain supermarket across Turkey in 2017 [2], where the data consists of 1.2 million records and 26 attributes.

Query Benchmarks. For each dataset, we model several meaningful aggregate threshold queries as summarized in Table 1. In this paper, we present our experimental results for the COUNT aggregate function only. Queries with other aggregate functions (e.g. AVERAGE), which show similar results, are included in Appendix E. As per our previous definition of conjunction/disjunction queries, all of our sub-queries have the same predicates but different filters on certain attributes (e.g. sub-queries on the UCI dataset all check the same 41 rooms every day for 14 days for the count of records with filters on different attributes). For the Sales dataset specifically, we model our queries based on frequently used retail and sales KPIs. Table 2 summarizes the KPIs used, their respective definitions, and the aggregate functions used to represent them. For example, to illustrate the User Retention Rate KPI, we evaluate the distinct count of customers who visited a specific business in a specific time period. For the UCI and NYTaxi datasets, which do not have clearly defined KPIs, we select meaningful queries that are modeled based on anomaly detection or performance evaluation scenarios typically used in decision support applications, for example, selecting statistics based on specific demographics (e.g. age, gender, etc.) or specific performance criteria (e.g. fare amount is above the norm for a specific location). For each query, we select the corresponding threshold using the Z-score outlier detection method, to further emulate the concept of anomaly detection that decision support applications implement.

Algorithms. We test our ProBE approach and compare it against the Naive approach mentioned in the Introduction. This Naive approach simply proportions the FNR bound β into equal parts across the sub-queries Q_i and calls sub-mechanisms with this β_i. This approach does not optimize the privacy budget given the β constraint, nor does it offer utility guarantees on both β or α. In other words, this baseline is an extension of [24] where the β budget is split evenly across sub-queries. We do not evaluate our approach against the algorithm in [23], as [24] is directly based on it with the additional focus on bounding the FNR only. We also evaluate two variations of our ProBE framework; the two-step approach of ProBE using the Naive approach (i.e. splitting the β budget equally between sub-queries) as its Phase One entitled ProBE-Naive, and the entropy-based iterative variation described in §5.3 entitled ProBE-Ent.

Parameters. For all four algorithms, we set a FNR bound of $\beta = 0.05$, a maximum privacy loss of $\epsilon_{max} = 5$. For ProBE-based algorithms, we choose a large starting uncertain region of $u_0 = 30\%$ of the value range, and set a FPR bound of $\alpha = 0.1$. For the Naive algorithm we choose $u = 12\%$ as a default but explore other values in the experiments; we set this smaller default value due to the fact that the FPR is not bound in the Naive approach, meaning that a large u would result in a very high FPR. We run each algorithm over 100 iterations.

6.2 Experimental Results

Privacy Results. We use ex-post differential privacy, denoted by ϵ, as our privacy metric to evaluate the performance of our ProBE optimization as implemented in the previously mentioned algorithms[1]. We assess all four algorithms on exclusive conjunction, exclusive disjunction, and a randomized combination of both for a varying number of sub-queries (1 to 6) in order to evaluate the effect of the operator on privacy loss and utility. We only include conjunction and the combination in Figure 6, as disjunction yields the same results. Note that in the case of a single sub-query, the Naive and ProBE algorithms would yield the same privacy results provided that the second step of ProBE is not triggered.

As expected, the experiments show that ProBE-based algorithms achieve their respective bounds on the FPR and FNR. Furthermore, they achieve these bounds with minimal privacy loss. While the FPR and FNR bounds are set to 0.1 and 0.05 respectively, the actual rates are even smaller, while the maximum ϵ ranges from 1.5 in the Sales dataset to about 5 for the NYTaxi dataset. This difference is due to the underlying data distribution, which might trigger more predicates to be reran in the second step in order to guarantee the α bound. In contrast, the Naive algorithm results in a linear increase on the privacy budget, as well as high values specifically for the FPR. The iterative, entropy-based algorithm ProBE-Ent achieves close or the lowest privacy loss overall. This is due to the fact that ProBE-Ent makes use of the underlying data distribution to progressively compute the results of the query at the lowest privacy cost, while also allowing for early stopping at a much lower privacy loss if all predicates are properly classified outside of the uncertain region. As expected, the privacy loss increases as the number of sub-queries increases, but the Naive algorithm has a linear increase, whereas ProBE is less affected by the increasing complexity. The nature of the query (conjunction versus combination) does not cause a significant difference in pattern for privacy loss for the same data set.

Comparison of Naive versus Optimized Phase One. By analyzing the difference in results between the ProBE-Naive and ProBE algorithms, we are able to highlight the importance of the β budget optimization performed by ProBE in Phase One. In cases where query sensitivities and domain sizes dramatically differ (i.e. resulting in vastly different uncertain regions) across sub-queries, the optimal distribution would be to allocate a smaller β budget for the sub-query with the larger uncertain region u as they are inversely

[1]We include the additional privacy results in terms of Min-Entropy in the Appendix.

Dataset	Attributes used	Predicates	# of predicates p
UCI Dataset	*, age, userType, gender, groupName, office	room, date	41(rooms)*14(days)=574
Turkish Market Sales	*, region, netTotal, customerId, category, gender	city, date	43 (cities)*14(days)=602
NYTaxi	*, fareAmnt, totalAmnt, paymentType, location, storeFwdFlag	location, date	34(location)*15(days)=476

Table 1: Datasets used for experiments and respective attributes/predicates used

Sales KPI	Definition	Query Equivalent
Sales Volume	# of sales	Q_1: SELECT a GROUP BY a HAVING COUNT(*) > C
Regional Sales Volume	# of sales in region	Q_2: SELECT a WHERE region='A' GROUP BY a HAVING COUNT(*) >C
User Retention Rate	# of customers per period	Q_3: SELECT a GROUP BY a HAVING COUNT(DISTINCT(customerId)) > C
Conversion Rate	# of sales / # of customers	Q_4: SELECT a GROUP BY a HAVING COUNT(DISTINCT(customerId)) > C

Table 2: KPI-based Queries used for Sales Dataset experiments

correlated (a direct reflection of the trade-off between FN and FP), and allocate a larger β to the sub-query with a smaller u. In cases such as this, the Naive approach would result in a non-optimal equal distribution of the β budget and subsequently a higher privacy loss. We see this difference in privacy loss in Figure 6 in specific cases such as the Sales (row 2) and UCI (row 3) datasets where the addition of specific sub-queries results in a much higher privacy loss depending on underlying the data distribution.

Accuracy Results. To evaluate our algorithms, we use two accuracy metrics, the FNR, which is defined as the number of false negatives divided by the total number of positives, and the FPR, defined as the number of false positives divided by the total number of negatives. These metrics are averaged over the number of iterations run per algorithm, which we set to 100. We again use the default value of $\beta = 0.05$ for the bound on FNR and $\alpha = 0.1$ for the bound on FPR. We use the same uncertain region parameters as previously mentioned.

All algorithms achieve the guaranteed bound of $\beta = 0.05$ as seen in Figure 6 (columns 3-4), where the FNR is zero as the number of sub-queries increases for all datasets, hence the overlap of solid lines (FNR) for all algorithms. For the Naive algorithm, the FPR mostly sees a steady increase both in the conjunction-only and combination queries, whereas ProBE-based algorithms successfully meet the bound on FPR α at a lower privacy loss due to the upper bound estimation and dynamic re-computation of the uncertain region parameter u_{opt}. The pattern, however, largely depends on the data distribution of the underlying dataset; e.g. for the Sales dataset we note that the FPR decreases as the number of sub-queries increases, which may be attributed to selectivity of the query along with the distance of the data points from their respective thresholds.

Varying Uncertain Region. We vary the uncertain region parameter u for the Naive algorithm by setting its values to $\{5, 10, ..., 30\}$% of the data range on the Sales dataset in order to evaluate its performance compared to ProBE-based algorithms, which internally set their u (first by setting its initial value to a conservative 30% then by recomputing it in the second step). We use the query $Q_1 \cup (Q_2 \cap Q_3)$ where Q_1, Q_2, Q_3 refer to the first three KPIs from Table 2, and the thresholds are computed using the outlier method again. We fix the other parameters to $\beta = 0.05$ and $\alpha = 0.1$.

Figure 7(a-b) shows that as the uncertain region increases, privacy loss declines for all algorithms as expected, seen in Figure 7a. We see that for the Naive algorithm, the FPR steadily increases as the uncertain region increases, whereas the privacy loss ϵ steadily decreases as the u increases. This shows not only the direct correlation between false positives and u, but also the extent of the trade-off between privacy loss and the FPR. For our ProBE algorithms, since they do not take the parameter u but rather internally set it in an optimal way, the privacy level and FPR are constant across plots. The FNR does not change for any algorithm due to the upheld β bound guarantee.

Query Trees and Operator Distribution. As discussed in §4.3, the structure of a query tree has an effect on privacy loss. To evaluate the impact of operator distribution (i.e. distributing a conjunction over a disjunction and vice versa) we run ProBE-based algorithms and Naive algorithms for the two queries illustrated in Figure 4 (c-d): $Q_I = Q_1 \cup (Q_2 \cap Q_3)$ and $Q_{II} = (Q_1 \cup Q_2) \cap (Q_1 \cup Q_3)$. We run these queries on the Sales dataset with the default parameters of $\beta = 0.05$, as well as a uncertain region parameter of $u_0 = 12\%$ for the Naive algorithm. Figure 7 shows that the distributed query Q_{II} incurs a higher ex-post privacy cost than the grouped query Q_I for all algorithms. Similarly, the FPR for the Naive algorithm sees a slightly higher value for query Q_{II} as compared to the original query Q_I. Experiments ran on distributing the AND operator over or (i.e. $Q_1 \cap (Q_2 \cup Q_3)$ vs. $(Q_1 \cap Q_2) \cup (Q_1 \cap Q_3)$) showed similar results. This is attributed to the fact that Q_1 was allocated an additional privacy budget due to its second occurrence in Q_{II}, thus incurring a higher cost on the overall privacy loss due to the use of the Sequential Composition theorem.

Varying β and α Parameters. We analyze the effect of selecting different values for the user-set FNR bound β and FPR bound α on all four algorithms. We vary the two bounds by setting their values to $\{0.025, 0.05, 0.075, 0.1, 0.125, 0.15\}$ on the Taxi dataset with a 3-disjunction query (i.e. $Q_1 \cup Q_2 \cup Q_3$). Figure 8 (a) shows that varying β causes privacy loss ϵ to decrease as β increases across all algorithms, which is to be expected due to the inverse correlation between ϵ and β. The accuracy measures depicted in Figure 8 (b), i.e. the FNR and FPR, are not significantly impacted as their respective bounds are met by the ProBE-based algorithms. Conversely Figure 8 (c-d) depicts the effect of varying the α bound on FPR. As the Naive algorithm does not support a mechanism to bound the FPR, the privacy and accuracy results remain constant. On the other hand, we note that the privacy loss ϵ also decreases as the α bound

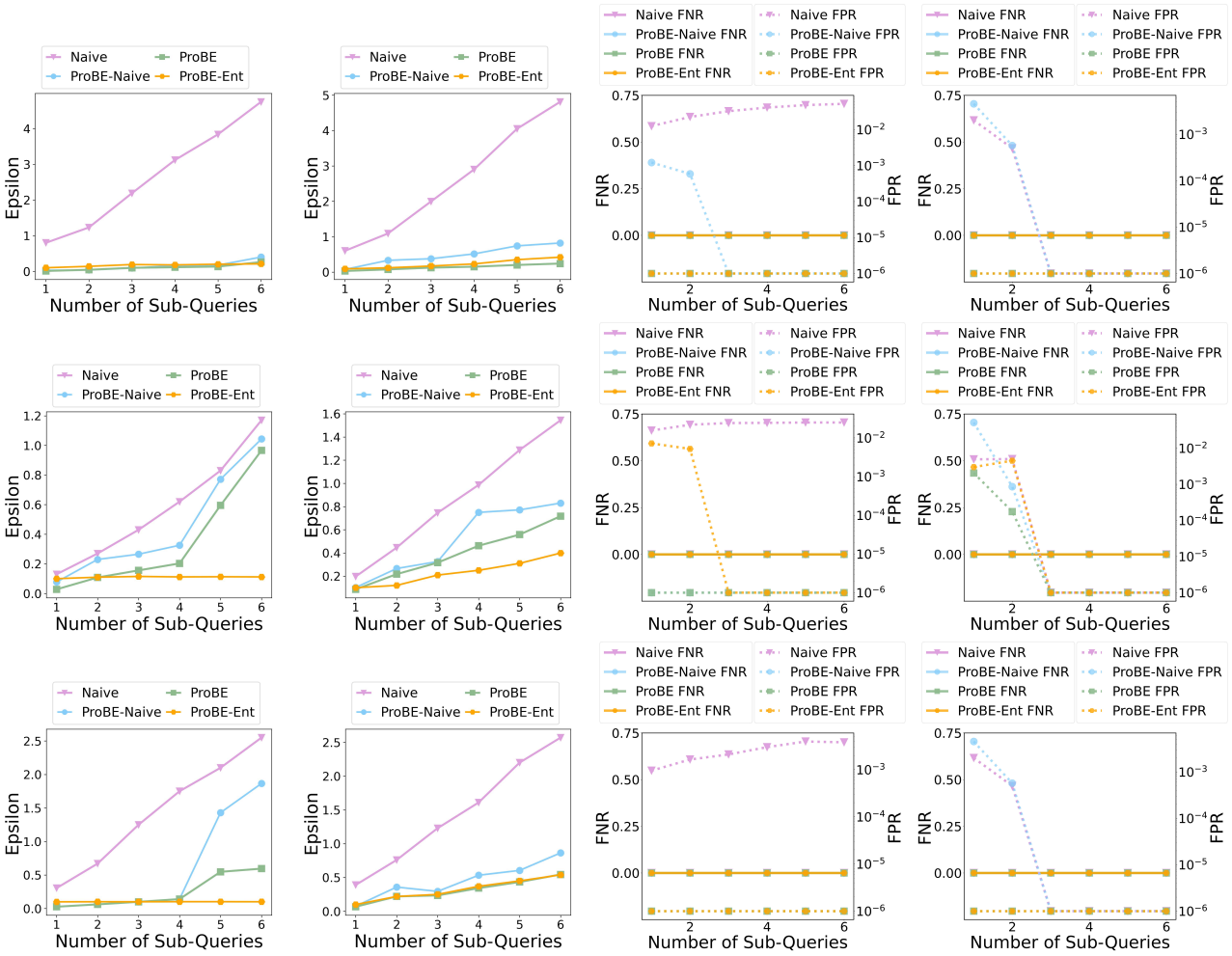


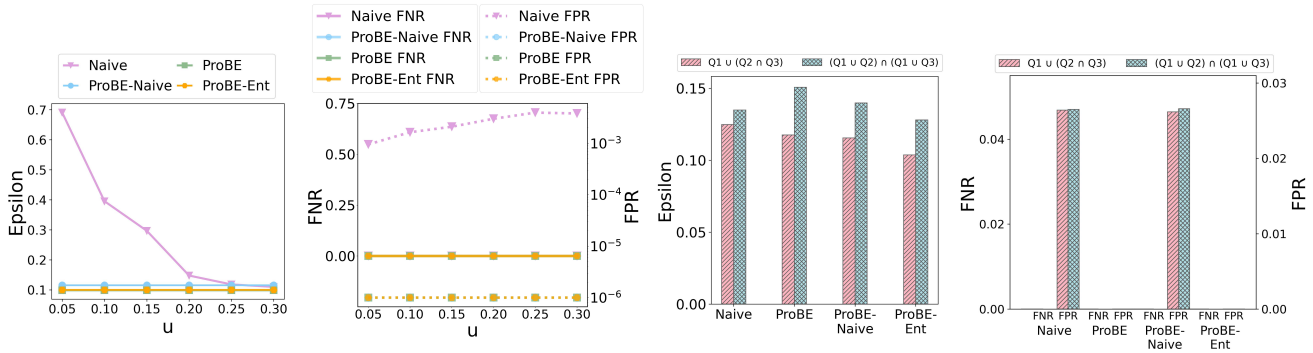
Figure 6: Privacy Loss in terms of Ex-Post DP ϵ (cols 1,2) and Accuracy in terms of FNR and FPR (cols 3,4) for 1-6 sub-queries at $\beta = 0.05$ for Conjunction (cols 1,3) and Combined Conjunction/Disjunction (cols 2,4) using NYTaxi (row 1), Sales (row 2) and UCI (row 3) data. The Accuracy plots have two axes, left for FNR (with a range of $[0, 0.75]$) and right for FPR (with a range of $[10^{-6}, 10^{-2}]$).

is increased for ProBE-based algorithms. This is attributed to the fact that, as the α bound increases, the probability of Phase Two being run decreases as the tolerance for FP errors is higher, thus avoiding the additional privacy cost of the second step. However, this decrease is dependent upon the data distribution and how many elements are in the uncertain region; if the estimated FPR is lower than a smaller bound (e.g. 0.1) then increasing α will have no additional on privacy loss, hence the somewhat constant FPR between $0.1 - 0.15$ for the ProBE-based algorithms. In terms of accuracy, the FNR and FPR similarly do not see a significant change, as their respective bounds are met regardless of their values. Setting a value for the β and α bounds are thus entirely dependent on the use case of the decision support application built upon our ProBE algorithm and its purpose, as well as the underlying data distribution. Therefore, ProBE allows the user the flexibility of exploring the trade-off between privacy and accuracy in a way that meets their various requirements.

7 RELATED WORK

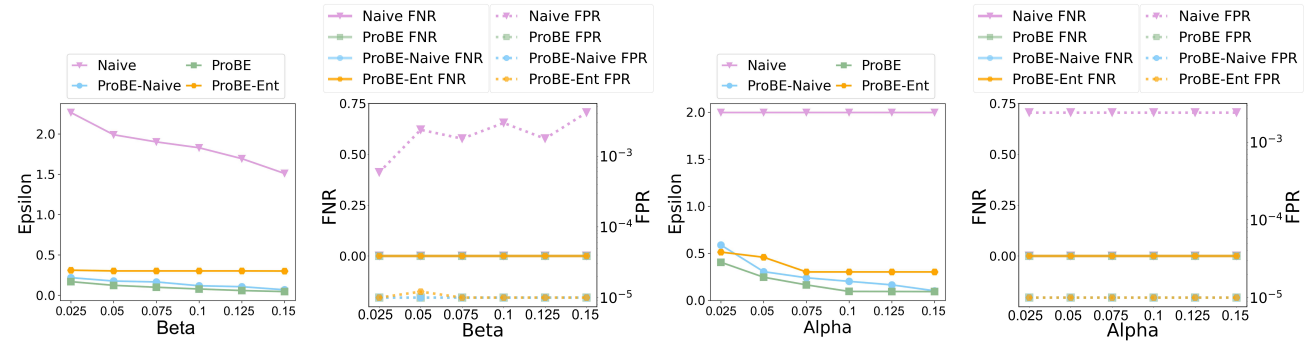
Differential Privacy [18, 19] has become a well-studied standard for privacy-preserving data exploration and analysis [12, 27, 48, 51]. Various work has been proposed to answer queries with DP, such as range queries [16, 30, 54], and linear counting queries [32, 38, 39]. This body of work, however, is not applicable to the aggregate threshold queries that our solution considers. Join queries [11, 13, 28, 52] may be more applicable to such queries (specifically for conjunctions), but they do not encompass the full scope of the complex queries we tackle, nor does recent work apply a utility-first approach to their solutions.

Accuracy-constrained systems for differentially private data analysis have been proposed in recent years [23, 33, 34, 36], which allow



(a) Ex-Post DP ϵ (b) FNR and FPR (c) Ex-Post DP ϵ (d) FNR and FPR

Figure 7: Privacy (ϵ) and Accuracy (FNR and FPR) at $\beta = 0.05$ for varying u (a-b) and for two equivalent query trees (c-d) on Sales data



(a) Ex-Post DP ϵ (b) FNR and FPR (c) Ex-Post DP ϵ (d) FNR and FPR

Figure 8: Ex-Post DP (ϵ) and Accuracy (FNR, FPR) Results for varying β (a-b) and α (c-d) values on Taxi data.

data analysts to interactively specify accuracy requirements over their queries while providing a formal privacy guarantee. However, these solutions either do not specifically focus on decision support queries (CacheDP [36], Ligett et al. [33]), or do not take into account their asymmetric utility accuracy requirements (APEx [23] and DPella [34]).

The problem of incorporating differential privacy in decision support applications has been tackled in MIDE [24], which we have previously described and extended in our work. Other solutions such as Fioretto et al. [22] also studies decision support on differentially private data, but does so from a fairness lens by proposing recommendations to mitigate bias resulting from making decisions on DP data. Detailed related work can be found in Appendix C.

8 CONCLUSION AND FUTURE WORK

In this paper, we proposed ProBE, an optimization framework which enables the execution of complex decision support queries under utility requirements on the false positive and negative rates at a minimal privacy loss. A natural generalization of our framework is implementing it with different DP mechanisms. One such mechanism defined in [24] explores a more fine-grained definition of DP, where predicates have different privacy budgets, thus necessitating a new metric to quantify this privacy loss entitled Min-Entropy. We thus extend the entropy-based algorithm from [24] to answer complex queries with guarantees on both error rates as formalized in Appendix C. Another interesting direction would be to implement the ProBE framework with other widely used DP mechanisms such as the exponential mechanism [19] or the matrix mechanism [32] to compare their performance to the algorithms previously used.

ACKNOWLEDGMENTS

This work was supported by the HPI Research Center in Machine Learning and Data Science at UC Irvine. It is also supported by NSF Grants No. 2032525, 1545071, 1527536, 1952247, 2008993, 2133391, and 2245372.

REFERENCES

[1] 2020. TLC Trip Record Data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page. Accessed: 2021-12-31.
[2] 2020. Turkish Market Sales Dataset. https://www.kaggle.com/datasets/omercolakoglu/turkish-market-sales-dataset-with-9000items. Accessed = 2023-01-15.
[3] Amitav Banerjee, U. B. Chitnis, S. L. Jadhav, J. S. Bhawalkar, and S. Chaudhury. 2009. Hypothesis testing, type I and type II errors. *Industrial Psychiatry Journal* 18 (2009). Issue 2.
[4] Dimitri P. Bertsekas. 1982. *Constrained Optimization and Lagrange Multiplier Methods.* Academic Press. https://doi.org/10.1016/C2013-0-10366-2

[5] Surajit Chaudhuri and Umeshwar Dayal. 1997. Data warehousing and OLAP for decision support. *Proceedings of the 1997 ACM SIGMOD International Conference on Management* (1997), 507–508. https://doi.org/10.1145/253260.253373

[6] Surajit Chaudhuri, Bolin Ding, and Srikanth Kandula. 2017. Approximate Query Processing: No Silver Bullet. *SIGMOD '17* (2017), 511–519. https://doi.org/10.1145/3035918.3056097

[7] Gabriella Dellino, Teresa Laudadio, Renato Mari, Nicola Mastronardi, and Carlo Meloni. 2018. A reliable decision support system for fresh food supply chain management. *International Journal of Production Research* 56 (2018), 1458–1485. Issue 4.

[8] Irit Dinur and Kobbi Nissim. 2003. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. 202–210.

[9] Uwe Dombrowski, David Ebentreich, and K. Schmidtchen. 2013. Balanced Key Performance Indicators in Product Development. *International Journal of Materials, Mechanics and Manufacturing* 1, 1 (2013), 27–31. doi.org/10.7763/IJMMM.2013.V1.6

[10] Jinshuo Dong, Aaron Roth, and Weijie J. Su. 2022. Gaussian Differential Privacy. *Journal of the Royal Statistical Society Series B: Statistical Methodology* 84 (2022), 3–37. Issue 1. https://doi.org/10.1111/rssb.12454

[11] Wei Dong, Juanru Fang, Ke Yi, Yuchao Tao, and Ashwin Machanavajjhala. 2023. R2T: Instance-optimal Truncation for Differentially Private Query Evaluation with Foreign Keys. *ACM SIGMOD Record* 52 (2023), 115–123. Issue 1. https://doi.org/10.1145/3604437.3604462

[12] Wei Dong and Ke Yi. 2021. Residual Sensitivity for Differentially Private Multi-Way Joins. *SIGMOD '21* (2021), 432–444. https://doi.org/10.1145/3448016.3452813

[13] Wei Dong and Ke Yi. 2021. Residual Sensitivity for Differentially Private Multi-Way Joins. *SIGMOD '21* (2021), 432–444. https://doi.org/10.1145/3448016.3452813

[14] Haris Doukas, Konstantinos D. Patlitzianas, Konstantinos Iatropoulos, and John Psarras. 2007. Intelligent building energy management system using rule sets. *Building and Environment* 42 (2007), 3562–3569. Issue 10.

[15] Shirley Dowdy, Stanley Wearden, and Daniel Chilko. 2011. *Statistics for research*. John Wiley & Sons,.

[16] Linkang Du, Zhikun Zhang, Shaojie Bai, Changchang Liu Liu, Shouling Ji, Peng Cheng, and Jiming Chen. 2021. AHEAD: Adaptive Hierarchical Decomposition for Range Query under Local Differential Privacy. *CCS '21* (2021), 1266–1288. https://doi.org/10.1145/3460120.3485668

[17] Fikri Dweiri, Sameer Kumar, Sharfuddin Ahmed Khan, and Vipul Jain. 2016. Designing an integrated AHP based decision support system for supplier selection in automotive industry. *Expert Systems with Applications: An International Journal* 62 (2016), 273–283. Issue C. https://doi.org/10.1016/j.eswa.2016.06.030

[18] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. *Proceedings of the Third Conference on Theory of Cryptography* (2006), 265–284. https://doi.org/10.1007/11681878_14

[19] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Foundation and Trends in Theoretical Computer Science* 9, 3–4 (Aug. 2014), 211–407. https://doi.org/10.1561/0400000042

[20] Cynthia Dwork and Sergey Yekhanin. 2008. New efficient attacks on statistical disclosure control mechanisms. In *Annual International Cryptology Conference*. Springer, 469–480.

[21] Giulia Fanti, Vasyl Pihur, and Úlfar Erlingsson. 2016. Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries. *Proceedings on Privacy Enhancing Technologies* 3 (2016).

[22] Ferdinando Fioretto, Cuong Tran, Pascal Van Hentenryck, and Zhiyan Yao. 2021. Decision Making with Differential Privacy under a Fairness Lens. https://doi.org/10.24963/ijcai.2021/78

[23] Chang Ge, Xi He, Ihab F Ilyas, and Ashwin Machanavajjhala. 2019. APEx: Accuracy-Aware Differentially Private Data Exploration *(SIGMOD)*.

[24] Sameera Ghayyur, Dhrubajyoti Ghosh, Xi He, and Sharad Mehrotra. 2022. MIDE: Accuracy Aware Minimally Invasive Data Exploration For Decision Support. *Proceedings of the VLDB Endowment* 15, 11 (2022), 2653–2665. https://doi.org/10.14778/3551793.3551821

[25] Kannan Govindan, Hassan Mina, and Behrouz Alavi. 2020. A decision support system for demand management in healthcare supply chains considering the epidemic outbreak: A case study of coronavirus disease 2019. *Transportation Research Part E: Logistics and Transportation Review* 138 (2020). https://doi.org/10.1016/j.tre.2020.101967

[26] Samuel Haney, Ashwin Machanavajjhala, John M. Abowd, Matthew Graham, Mark Kutzbach, and Lars Vilhuber. 2017. Utility cost of formal privacy for releasing national employer-employee statistics. *SIGMOD '17* (2017), 1339–1354. https://doi.org/10.1145/3035918.3035940

[27] Fumiyuki Kato, Tsubasa Takahashi, Shun Takagi, Yang Cao, Seng Pei Liew, and Masatoshi Yoshikawa. 2022. HDPView: Differentially Private Materialized View for Exploring High Dimensional Relational Data. *Proceedings of the VLDB Endowment* 15, 9 (2022).

[28] Ios Kotsogiannis, Yuchao Tao, Xi He, Maryam Fanaeepour, Ashwin Machanavajjhala, Michael Hay Hay, and Gerome Miklau. 2019. PrivateSQL: A Differentially Private SQL Query Engine. *Proceedings of the VLDB Endowment* 12 (2019), 1371–1384. Issue 11. https://doi.org/10.14778/3342263.3342274

[29] Stéphane Lallich, Olivier Teytaud, and Elie Prudhomme. 2006. Statistical inference and data mining: false discoveries control. *Compstat 2006 - Proceedings in Computational Statistics* (2006), 325–336.

[30] Chao Li, Michael Hay, Gerome Miklau, and Yue Wang. 2014. Data- and Workload-Aware Algorithm for Range Queries Under Differential Privacy. *Proceedings of the VLDB Endowment* 7 (2014), 341–352. Issue 5. https://doi.org/10.14778/2732269.2732271

[31] Chao Li, Michael Hay, Gerome Miklau, and Yue Wang. 2014. A data-and workload-aware algorithm for range queries under differential privacy. *Proceedings of the VLDB Endowment* 7, 5 (2014), 341–352. https://arxiv.org/abs/1410.0265

[32] Chao Li, Gerome Miklau, Michael Hay, Andrew McGregor, and Vibhor Rastogi. 2015. The matrix mechanism: optimizing linear counting queries under differential privacy. *VLDB Journal* 24, 6 (2015). https://dl.acm.org/doi/10.1007/s00778-015-0398-x

[33] Katrina Ligett, Seth Neel, Aaron Roth, Bo Waggoner, and Zhiwei Wu. 2017. Accuracy First: Selecting a Differential Privacy Level for Accuracy-Constrained ERM. *Journal of Privacy and Confidentiality* 9 (05 2017). https://doi.org/10.29012/jpc.682

[34] E. Lobo-Vesga, A. Russo, and M. Gaboardi. 2020. A Programming Framework for Differential Privacy with Accuracy Concentration Bounds. In *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, Los Alamitos, CA, USA, 411–428. https://doi.org/10.1109/SP40000.2020.00086

[35] S. Afshin Mansouri, David Gallear, and Mohammad H. Askariazad. 2012. Decision support for build-to-order supply chain management through multiobjective optimization. *International Journal of Production Economics* 135 (2012), 24–36. Issue 1.

[36] Miti Mazmudar, Thomas Humphries, Jiaxiang Liu, Matthew Rafuse, and Xi He. 2022. Cache Me If You Can: Accuracy-Aware Inference Engine for Differentially Private Data Exploration. *Proceedings of the VLDB Endowment* 16 (2022), 574–586. Issue 4. https://doi.org/10.14778/3574245.3574246

[37] E. J. McCluskey. 1956. Minimization of Boolean functions. *The Bell System Technical Journal* 35, 6 (1956).

[38] Ryan McKenna, Raj Kumar Maity, Arya Mazumdar, and Gerome Miklau. 2020. A workload-adaptive mechanism for linear queries under local differential privacy. *Proceedings of the VLDB Endowment* 13, 12 (2020), 1905–1918. https://doi.org/10.14778/3407790.3407798

[39] Ryan McKenna, Gerome Miklau, Michael Hay, and Ashwin Machanavajjhala. 2018. Optimizing error of high-dimensional statistical queries under differential privacy. *Proceedings of the VLDB Endowment* 11, 10 (2018), 1206–1219. https://doi.org/10.14778/3231751.3231769

[40] Sharad Mehrotra, Kobsa Alfred, Nalini Venkatasubramanian, and Siva Raj Rajagopalan. 2016. TIPPERS: A privacy cognizant IoT environment. In *2016 IEEE PerCom Workshops*.

[41] Ilya Mironov. 2017. Rényi Differential Privacy. *2017 IEEE 30th Computer Security Foundations Symposium (CSF)* (2017), 263–275. https://doi.org/10.1109/CSF.2017.11

[42] Moti Mizrahi. 2020. Hypothesis Testing in Scientific Practice: An Empirical Study. *International Studies in the Philosophy of Science* 33 (2020). Issue 1.

[43] Prashanth Mohan, Abhradeep Thakurta, Elaine Shi, Dawn Song, and David Culler. 2012. GUPT: privacy preserving data analysis made easy. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*. 349–360.

[44] Mark A. Musen, Blackford Middleton, and Robert A. Greenes. 2021. Clinical Decision-Support Systems. In *Biomedical Informatics*. Springer, 795–840.

[45] Whitney K. Newey and Daniel McFadden. 1994. Large sample estimation and hypothesis testing. In *Handbook of Econometrics*. Elsevier B.V., 2111–2245.

[46] N. Peiffer-Smadja, T.M. Rawson Rawson, R. Ahmad, A. Buchard, P. Georgiou, F.-X. Lescure Lescure, G. Birgand, and A.H. Holmes. 2020. Machine learning for clinical decision support in infectious diseases: a narrative review of current applications. *Clinical Microbiology and Infection* 26 (2020). Issue 5.

[47] Meikel Poess, R.O. Nambiar, and David Walrath. 2007. Why you should run TPC-DS: a workload analysis. *Proceedings of the 33rd International Conference on Very Large Databases* (2007), 1138–1149. https://dl.acm.org/doi/10.5555/1325851.1325979

[48] David Pujol, Albert Sun, Brandon Fain, and Ashwin Machanavajjhala. 2022. Multi-Analyst Differential Privacy for Online Query Answering. *Proceedings of the VLDB Endowment* 16, 4 (2022).

[49] Ryan Rogers, Aaron Roth, Jonathan Ullman, and Salil Vadhan. 2016. Privacy Odometers and Filters: Pay-as-you-Go Composition. *NIPS'16* (2016), 1929–1937.

[50] Reed T. Sutton, David Pincock, Daniel C. Baumgart, Daniel C. Sadowski, Richard N. Fedorak, and Karen I. Kroeker. 2020. An overview of clinical decision support systems: benefits, risks, and strategies for success. *npj Digit. Med* 3 (2020). Issue 17.

[51] Yuchao Tao, Amir Gilad, Ashwin Machanavajjhala, and Sudeepa Roy. 2022. DPXPlain: Privately Explaining Aggregate Query Answers. *Proceedings of the VLDB Endowment* 16, 1 (2022).

[52] Yuchao Tao, Xi He, Ashwin Machanavajjhala, and Sudeepa Roy. 2020. Computing Local Sensitivities of Counting Queries with Joins. *SIGMOD '20* (2020), 479–494.

https://doi.org/10.1145/3318464.3389762
[53] Benjamin P. Thompson and Lawrence C. Bank. 2010. Use of system dynamics as a decision-making tool in building design and operation. *Building and Environment* 45 (2010), 1006–1015. Issue 4.
[54] Jianyu Yang, Tianhao Wang, Ninghui Li, Xiang Cheng, and Sen Su. 2020. Answering Multi-Dimensional Range Queries under Local Differential Privacy. *Proceedings of the VLDB Endowment* 14, 3 (2020), 378–390. https://doi.org/10.14778/3430915.3430927

A PROOFS

A.1 Proofs for Single Operator Queries (FNR)

1. Proof For 2-Query Conjunction. Consider a complex decision support query Q composed of two atomic aggregate threshold queries $Q = Q_1 \cap Q_2$. Q_1 and Q_2 are answered by TSLM M_1, M_2 respectively, and mechanism $M = M_1 \cap M_2$ is used to answer query Q. If M_1 and M_2 guarantee an FNR bound of β_1 and β_2 respectively, then M has an FNR bound of $\beta_1 + \beta_2$.

Proof. $\forall \lambda_i \in \Lambda$,
Let A be the event that $\lambda_i \in M_1(D)$, B be the event that $\lambda_i \in M_2(D)$, C be the event that $\lambda_i \in Q_1(D)$ and D be the event that $\lambda_i \in Q_2(D)$. The probability of false negatives can be derived as:

$$\begin{aligned} FNR &= P[\lambda_i \notin M_1(D) \cap M_2(D) | \lambda_i \in Q_1(D) \cap Q_2(D)] \\ &= P[\overline{AB}|CD] = P[\overline{A} \vee \overline{B}|CD] = P[\overline{A}B \vee A\overline{B} \vee \overline{AB}|CD] \\ &= \frac{P[(\overline{A}B \vee A\overline{B} \vee \overline{AB}) \wedge CD]}{P[CD]} \\ &= \frac{P[(\overline{A}B \wedge CD) \vee (A\overline{B} \wedge CD) \vee (\overline{AB} \wedge CD)]}{P[CD]} \end{aligned} \tag{28}$$

Given that the three clauses are mutually exclusive events

$$\begin{aligned} &= \frac{P[\overline{A}B \wedge CD] + P[A\overline{B} \wedge CD] + P[\overline{AB} \wedge CD]}{P[CD]} \\ &= P[\overline{A}B|CD] + P[A\overline{B}|CD] + P[\overline{AB}|CD] \end{aligned}$$

We know that M_1 and M_2 are mechanisms of independent randomness, i.e. add random noise. This means that A is independent of B given CD and vice versa. Therefore we can rewrite $P[\overline{A}B|CD] = P[\overline{A}|CD]P[B|CD]$. We obtain:

$$= P[\overline{A}|CD]P[B|CD] + P[A|CD]P[\overline{B}|CD] + P[\overline{A}|CD]P[\overline{B}|CD]$$

There is a conditional independence between A and D given C due to the nature of the randomized mechanisms. Similarly there is a conditional independence between B and C given D. Thus, we can rewrite $P[\overline{A}|CD]P[B|CD] = P[\overline{A}|C]P[B|D]$. Therefore,

$$= P[\overline{A}|C]P[B|D] + P[A|C]P[\overline{B}|D] + P[\overline{A}|C]P[\overline{B}|D]$$

Knowing that $FNR_1 = P[\overline{A}|C]$, $FNR_2 = P[\overline{B}|D]$, $TPR_1 = P[A|C]$ and $TPR_2 = P[B|D]$, we substitute:

$$\begin{aligned} &= FNR_1 \cdot TPR_2 + TPR_1 \cdot FNR_2 + FNR_1 \cdot FNR_2 \\ &= FNR_1(1 - FNR_2) + (1 - FNR_1)FNR_2 + FNR_1 FNR_2 \\ &= FNR_1 + FNR_2 - FNR_1 FNR_2 \\ &\leq FNR_1 + FNR_2 \leq \beta_1 + \beta_2 \end{aligned} \tag{29}$$

□

2. Proof For 2-Query Disjunction. Consider a complex decision support query Q composed of two atomic aggregate threshold queries $Q = Q_1 \cup Q_2$. Q_1 and Q_2 are answered by TSLM M_1, M_2 respectively, and mechanism $M = M_1 \cup M_2$ is used to answer query Q. If M_1 and M_2 guarantee an FNR bound of β_1 and β_2 respectively, then M has an FNR bound of $\beta_1 + \beta_2$.

Proof. $\forall \lambda_i \in \Lambda$,
Let A be the event that $\lambda_i \in M_1(D)$, B be the event that $\lambda_i \in M_2(D)$, C be the event that $\lambda_i \in Q_1(D)$ and D be the event that $\lambda_i \in Q_2(D)$. The probability of false negatives can be derived as:

$$\begin{aligned} FNR &= P[\lambda_i \notin M_1(D) \cap M_2(D) | \lambda_i \in Q_1(D) \cap Q_2(D)] \\ &= P[\overline{A \vee B}|C \vee D] \\ &= \frac{P[(\overline{AB}) \wedge (\bar{C}D \vee C\bar{D} \vee CD)]}{P[C \vee D]} \end{aligned}$$

Given that the three clauses are mutually exclusive events

$$\begin{aligned} &= \frac{P[(\overline{AB}) \wedge \bar{C}D] + P[(\overline{AB}) \wedge C\bar{D}] + P[(\overline{AB}) \wedge CD]}{P[C \vee D]} \\ &= \frac{P[(\overline{AB})|\bar{C}D]P[\bar{C}D]}{P[C \vee D]} + \frac{P[(\overline{AB})|C\bar{D}]P[C\bar{D}]}{P[C \vee D]} \\ &+ \frac{P[(\overline{AB})|CD]P[CD]}{P[C \vee D]} \\ &\leq P[\overline{AB}|\bar{C}D] + P[\overline{AB}|C\bar{D}] + P[\overline{AB}|CD] \end{aligned}$$

We know that M_1 and M_2 are mechanisms of independent randomness, i.e. add random noise. This means that A is independent of B given CD and vice versa. Therefore we can rewrite $P[\overline{AB}|\overline{C}D] = P[\overline{A}|\overline{C}D]P[\overline{B}|\overline{C}D]$. We obtain:

$$= P[\overline{A}|\overline{C}D]P[\overline{B}|\overline{C}D] + P[\overline{A}|C\overline{D}]P[\overline{B}|C\overline{D}] + P[\overline{A}|CD]P[\overline{B}|CD]$$

There is a conditional independence between A and D given C due to the nature of the randomized mechanisms. Similarly there is a conditional independence between B and C given D. Thus, we can rewrite $P[\overline{A}|\overline{C}D]P[\overline{B}|\overline{C}D] = P[\overline{A}|\overline{C}]P[\overline{B}|D]$. Therefore,

$$= P[\overline{A}|\overline{C}]P[\overline{B}|D] + P[\overline{A}|C]P[\overline{B}|\overline{D}] + P[\overline{A}|C]P[\overline{B}|D]$$

Knowing that $FNR_1 = P[\overline{A}|C]$, $FNR_2 = P[\overline{B}|D]$, $TNR_1 = P[\overline{A}|\overline{C}]$ and $TNR_2 = P[\overline{B}|\overline{D}]$, we substitute:

$$\begin{aligned} &= TNR_1 \cdot FNR_2 + FNR_1 \cdot TNR_2 + FNR_1 \cdot FNR_2 \\ &= TNR_1 \cdot FNR_2 + FNR_1(TNR_2 + FNR_2) \\ &\leq FNR_2 + FNR_1 \leq \beta_1 + \beta_2 \end{aligned} \tag{30}$$

□

3. Proof for Theorem 4.2. Given a disjunction/conjunction query Q answered by a disjunction/conjunction mechanism $M(D)$ where M_i is a Threshold Shift Laplace mechanism and $\epsilon = \epsilon_1 + \epsilon_2$, we achieve minimum privacy loss ϵ and a β-bound on FNR by budgeting β as:

$$\beta_1 = \frac{u_2 \Delta g_1 \beta}{u_1 \Delta g_2 + u_2 \Delta g_1}, \beta_2 = \frac{u_1 \Delta g_2 \beta}{u_1 \Delta g_2 + u_2 \Delta g_1} \tag{31}$$

Proof. Consider a complex decision support query Q composed of two atomic aggregate threshold queries connected by the disjunction or conjunction operator. For both conjunction and disjunction mechanisms, we obtained the optimization problem of:

$$\begin{aligned} &\text{maximize} \quad (\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}} \\ &\text{subject to} \quad \beta_1 + \beta_2 \leq \beta \end{aligned} \tag{32}$$

from Eq. (11), (15) for conjunction and Eq. (19), (20) for disjunction. We use the Lagrange Multipliers method to solve this constrained optimization problem, which consists of solving the Lagrangian function that sets the gradient of the function equal to the gradient of the constraint multiplied by the Lagrange Multiplier λ. We solve for λ below:

$$\begin{aligned} \frac{d}{d\beta_2}((\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}}) &= \lambda \frac{d}{d\beta_2}(\beta_1 + \beta_2) \\ (\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}-1} \cdot \frac{\Delta g_2}{u_2} &= \lambda \end{aligned} \tag{33}$$

Similarly,

$$\begin{aligned} \frac{d}{d\beta_1}((\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}}) &= \lambda \frac{d}{d\beta_1}(\beta_1 + \beta_2) \\ (\beta_1)^{\frac{\Delta g_1}{u_1}-1}(\beta_2)^{\frac{\Delta g_2}{u_2}} \cdot \frac{\Delta g_1}{u_1} &= \lambda \end{aligned} \tag{34}$$

From setting Equations (33) and (34) equal we obtain,

$$\beta_1 = \frac{u_2\beta_2\Delta g_1}{u_1\Delta g_2}, \beta_2 = \frac{u_1\beta_1\Delta g_2}{u_2\Delta g_1}$$

Substituting in the original optimization inequality constraint,

$$\begin{aligned} \beta_2 + (\frac{u_2\beta_2\Delta g_1}{u_1\Delta g_2}) &= \beta \\ \beta_2 &= \frac{u_1\Delta g_2\beta}{u_1\Delta g_2 + u_2\Delta g_1} \end{aligned}$$

Similarly,

$$\begin{aligned} \beta_1 + (\frac{u_1\beta_1\Delta g_2}{u_2\Delta g_1}) &= \beta \\ \beta_1 &= \frac{u_2\Delta g_1\beta}{u_1\Delta g_2 + u_2\Delta g_1} \end{aligned}$$

□

A.2 Proofs of PROBE Generalization for n-Queries

Proof for Theorem 4.3. Given a complex decision support query $Q^{\Lambda,F}$ composed of n aggregate threshold queries connected by $n-1$ conjunctions/disjunctions, we achieve minimum privacy loss ϵ by budgeting the false negative bound β as:

$$\beta_j = \frac{\Delta g_j \beta \prod_{x=1}^{n,x\neq j}(u_x)}{\sum_{y=1}^{n}\prod_{x=1}^{n,x\neq y}(u_x\Delta g_y)}, \forall j = \{1,2,...,n\} \tag{35}$$

Proof. We will consider exclusive conjunction for this proof because the false negative rate equation is identical for both 2-way conjunction and disjunction. Consider the conjunction of n aggregate threshold queries $Q_1, Q_2, ..., Q_n$, where Q_j is defined as $Q^{\Lambda,f_j}_{\mathrm{g_j}(.)>C_j}(D) = \{\lambda_i \in \Lambda \,|\, \mathrm{g_j}(D^{f_j}_{\lambda_i}) > c_{j_i}\}$. All Q_j have the same predicates $\Lambda = \lambda_i, \lambda_2, ..., \lambda_k$ but different filters f_j, aggregate function $g_j(.)$ and threshold $C_j = c_{j_1}, ..., c_{j_k}$. Let mechanism $M_i : \mathcal{D} \rightarrow O_i$ satisfy a β_i-bound on FNR for aggregate threshold query Q_i. We can answer query Q which is a conjunction of n aggregate threshold queries using mechanism M where $M(D) = M_1(D) \cap M_2(D) \cap ... \cap M_n(D)$. First, we formally define M's overall false negative rate bound for exclusive conjunction.

We can evaluate mechanism M by sequentially evaluating the 2-way conjunction of each pair of sub-mechanisms (M_i, M_{i+1}). Let $M_{i\ i+1}$ be the resulting conjunction mechanism. For an even number of queries n we obtain:

$$M(D) = M_{12}(D) \cap M_{34}(D) \cap ... \cap M_{n-1\,n}$$

Each mechanism $M_{i\,i+1}$ has the resulting FNR bound of $\beta_i + \beta_{i+1}$ according to Theorem 4.2, e.g. $M_{1,2}$ has an FNR bound of $\beta_{12} = \beta_1 + \beta_2$. We subsequently run the 2-way conjunction mechanism on every pair $(M_{i\,i+1}, M_{i+2\,i+3})$. We obtain:

$$M(D) = M_{1234}(D) \cap ... \cap M_{n-3\,n-2\,n-1\,n}(D)$$

Where each mechanism $M_{i\,i+1\,i+2\,i+3}$ will have the FNR bound $\beta_{i\,i+1} + \beta_{i+2\,i+3}$ e.g. $\beta_{1234} = \beta_{12} + \beta_{34} = \beta_1 + \beta_2 + \beta_3 + \beta_4$. We can thus recursively run the 2-way conjunction mechanism in this subsequent manner to obtain:

$$\beta_1 + \beta_2 + ... + \beta_n \leq \beta \tag{36}$$

For an odd number of atomic queries n we similarly recursively group sub-mechanisms in pairs except for the last sub-mechanism M_n, which will be paired last with the mechanism composed of n_1 sub-mechanisms. We thus obtain the same result.

Second, we define mechanism M's overall privacy loss ϵ in terms of n-bounds $\beta_1, \beta_2, ...\beta_n$. For conjunction and disjunction alike, we use the sequential composition theorem (Def. 2.2) to compute the final privacy loss. Formally,

$$\begin{aligned} \epsilon &= \epsilon_1 + \epsilon_2 + ... + \epsilon_n \\ &= \Delta g_1 \frac{\ln(1/(2\beta_1))}{u_1} + \Delta g_2 \frac{\ln(1/(2\beta_2))}{u_2} + ... + \Delta g_n \frac{\ln(1/(2\beta_n))}{u_n} \\ &= \ln(1/2\beta_1)^{\frac{\Delta g_1}{u_1}} + \ln(1/2\beta_2)^{\frac{\Delta g_2}{u_2}} + ... + \ln(1/2\beta_n)^{\frac{\Delta g_n}{u_n}} \\ &= -\ln(2\beta_1)^{\frac{\Delta g_1}{u_1}} - \ln(2\beta_2)^{\frac{\Delta g_n}{u_2}} - ... - \ln(2\beta_n)^{\frac{\Delta g_n}{u_n}} \\ &= -\ln((2\beta_1)^{\frac{\Delta g_1}{u_1}}(2\beta_2)^{\frac{\Delta g_2}{u_2}} ... (2\beta_n)^{\frac{\Delta g_n}{u_n}}) \end{aligned}$$

To minimize ϵ we thus need to maximize:

$$f_\beta(\beta_1, \beta_2, ..., \beta_n) = (\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}} ... (\beta_n)^{\frac{\Delta g_n}{u_n}} \tag{37}$$

We thus use the Lagrange method to solve the following optimization problem:

$$\begin{aligned} &\text{maximize} \quad f_\beta(\beta_1, \beta_2, ..., \beta_n) = (\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}} ... (\beta_n)^{\frac{\Delta g_n}{u_n}} \\ &\text{subject to} \quad \beta_1 + \beta_2 + ... + \beta_n \leq \beta \end{aligned} \tag{38}$$

Applying the Lagrangian function to our optimization problem,

we obtain n-functions:

$$\frac{d}{d\beta_1}((\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}}...(\beta_n)^{\frac{\Delta g_n}{u_n}}) = \lambda\frac{d}{d\beta_1}(\beta_1+\beta_2+...+\beta_n)$$
$$\frac{d}{d\beta_2}((\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}}...(\beta_n)^{\frac{\Delta g_n}{u_n}}) = \lambda\frac{d}{d\beta_2}(\beta_1+\beta_2+...+\beta_n)$$
$$\vdots$$
$$\frac{d}{d\beta_n}((\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}}...(\beta_n)^{\frac{\Delta g_n}{u_n}}) = \lambda\frac{d}{d\beta_n}(\beta_1+\beta_2+...+\beta_n)$$

Solving the equations results in,

$$\begin{aligned}\frac{\Delta g_1}{u_1}(\beta_1)^{\frac{\Delta g_1}{u_1}-1}(\beta_2)^{\frac{\Delta g_2}{u_2}}...(\beta_n)^{\frac{\Delta g_n}{u_n}} &= \lambda\\ \frac{\Delta g_2}{u_2}(\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}-1}...(\beta_n)^{\frac{\Delta g_n}{u_n}} &= \lambda\\ \vdots\\ \frac{\Delta g_n}{u_n}(\beta_1)^{\frac{\Delta g_1}{u_1}}(\beta_2)^{\frac{\Delta g_2}{u_2}}...(\beta_n)^{\frac{\Delta g_n}{u_n}-1} &= \lambda\end{aligned}$$

By setting every combination of equations equal, we obtain:

$$\frac{\Delta g_1}{\beta_1 u_1} = \frac{\Delta g_2}{\beta_2 u_2} = ... = \frac{\Delta g_n}{\beta_n u_n}$$

Solving for individual β_i,

$$\begin{aligned}\beta_1 &= \frac{\beta_2 u_2\Delta g_1}{u_1\Delta g_2} = \frac{\beta_3 u_3\Delta g_1}{u_1\Delta g_3} = ... = \frac{\beta_n u_n\Delta g_1}{u_1\Delta g_n}\\ \beta_2 &= \frac{\beta_1 u_1\Delta g_2}{u_2\Delta g_1} = \frac{\beta_3 u_3\Delta g_2}{u_2\Delta g_3} = ... = \frac{\beta_n u_n\Delta g_2}{u_2\Delta g_n}\\ &\vdots\\ \beta_n &= \frac{\beta_1 u_1\Delta g_n}{u_n\Delta g_1} = \frac{\beta_2 u_2\Delta g_n}{u_n\Delta g_2} = ... = \frac{\beta_{n-1} u_{n-1}\Delta g_n}{u_n\Delta g_{n-1}}\end{aligned}$$

We can now substitute $\beta_{j\neq i}$ in the original constraint to solve for individual β_i:

$$\beta = \beta_1 + \frac{\beta_1 u_1\Delta g_2}{u_2\Delta g_1} + ... + \frac{\beta_1 u_1\Delta g_n}{u_n\Delta g_1}$$

For β_1 we obtain:

$$\begin{aligned}\beta_1 &= \frac{\beta}{(1+\frac{u_1}{\Delta g_1}(\frac{\Delta g_2}{u_2}+...+\frac{\Delta g_n}{u_n}))}\\ &= \frac{\beta}{(1+\frac{u_1}{\Delta g_1}(\frac{\Delta g_2 u_3u_4...u_n+\Delta g_3u_2u_4...u_n+...+\Delta g_nu_2u_3...u_{n-1}}{u_2u_3...u_n}))}\\ &= \frac{\beta}{(1+\frac{u_1}{\Delta g_1}(\frac{\sum_{i=1}^{n,i\neq1}\prod_{j=1}^{n,j\neq i,1}u_j\Delta g_i}{\prod_{i=1}^{n,i\neq1}u_i}))}\\ &= \frac{\Delta g_1\beta\prod_{i=1}^{n,i\neq1}u_i}{\Delta g_1\prod_{i=1}^{n,i\neq1}u_i + u_1\sum_{i=1}^{n,i\neq1}\prod_{j=1}^{n,j\neq i,1}u_j\Delta g_i}\\ &= \frac{\Delta g_1\beta\prod_{i=1}^{n,i\neq1}u_i}{\sum_{i=1}^{n}\prod_{j=1}^{n,j\neq i}u_j\Delta g_i}\end{aligned}$$

Similarly,

$$\beta_2 = \frac{\Delta g_2\beta\prod_{i=1}^{n,i\neq2}u_i}{\sum_{i=1}^{n}\prod_{j=1}^{n,j\neq i}u_j\Delta g_i}, ..., \beta_n = \frac{\Delta g_n\beta\prod_{i=1}^{n,i\neq n}u_i}{\sum_{i=1}^{n}\prod_{j=1}^{n,j\neq i}u_j\Delta g_i}$$

Thus, we can generalize the expression for $i \in 1, 2, ..., n$

$$\beta_i = \frac{\Delta g_i\beta\prod_{x=1}^{n,x\neq i}(u_x)}{\sum_{y=1}^{n}\prod_{x=1}^{n,x\neq y}(u_x\Delta g_y)}, \forall i = \{1, 2, ..., n\} \tag{39}$$

□

Proof for Theorem 4.5. Given a complex decision support query $Q^{\Lambda,F}$ with query tree T composed of n aggregate threshold queries with an associated o_i number of occurrences within the tree, we achieve minimum privacy loss ϵ by budgeting the β-bound on FNR as:

$$\beta_i = \frac{\Delta g_i\beta\prod_{x=1}^{n,x\neq i}(u_x)}{\sum_{y=1}^{n}\prod_{x=1}^{n,x\neq y}(u_xo_y\Delta g_y)}, \forall i = \{1, 2, ..., n\} \tag{40}$$

Proof. We first prove that the apportioning function f_β that optimally apportions the FNR bound β over the sub-queries given a query tree T_c is:

$$f_\beta(\beta_1, \beta_2, ..., \beta_n) = \sum_{i=1}^{n} o_i\beta_i \leq \beta \tag{41}$$

We prove this by induction as follows:
Base case: *Query tree with two nodes.* consider a complex query $Q^{\Lambda,F}$ with compact query tree T composed of $n = 2$ leaf nodes containing two sub-queries Q_1, Q_2 with occurrences $o_1 = o_2 = 1$ connected by an operator $\cap$ or $\cup$. It follows from Appendix A.2 that the apportioning of the FNR in terms of β_i is:

$$\begin{aligned}f_\beta(\beta_1, \beta_2) &= \beta_1 + \beta_2 \leq \beta\\ f_{\beta_1,\beta_2} &= o_1\beta_1 + o_2\beta_2 \leq \beta\end{aligned}$$

Therefore condition (41) holds for the base case of $n = 2$.
Induction step: Suppose that for a query tree T of size $n = k$ nodes containing m sub-queries $Q_1, Q_2, ..., Q_m$ with occurrences $o_1, o_2, ..., o_m$, the FNR is apportioned in terms of β_i and bound by β as:

$$f_\beta(\beta_1, \beta_2, ..., \beta_m) = \sum_{i=1}^{m} o_i\beta_i \leq \beta \tag{42}$$

Now we show that the condition holds for $n = k + 1$. Let T be a compact query tree composed of $n = k + 1$ nodes containing m sub-queries $Q_1, Q_2, ..., Q_m$ with occurrences $o_1, o_2, ..., o_m$ connected by an operator $\cap$ or $\cup$. We refer to the right sub-tree as T_R and the left sub-tree as T_L. Each sub-tree contains the sub-queries $Q_1, Q_2, ..., Q_m$ with local occurrences $T_L.o_i$ or $T_R.o_i$ which add up to the overall occurrence number in the tree $T.o_i$, i.e. $\forall i \in m$

$$T.o_i = T_L.o_i + T_R.o_i \tag{43}$$

For the right sub-tree T_R, we know that the condition (42) holds for $n = k$. Therefore the FNR for T_R in terms of β_i is:

$$\beta_{T_R} = \sum_{i=1}^{m} T_R.o_i\beta_i \tag{44}$$

Similarly for the left sub-tree T_L, we obtain the FNR,

$$\beta_{T_L} = \sum_{i=1}^{m} T_L.o_i\beta_i \tag{45}$$

For overall tree T, we know from 2-way conjunction/disjunction that:

$$f_{\beta_T}(\beta_1, \beta_2, ..., \beta_m) = \beta_{T_L} + \beta_{T_R} \leq \beta$$

Substituting in Eq. (44) and (45) we obtain:

$$\begin{aligned} f_{\beta_T}(\beta_1, \beta_2, ..., \beta_m) &= \sum_{i=1}^{m} T_L.o_i\beta_i + \sum_{i=1}^{m} T_R.o_i\beta_i \leq \beta \\ &= \sum_{i=1}^{m} T_L.o_i\beta_i + T_R.o_i\beta_i \leq \beta \end{aligned}$$

From Eq. (43) we obtain

$$f_{\beta T}(\beta_1, \beta_2, ..., \beta_m) = \sum_{i=1}^{m} T.o_i\beta_i \leq \beta$$

Therefore Eq. (42) holds for all query trees T with m atomic aggregate threshold queries.

Second, we derive the apportionment of β_i from Eq. (40). To do so, we derive our optimization problem which minimizes ϵ given the constraint from Eq. (41). Given that we run each sub-query Q_i using M_i based on its number of occurrences o_i in query tree T, its privacy budget will be proportional to the number of occurrences, i.e. the loss will be $o_i\epsilon_i$ for each Q_i. This means that the overall privacy loss, when using the Sequential Composition theorem, will be $\epsilon = \sum_{i=1}^{n} o_i\epsilon_i$. As we use TSLM for the sub-mechanisms M_i, we can use the formulation of $\epsilon_i = \Delta g_i \frac{\ln(1/(2\beta_i))}{u_i}$. Substituting in:

$$\begin{aligned} \epsilon &= \sum_{i=1}^{n} o_i\epsilon_i \\ &= \sum_{i=1}^{n} o_i\Delta g_i \frac{\ln(1/(2\beta_i))}{u_i} \\ &= -\sum_{i=1}^{n} \ln(2\beta_i)^{\frac{o_i\Delta g_i}{u_i}} \\ &= -\ln((2\beta_1)^{\frac{o_1\Delta g_1}{u_1}} (2\beta_2)^{\frac{o_2\Delta g_2}{u_2}} ... (2\beta_n)^{\frac{o_n\Delta g_n}{u_n}}) \end{aligned}$$

To minimize ϵ we thus need to maximize:

$$f_\beta(\beta_1, \beta_2, ..., \beta_n) = \prod_{i=1}^{n} (\beta_i)^{\frac{o_i\Delta g_i}{u_i}} \tag{46}$$

So our optimization problem is updated to:

$$\begin{aligned} &\text{maximize} \quad \prod_{i=1}^{n} (\beta_i)^{\frac{o_i\Delta g_i}{u_i}} \\ &\text{subject to} \quad \sum_{i=1}^{n} o_i\beta_i \leq \beta \end{aligned} \tag{47}$$

We thus apply the Lagrange Multipliers Method as with the previous proof to obtain:

$$\beta_i = \frac{\Delta g_i \beta \prod_{x=1}^{n, x\neq i}(u_x)}{\sum_{y=1}^{n} \prod_{x=1}^{n, x\neq y}(u_x o_y \Delta g_y)}, \forall i = \{1, 2, ..., n\} \tag{48}$$

□

A.3 Proofs For FPR Bounds

1. Proof For 2-Query Conjunction. Consider a complex decision support query Q composed of two atomic aggregate threshold queries $Q = Q_1 \cap Q_2$. Q_1 and Q_2 are answered by TSLM M_1, M_2 respectively, and mechanism $M = M_1 \cap M_2$ is used to answer query Q. If M_1 and M_2 guarantee an FPR bound of α_1 and α_2 respectively, then M has an FPR bound of $\alpha_1 + \alpha_2$. If we set $\alpha_i = \alpha/n$, we obtain a α bound on FPR.

Proof. $\forall \lambda_i \in \Lambda$,
Let A be the event that $\lambda_i \in M_1(D)$, B be the event that $\lambda_i \in M_2(D)$, C be the event that $\lambda_i \in Q_1(D)$ and D be the event that $\lambda_i \in Q_2(D)$. The probability of false positives can be derived as:

$$\begin{aligned} FPR &= P[\lambda_i \in M_1(D) \cap M_2(D) | \lambda_i \notin Q_1(D) \cap Q_2(D)] \\ &= P[AB|\overline{CD}] = P[AB|\overline{C} \vee \overline{D}] = P[AB|\overline{C}D \vee C\overline{D} \vee \overline{CD}] \\ &= \frac{P[AB \wedge \overline{C}D \vee C\overline{D} \vee \overline{CD}]}{P[CD]} \\ &= \frac{P[(AB \wedge \overline{C}D) \vee (AB \wedge C\overline{D}) \vee (AB \wedge \overline{CD})]}{P[CD]} \end{aligned} \tag{49}$$

Given that the three clauses are mutually exclusive events

$$\begin{aligned} &= \frac{P[AB \wedge \overline{C}D] + P[AB \wedge C\overline{D}] + P[AB \wedge \overline{CD}]}{P[CD]} \\ &= \frac{P[AB|\bar{C}D]P[\bar{C}D]}{P[CD]} + \frac{P[AB|C\bar{D}]P[C\bar{D}]}{P[CD]} \\ &+ \frac{P[AB|\overline{CD}]P[\overline{CD}]}{P[CD]} \\ &\leq P[AB|\bar{C}D] + P[AB|C\bar{D}] + P[AB|\overline{CD}] \end{aligned}$$

We know that M_1 and M_2 are mechanisms of independent randomness, i.e. add random noise. This means that A is independent of B given CD and vice versa. Therefore we can rewrite $P[AB|\overline{C}D] = P[A|\overline{C}D]P[B|\overline{C}D]$. We obtain:

$$= P[A|\overline{C}D]P[B|\overline{C}D] + P[A|C\overline{D}]P[B|C\overline{D}] + P[A|\overline{CD}]P[B|\overline{CD}]$$

There is a conditional independence between A and D given C due to the nature of the randomized mechanisms. Similarly there is a conditional independence between B and C given D. Thus, we can rewrite $P[A|\overline{C}D]P[B|\overline{C}D] = P[A|\overline{C}]P[B|D]$. Therefore,

$$= P[\overline{A}|C]P[B|D] + P[A|C]P[\overline{B}|\overline{D}] + P[A|\overline{C}]P[B|\overline{D}]$$

Knowing that $FPR_1 = P[A|\overline{C}]$, $FPR_2 = P[B|\overline{D}]$, $TPR_1 = P[A|C]$ and $TPR_2 = P[B|D]$, we substitute:

$$\begin{aligned} &= FPR_1 \cdot TPR_2 + TPR_1 \cdot FPR_2 + FPR_1 \cdot FPR_2 \\ &= FPR_1 \cdot TPR_2 + FPR_2(FPR_1 + TPR_1) \\ &\leq FPR_1 + FPR_2 \leq \alpha_1 + \alpha_2 \end{aligned} \tag{50}$$

So if we set $\alpha_1 = \alpha_2 = \alpha/2$, then $FPR \leq \alpha$. □

2. Proof For 2-Query Disjunction. Consider a complex decision support query Q composed of two atomic aggregate threshold queries $Q = Q_1 \cup Q_2$. Q_1 and Q_2 are answered by TSLM M_1, M_2 respectively, and mechanism $M = M_1 \cup M_2$ is used to answer query Q. If M_1 and M_2 guarantee an FPR bound of α_1 and α_2 respectively, then M has an FPR bound of $\alpha_1 + \alpha_2$. If we set $\alpha_i = \alpha/n$, we obtain a α bound on FPR.

Proof. $\forall \lambda_i \in \Lambda$,
Let A be the event that $\lambda_i \in M_1(D)$, B be the event that $\lambda_i \in M_2(D)$, C be the event that $\lambda_i \in Q_1(D)$ and D be the event that $\lambda_i \in Q_2(D)$. The probability of false positives can be derived as:

$$\begin{aligned} FPR &= P[\lambda_i \in M_1(D) \cap M_2(D) | \lambda_i \notin Q_1(D) \cap Q_2(D)] \\ &= P[A \vee B | \overline{C \vee D}] \\ &= \frac{P[(\overline{A}B \vee A\overline{B} \vee AB) \wedge \overline{CD}]}{P[\overline{C \vee D}]} \end{aligned}$$

Given that the three clauses are mutually exclusive events

$$\begin{aligned} &= \frac{P[\overline{A}B \wedge \overline{CD}] + P[A\overline{B} \wedge \overline{CD}] + P[AB \wedge \overline{CD}]}{P[\overline{C \vee D}]} \\ &= P[\overline{A}B|\overline{CD}] + P[A\overline{B}|\overline{CD}] + P[AB|\overline{CD}] \end{aligned}$$

We know that M_1 and M_2 are mechanisms of independent randomness, i.e. add random noise. This means that A is independent of B given CD and vice versa. Therefore we can rewrite $P[\overline{A}B|\overline{CD}] = P[\overline{A}|\overline{CD}]P[B|\overline{CD}]$. We obtain:

$$= P[\overline{A}|\overline{CD}]P[B|\overline{CD}] + P[A|\overline{CD}]P[\overline{B}|\overline{CD}] + P[A|CD]P[B|CD]$$

There is a conditional independence between A and D given C due to the nature of the randomized mechanisms. Similarly there is a conditional independence between B and C given D. Thus, we can rewrite $P[\overline{A}|\overline{CD}]P[B|\overline{CD}] = P[\overline{A}|\overline{C}]P[B|\overline{D}]$. Therefore,

$$= P[\overline{A}|\overline{C}]P[B|\overline{D}] + P[A|\overline{C}]P[\overline{B}|\overline{D}] + P[A|\overline{C}]P[B|\overline{D}]$$

Knowing that $FPR_1 = P[A|\overline{C}]$, $FPR_2 = P[B|\overline{D}]$, $TNR_1 = P[\overline{A}|\overline{C}]$ and $TNR_2 = P[\overline{B}|\overline{D}]$, we substitute:

$$\begin{aligned} &= TNR_1 \cdot FPR_2 + FPR_1 \cdot TNR_2 + FPR_1 \cdot FPR_2 \\ &= (1 - FPR_1)FPR_2 + FPR_1(1 - FPR_2) + FPR_1FPR_2 \\ &\leq FPR_1 + FPR_2 - FPR_1FPR_2 \leq FPR_1 + FPR_2 \leq \alpha_1 + \alpha_2 \end{aligned} \quad (51)$$

So if we set $\alpha_1 = \alpha_2 = \alpha/2$, then $FPR \leq \alpha$. □

Generalization of FPR bound. By using a similar proof by induction as in Appendix A.2 (Proof for Theorem 4.5), we obtain the bound on FPR given any query T, where each sub-query Q_i of n sub-queries, with an associated o_i number of occurrences, as follows:

$$FPR \leq \sum_{i=1}^{n} o_i \alpha_i$$

If we set $\alpha_i = \alpha/2o_i$ for each sub-query, then $FPR \leq \alpha$.

A.4 Proof of ProBE Algorithm

Proof for Theorem 5.1. Algorithm 2 satisfies $\epsilon_{\max}$-DP, a β-bound on the False Negative Rate and a α bound on the False Positive Rate.

Proof. First, we show the proof that TSLM offers the guarantee that for an atomic aggregate threshold query Q_{a_i}, setting the privacy budget to $\epsilon_i = \frac{\Delta g_i \ln(1/(2\beta_i))}{u_i}$ guarantees the FNR is bounded by β_i as defined in [24]. For all predicates $\lambda_i \in \Lambda$:

$$\begin{aligned} & P[\lambda_i \notin M(D) | \lambda_i \in Q(D)] \\ = \; & P[g(D^f_{\lambda_i}) + \eta_i \leq c_i - u_i | g(D^f_{\lambda_i}) > c_i] \\ \leq \; & P[\eta_i \leq -u_i] \leq \frac{e^{-ln(1/2\beta)}}{2} \leq \beta \end{aligned}$$

Second, we prove that setting β per iteration to $\beta/2$ upholds a β bound on the overall mechanism's FNR.

We know that the FNR is bounded as $FNR \leq \sum_{i=1}^{n} \beta_i$. Therefore, if we set $\beta_{step} = \beta_2$ where $step = i \in \{1, 2\}$, then we obtain $FNR \leq \beta/2 + \beta/2 = \beta$. Therefore the FNR is bounded by β.

Third, the algorithm satisfies ϵ_{max}-DP.

(i) Phase One of ProBE adds noise from the Laplace Distribution according to the Laplace Mechanism[19], i.e. with a mean of 0 and a standard deviation of $1/\epsilon_i$. This means that Phase one is ϵ_i-differentially private. (ii) Phase Two of ProBE similarly adds noise from the Laplace Distribution according to the Laplace Mechanism with a newly computed ϵ_j privacy budget, making phase two ϵ_j-differentially private. (iii) Because the two phases are executed sequentially, their composition (i.e. the overall ProBE mechanism) is $\epsilon_i + \epsilon_j$-differentially private. (iv) It follows that ProBE satisfies ϵ_{max}-DP as it checks that the current privacy budget (i.e. $\epsilon_i + \epsilon_j$) does not exceed the limit of ϵ_{max}.

Fourth, the algorithm satisfies the α-bound on FPR because we estimate an upper bound on the FPR by deriving an upper bound on FP and a lower bound on N. Thus, since $FPR \leq FPR_{est}$, then enforcing a bound on the upper bound $FPR_{est} \leq \alpha$ (by exiting the algorithm if it is exceeded) will enforce a bound on the actual FPR. i.e. $FPR \leq FPR_{est} \leq \alpha \Rightarrow FPR \leq \alpha$.

□

B PROBE COMPUTATIONAL COMPLEXITY ANALYSIS

We analyze the computational complexity of ProBE as it relates to regular SQL queries by using the example query below:

```
SELECT disease, count(*) FROM PATIENT_DATA
WHERE disease_type ='viral'
GROUP BY disease
HAVING (count(*) > c1 AND avg(age) > 65)
```

If executed over an SQL system such as MySQL, the query evaluation plan would be:

(1) Full Table Scan on *PATIENT_DATA*
(2) GROUP BY disease attribute
(3) Compute aggregations on $count(*)$ and on avg(age) based on the tuples in a group per group and filter matching groups based on the count(*) > c1 and avg(age) > 65 conditions
(4) Return matching groups

In ProBE, if we break the query into sub-queries and execute them separately then apply the conjunction or disjunction as discussed in the paper, the approach will correspond to the following plan:

For Q1:

(1) Full Table Scan on *PATIENT_DATA*
(2) GROUP BY disease attribute
(3) Compute aggregations with noise on count(*) based on the tuples in a group per group and filter matching groups based on count(*) > c1 condition

For Q2:

(1) Full Table Scan on *PATIENT_DATA*

(2) GROUP BY disease attribute
(3) Compute aggregations with noise on avg(age) based on the tuples in a group per group and filter matching groups based on avg(age) > 65 condition

And lastly:

(1) Compute conjunction of Q1 and Q2
(2) Return matching groups

In the execution plan above, if we have k operators (i.e. conjunctions and/or disjunctions), the technique will result in a computational complexity of almost k times. Note that when we scan for each of the sub-queries, we do not need to do this scan independently. Indeed, a better plan would be:

(1) Full Table Scan on *PATIENT_DATA*
(2) GROUP BY disease attribute
(3) Compute all aggregations with noise i.e. count(*) and avg(age) and filter matching groups based on the count(*) > c1 and avg(age) >65 conditions
(4) Compute conjunction of Q1 and Q2
(5) Return matching groups

The above execution plan would meet our requirement, and has complexity similar to the original SQL plan, but will require an in-database implementation to be slightly modified to achieve the above. Our goal in this paper is primarily to develop the foundation for answering complex queries in a differentially private manner and not on optimizing the query performance in terms of execution time. Nonetheless, this aspect of optimizing the execution of our algorithm is a very interesting direction of future work.

Note that implementation of the GROUP BY operation has to be special in that all possible groups to be returned (i.e. all the diseases) must be pre-determined, as opposed to the SQL GROUP BY implementation which does not have knowledge of the existing groups. This issue applies to both second and third plan.

C DETAILED MULTI-STEP ENTROPY-BASED ALGORITHM

The two-step algorithm ProBE assigns different privacy levels to different predicates due to the early elimination of data points at the first step, meaning predicates that go on to the second step have a higher privacy loss ϵ. This concept is captured through the definition of Predicate-wise DP (PWDP)[24], a fine-grained extension of differential privacy which quantifies the different levels of privacy loss data points may have in multi-step algorithms. Formally,

Definition C.1 (Predicate-wise Differential Privacy (PWDP)). Given a set of mutually exclusive predicates and their corresponding privacy budgets $\Theta = \{(\lambda_1, \epsilon_1), ..., (\lambda_k, \epsilon_k)\}$, we say a randomized mechanism M satisfies θ-PWDP if for all i, for any neighboring databases D and D' differing in at most one record that satisfies λ_i, the following holds:

$$P[M(D) \in O] \leq e^{\epsilon_i} P[M(D') \in O] \tag{52}$$

Naturally, a measure to quantify this new definition of privacy is needed Previous work tackles this problem by proposing a new privacy metric for PWDP entitled *Min-Entropy* [24] $\mathcal{H}_{min}$ which measures a lower bound on the level of uncertainty given the set of predicates and their respective privacy levels $\Theta = \{(\lambda_1, \epsilon_1), ..., (\lambda_k, \epsilon_k)\}$. Formally,

Definition C.2 (PWDP Min-Entropy). The Min-Entropy of a Θ-predicate-wise differentially private mechanism with $\Theta = \{(\lambda_1, \epsilon_1), ..., (\lambda_k, \epsilon_k)\}$ is defined as:

$$\mathcal{H}_{min}(\Theta) = min\ \Sigma_{i=1}^{k} - \hat{p_i} \log \hat{p_i}$$
$$s.t. \frac{e^{-\epsilon_i}}{\Sigma_i e^{\epsilon_i}} \leq \hat{p_i} \leq \frac{e^{\epsilon_i}}{\Sigma_i e^{-\epsilon_i}}\ \forall i \in [1, k]\ and\ \Sigma_i \hat{p_i} = 1 \tag{53}$$

where $\hat{p_i}$ is the probability that a random tuple x will take a value t that satisfies λ_i.

the Data Dependent Predicate-Wise Laplace Mechanism (DDPWLM) also introduced in [24] maximizes min-entropy (i.e. maximizing the lower bound on uncertainty) in a multi-step algorithm by also using the noisy aggregate values obtained from previous iterations to compute the best privacy level ϵ at which Min-Entropy is maximized for the set of elements in the uncertain region. This algorithm similarly guarantees a β-bound on FNR for a single aggregate threshold query by setting the privacy loss to $\epsilon = \frac{\Delta g \ln(1/(2\beta))}{u}$, but it does so by setting a starting privacy level ϵ_s as well as a maximum level ϵ_m prior to execution which is spent across a fixed number of iterations m in a way that maximizes min-entropy. In each iteration, the privacy budget is further distributed across fine-grained steps m_f and the privacy level with the highest min-entropy is chosen as the next iteration's budget. It follows that the uncertain region parameter u is, again, chosen statically at the beginning, then chosen to minimize privacy loss rather than in a way that can provide a bound on false positives.

We modify this algorithm by integrating our two-step algorithm ProBE into its framework in order to not only answer complex DS queries, but also to provide a post-facto bound on false positives. Our modified algorithm is entitled ProBE-Ent as described in Algorithm 5. Instead of having a fixed starting ϵ_s, we internally choose a starting u_0 and compute the initial privacy level (line 7). Within the first step of DDPWLM, we run the FP estimation algorithm (Function in Algorithm 3) in order to compute the optimal uncertain region parameter u_{opt}. We use this new uncertain region as an upper bound for the algorithm, i.e. we compute the ϵ_m upper bound that represents the exit condition for the algorithm. Additionally, we provide a β budget optimization which exploits the multi-step nature of the algorithm in a way that provides a potentially higher budget if previous sub-queries exit early.

β-Budget Redistribution. In the case of multi-step algorithms, we exploit their iterative nature to further optimize β budget allocation during execution. Given that certain queries may terminate early (if an iteration $j < m$ yields no predicates within the uncertain region), the assigned β_i for such a query may not be fully used and may thus be wasted. We exploit the progressive nature of such algorithms to further optimize the privacy budget: if the previous sub-query terminates before the m-th step of its run, we can redistribute the leftover false negative rate budget to subsequent sub-queries, thus fully utilizing the false negative rate bound given and consequently minimizing the overall privacy loss. For each sub-query, we calculate the remaining β budget after running the MSPWLM algorithm by subtracting the used β_i budget from the overall β (line 9). For subsequent sub-queries, we check if the remaining β budget β_{rem} from the previous iteration is higher than 0. If this is not the case,

Algorithm 5 ProBE Entropy-based Mechanism. $Q = \{Q_1, ..., Q_n\}$, $u = \{u_1, ..., u_n\}$, $o = \{o_1, ..., o_n\}$, $\Delta g = \{\Delta g_1, ..., \Delta g_n\}$, $t = \{t_1, ..., t_{n-1}\}$ where $t_i = 0$ if conjunction, 1 if disjunction

1: **procedure** EntProbe($Q, D, u, \beta, o, \Delta g, \epsilon_{max}, m, m_f, t$)
2: Let $O_f \leftarrow \{\}, \epsilon_f \leftarrow 0, \beta_{rem} \leftarrow 0$
3: **for** $i = 1, ..., n$ **do**
4: **if** $\beta_{rem} > 0$ **then**
5: update overall FNR budget $\beta \leftarrow \beta_{rem}$
6: $Q \leftarrow \{Q_i, ..., Q_n\}$
7: $\beta_i \leftarrow \frac{\Delta g_i \beta \prod_{x=1}^{n, x\neq i}(u_0)}{o_i \sum_{y=1}^{n} \prod_{x=1}^{n, x\neq y}(u_0 \Delta g_y)}, \epsilon_s = \frac{\Delta g_i \ln(1/(2\beta_i/m))}{u_0}$
8: $O_i, \epsilon_i, \beta_{used} \leftarrow$ Ddpwlm($Q_i, u_i, \beta_i, \epsilon_{max}, \epsilon_s, m, m_f$)
9: $\beta_{rem} \leftarrow 0$ if $\beta_i = \beta_{used}$ **else** $\beta - \beta_{used}$
10: $\epsilon_f \leftarrow \epsilon_f + \epsilon_i$
11: **if** $\epsilon_f \leq \epsilon_{max}$ **then**
12: **if** $type = 0$ **then**
13: $O_f \leftarrow O_i$ if $O_f = \emptyset$ **else** $O_f \leftarrow O_f \cap O_i$
14: **return** O_f, ϵ_f if $O_i = \emptyset$
15: **else if** $type = 1$ **then**
16: $O_f \leftarrow O_f \cup O_i$
17: **else**
18: **return** 'Query Denied'
19: **return** O_f, ϵ_f

then we use the ProBE optimization with the original β and all the sub-queries. Otherwise, we update the overall β to the remaining β_{rem} and re-run the ProBE optimization using only the subsequent sub-queries $Q_i, \ldots, Q_n$ (lines 4-6). To illustrate this algorithm, we use the example below.

Example C.1. Consider a 3-step PPWLM run with a conjunction query composed of 4 aggregate threshold queries; if the first query finishes executing after its 1st step, its leftover overall false negative rate will be $\beta - \Delta\beta_1$, where $\Delta\beta_1 = \frac{1}{3}\beta_1$. Therefore we re-compute the false negative rates for subsequent queries $\beta_2, \beta_3, \beta_4$ using Eq. (25) with $\beta' = \beta - \Delta\beta_1$ and with the Q_2, Q_3, Q_4 leftover queries.

D DETAILED RELATED WORK

Differential Privacy [18, 19] has become a well-studied standard for privacy-preserving data exploration and analysis [12, 27, 48, 51]. Various work has proposed frameworks for answering specific query types, such as range queries [16, 30, 54], and linear counting queries [32, 38, 39]. This body of work, however, is not applicable to the aggregate threshold queries that our solution considers, nor does it take the approach of accuracy-first, which aims to minimize privacy loss given an accuracy constraint. Join queries may be more applicable to such queries, as conjunction-only queries may be written as join queries. However, such an approach complicates the queries at hand: the join operator is known for having high sensitivity resulting in low accuracy when answering such queries. Additionally, recent algorithms tackling join queries [11, 13, 28, 52] do not provide accuracy guarantees in terms of FNR or FPR, and their data-dependent nature make it difficult to successfully enforce error bounds.

Multiple accuracy-constrained systems for differentially private data analysis have been proposed in recent years [23, 33, 34, 36], which allow data analysts to interactively specify accuracy requirements over their queries while providing a formal privacy guarantee. However, these solutions do not specifically focus on decision support queries and do not take into account their specific accuracy requirements. CacheDP [36] proposes a query engine that uses previous answers stored in a differentially private cache to lower the overall privacy budget, but this framework does not support aggregate threshold queries, and hence is not applicable to our problem of complex DS queries. The system APEx [23], and programming framework DPella [34] both allow data analysts to provide accuracy bounds for differentially private query answering, and both support aggregate threshold queries (also referred to as iceberg counting queries), but neither support complex decision support queries which are compositions of atomic queries, nor do they incorporate the asymmetric utility characteristic (i.e. the importance of the false negative rate) that decision support queries have. Ligett et al. [33], which we extend in our work through the use of the ex-post DP notion, uses empirical error to determine the best privacy budget ϵ. Such an approach is not suitable for aggregate threshold queries, as the sensitivity of the error would result in a high privacy loss, and the testing for empirical error incurs an additional cost to privacy.

The problem of incorporating differential privacy in decision support has been tackled in MIDE [24], which our work extends. MIDE [24] makes use of the asymmetric utility feature in DS applications to formally define strict accuracy guarantees in terms of the false negative rate β, and proposes novel mechanisms to answer DS queries at a minimal privacy loss while upholding such guarantees. However, this work only addressed simple atomic queries, rather than complex DS queries composed of multiple aggregate statistics evaluated against their respective thresholds. Other work such as Fioretto et al. [22] also studies decision support on differentially private data, but does so from a fairness lens. It tackles aggregate threshold queries, but proposes solutions to mitigate bias resulting from making decisions on such data, rather than optimizing privacy loss or providing formal accuracy guarantees.

E LAGRANGE MULTIPLIERS METHOD

The Lagrange Multipliers method [4] is a optimization technique used to maximize or minimize multivariate functions subject to equality or inequality constraints. Given an objective function f, the goal is to find the extremum of such a function given a constraint function g. The intuition behind this method is that if f and g attain an extremum at x^*, then one of the level curves of f and g are tangent at x^*, meaning that their gradients ∇f and ∇g are parallel. This implies that the gradient of the constraint function ∇g is a multiple of the objective function gradient ∇f, and this multiple is referred to as a Lagrange Multiplier λ. The relationship between the gradients and the multiplier is depicted in the Lagrangian Function, which is the equation:

$$\nabla f(x) = \lambda \nabla g(x)$$

Solving this equation for the multiplier λ thus yields the optimal variable expression which can be substituted into the constraint function to obtain the corresponding extrema. Formally,

Theorem E.1 (Lagrange Multipliers Method). *Let $f : \mathbb{R}^d \rightarrow \mathbb{R}$ be the objective function, and $g : \mathbb{R}^d \rightarrow \mathbb{R}^n$ be the constraint function where both functions are C^1. If f attains a local extremum at*

x^* such that $rank(D(g(x^*)) = n$ given the constraint function, then there exists a unique multiplier λ such that:

$$\nabla f(x^*) = \lambda \nabla g(x^*) \tag{54}$$

F ADDITIONAL EXPERIMENTS

Min-Entropy Privacy Results. We extend our experiments to include Min-Entropy denoted $\mathcal{H}_{min}(\Theta)$ as part of our privacy metrics used to evaluate the performance of all algorithms. We use the same parameters as the main paper: we set a FNR bound of $\beta = 0.05$, a FPR bound of $\alpha = 0.1$, a maximum privacy loss of $\epsilon_{max} = 5$, a starting uncertain region of $u_0 = 30\%|D|$ for ProBE-based algorithms (the original ProBE algorithm, the ProBE-Naive variation with the Naive first step, and the ProBE-Ent entropy-based variation) and a default $u = 12\%$ for the Naive algorithm. We run each algorithm over a 100 iterations and vary the number of sub-queries from 1 to 6.

In terms of Min-entropy $\mathcal{H}_{min}(\Theta)$, the closer to 1 the results are (meaning that the lower bound on uncertainty is extremely high) the better. The results in Figure 9 thus show that ProBE-Ent outperforms other algorithms and stays constant or suffers only from a small decrease as the number of sub-queries increases. This directly follows from the fact that ProBE-Ent is designed to maximize min-entropy at each iteration, regardless of the number of sub-queries. Other algorithms suffer from a decrease in min-entropy as the number of sub-queries increase, as it is inversely correlated to the ex-post DP loss ϵ.

Varying β budget split across phases. We evaluate the privacy loss and accuracy measures of our proposed ProBE algorithm when varying the distribution of the overall β bound budget across the two phases of the algorithm. We use the following combinations of percentages: {10% − 90%, 90% − 10%, 50% − 50%, 40% − 60%, 60% − 40%, } where the first percentage is allocated to Phase One of ProBE and the second is allocated to Phase Two. We use a simple disjunction query $Q_1 \cup Q_2$ on the Taxi dataset with the fixed parameter of $\alpha = 0.1$ and vary the β value from the default $\beta = 5 \cdot 10^{-2}$ to a much smaller value of $5 \cdot 10^{-6}$. Our results are shown in Table 3. The different distributions used for the β budget do not significantly impact the privacy loss ϵ. There is no direct correlation observed between privacy loss and the β distribution across the two phases, as we observe that although the lowest privacy loss achieved is when 90% of β is allocated to the Phase One, this is not the case for the 60% allocation. We additionally note that the FPR is the highest when Phase One is allocated 90% of β and lowest when 90% is allocated to Phase Two, but the FPR bound of $\alpha = 0.1$ is upheld regardless of the split. As predicted from the inverse relationship between privacy loss and the FNR bound of β, the smaller the β value, the higher privacy loss ϵ is as seen in Table 3. When using smaller values of β, we note that the FNR as well as FPR bounds are still upheld, but the optimal distribution of the β budget across phases cannot be discerned despite varying β values and remains a non-trivial optimization problem.

AVERAGE aggregate function conjunction-only privacy and accuracy results. We model queries for the Sales dataset which use the AVERAGE aggregate function based on meaningful KPIs typically used in decision support applications (e.g. average transaction value). We include the results for conjunction-only complex queries in Figure 10 with a varying number of sub-queries from 1 to 6. We use the same default parameters to evaluate the performance of the Naive, ProBE, ProBE-Naive and ProBE-Ent algorithms. We set $u = 12\%$ for single-step algorithms, $u_0 = 30\%$ for multi-step algorithms and $\beta = 0.05$, $\alpha = 0.1$, as well as $m = 4$ steps and $m_f = 3$ fine-grained steps for ProBE-Ent. Our privacy results in terms of ex-post privacy loss ϵ and min-entropy $\mathcal{H}_{min}$ show that AVERAGE queries have similar results as COUNT queries but with a higher range of values for ϵ, with ProBE-Ent having the lowest privacy loss and highest min-entropy. The higher average for privacy loss is due to the sensitivity of the query being higher (AVG often has a higher sensitivity than COUNT). The accuracy results in terms of false negative rate (FNR) and false positive rate (FPR) show that all algorithms again achieve the bounded guarantees of $\beta = 0.05$, and the multi-step algorithms achieve the FPR bound of α.

Overall β	Phase One β %	Phase Two β %	Privacy loss ϵ	FNR	FPR
$5 \cdot 10^{-2}$	10	90	$4.55 \cdot 10^{-2}$	0	$1.81 \cdot 10^{-3}$
	40	60	$4.56 \cdot 10^{-2}$	0	$3.03 \cdot 10^{-3}$
	50	50	$4.44 \cdot 10^{-2}$	0	$2.40 \cdot 10^{-3}$
	60	40	$4.80 \cdot 10^{-2}$	0	$3.01 \cdot 10^{-3}$
	90	10	$4.16 \cdot 10^{-2}$	0	$5.45 \cdot 10^{-3}$
$5 \cdot 10^{-4}$	10	90	$5.46 \cdot 10^{-2}$	0	$1.21 \cdot 10^{-3}$
	40	60	$5.66 \cdot 10^{-2}$	0	$1.81 \cdot 10^{-3}$
	50	50	$5.12 \cdot 10^{-2}$	0	$4.84 \cdot 10^{-3}$
	60	40	$5.32 \cdot 10^{-2}$	0	$3.63 \cdot 10^{-3}$
	90	10	$5.15 \cdot 10^{-2}$	0	$1.21 \cdot 10^{-3}$
$5 \cdot 10^{-6}$	10	90	$8.15 \cdot 10^{-2}$	0	$1.2 \cdot 10^{-3}$
	40	60	$8.36 \cdot 10^{-2}$	0	$3.07 \cdot 10^{-4}$
	50	50	$8.12 \cdot 10^{-2}$	0	$1.21 \cdot 10^{-3}$
	60	40	$8.18 \cdot 10^{-2}$	0	$6.06 \cdot 10^{-4}$
	90	10	$8.09 \cdot 10^{-2}$	0	$1.12 \cdot 10^{-3}$

Table 3: Privacy loss and Accuracy when varying the β budget distribution across the two ProBE Phases

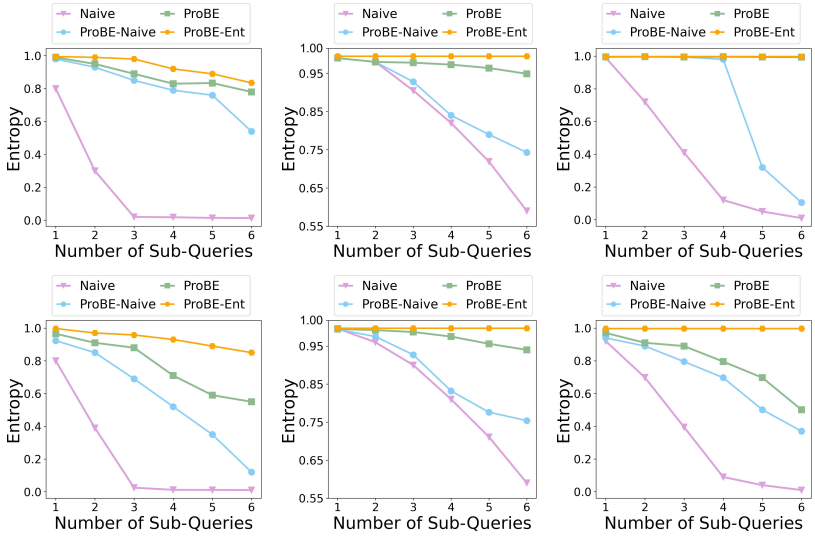
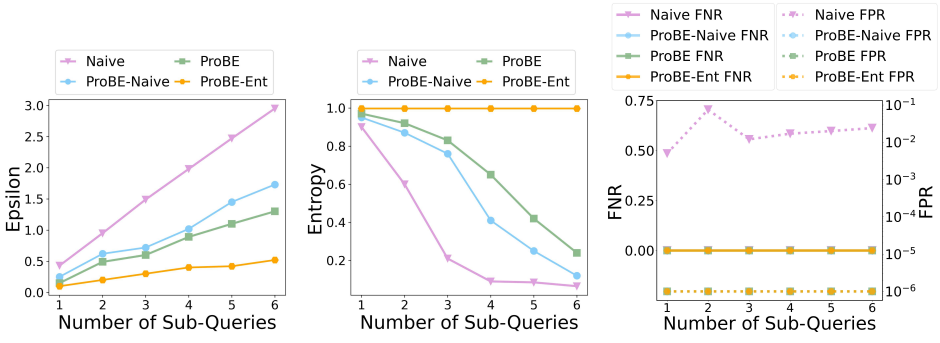


Figure 9: Privacy Loss in terms of Min-Entropy $\mathcal{H}_{min}(\Theta)$ for 1-6 sub-queries at $\beta = 0.05$ and $\alpha = 0.1$ for Conjunction (row 1) and Combined Conjunction/Disjunction (row 2) using NYTaxi (column 1), Sales (column 2) and UCI (column 3) data.



(a) Ex-Post DP ϵ **(b) Min-Entropy $\beta(\Theta)$** **(c) FNR and FPR**

Figure 10: Privacy (ϵ,$\beta(\Theta)$) and Accuracy (FNR, FPR) Results for AVG functions at $\beta = 0.05$ and $\alpha = 0.1$ on the Sales dataset.