Learning to Reconstruct: Statistical Learning Theory and Encrypted Database Attacks

Paul Grubbs, Marie-Sarah Lacharité, Brice Minaud, Kenneth G. Paterson

Learning to Reconstruct: Statistical Learning Theory and Encrypted Database Attacks

Paul Grubbs[1] Marie-Sarah Lacharité[2] Brice Minaud[3,4] Kenneth G. Paterson[2]

[1] Cornell Tech, USA
[2] Royal Holloway, University of London, UK
[3] École Normale Supérieure, CNRS, PSL University, France
[4] Inria, France

pag225@cornell.edu, marie-sarah.lacharite.2015@rhul.ac.uk, brice.minaud@inria.fr, kenny.paterson@rhul.ac.uk

Abstract

We show that the problem of reconstructing encrypted databases from access pattern leakage is closely related to statistical learning theory. This new viewpoint enables us to develop broader attacks that are supported by streamlined performance analyses. As an introduction to this viewpoint, we first present a general reduction from reconstruction with known queries to PAC learning. Then, we directly address the problem of ϵ-approximate database reconstruction (ϵ-ADR) from range query leakage, giving attacks whose query cost scales only with the relative error ϵ, and is independent of the size of the database, or the number N of possible values of data items. This already goes significantly beyond the state of the art for such attacks, as represented by Kellaris *et al.* (ACM CCS 2016) and Lacharité *et al.* (IEEE S&P 2018). We also study the new problem of ϵ-approximate order reconstruction (ϵ-AOR), where the adversary is tasked with reconstructing the order of records, except for records whose values are approximately equal. We show that as few as $\mathcal{O}(\epsilon^{-1} \log \epsilon^{-1})$ uniformly random range queries suffice. Our analysis relies on an application of learning theory to PQ-trees, special data structures tuned to compactly record certain ordering constraints. We then show that when an auxiliary distribution is available, ϵ-AOR can be enhanced to achieve ϵ-ADR; using real data, we show that devastatingly small numbers of queries are needed to attain very accurate database reconstruction. Finally, we generalize from ranges to consider what learning theory tells us about the impact of access pattern leakage for other classes of queries, focusing on prefix and suffix queries. We illustrate this with both concrete attacks for prefix queries and with a general lower bound for all query classes.

Contents

1 Introduction

This article concerns the analysis of *leakage* from *encrypted databases*. The latter are cryptographic techniques that allow a client to outsource a database to an untrusted server while maintaining the ability to make queries on the data. Fuller *et al.* [FVY+17] give a comprehensive survey of this area. All known techniques represent a trade-off between security and efficiency, with various forms of leakage being intrinsic to these approaches. For example, without taking special precautions such as using oblivious memory techniques, the *access pattern*, that is, the set of records returned in response to queries, leaks to the server. While in many cases, formal security proofs are able to establish that nothing more than the access pattern leaks to the adversarial server, there still remains the question: what is the practical impact of this leakage? A more refined version of the question is:

> **If an encrypted database supports a certain class of queries, but leaks the access pattern, then how damaging is that leakage as a function of the number of queries?**

The setting of this article is one in which *only* access pattern is leaked to an adversarial server. Access pattern leakage is inherent to nearly all practical constructions of encrypted databases, and the survey by Fuller *et al.* [FVY+17] overviews a plethora of schemes to which such attacks apply. In the particular case of range queries, all known practical solutions leak this information [LMP18]. Nevertheless, the above question is currently answered using *ad hoc* cryptanalysis, requiring cumbersome and laborious analyses to establish the impact of leakage as a function of the number of queries. The central aim of our work is to transform this situation by bringing statistical learning theory to bear on the problem.

1.1 Database Reconstruction: State of the Art

Range queries are fundamental to the operation of databases, and have rightfully received significant attention in the attack literature. The state of the art for attacks based on leakage from range queries is represented by the work of Kellaris-Kollios-Nissim-O'Neill (KKNO) [KKNO16] and Lacharité-Minaud-Paterson (LMP) [LMP18]. KKNO gave attacks showing that $\mathcal{O}(N^4 \log N)$ queries suffice to achieve *Full Database Reconstruction (FDR)*, that is, to reconstruct the *exact* value for every record. Here, N is the number of different possible values, which we assume without loss of generality come from the interval $[1, N]$. For *dense* data, where every possible value is in at least one record, this was improved to $\mathcal{O}(N^2 \log N)$ queries by KKNO and then to $\mathcal{O}(N \log N)$ queries by LMP. All of these results assume the query distribution is uniform on ranges (though for the results in the dense setting, this assumption is needed only to facilitate analysis and not for the algorithms to succeed).

A typical value of N might be, say, 125 for data pertaining to age in years, making even an $\mathcal{O}(N^4 \log N)$ attack potentially worrisome. But for many data types, N can be much larger – think of discrete data such as numerical zip codes, timestamps, or salary data. For large N, especially when the data is sparse rather than dense (as is typically the case), FDR is much too expensive (KKNO proved a general lower bound of $\Omega(N^4)$ on the number of range queries needed), and really too strong an attack goal.

For this reason, LMP introduced the notion of ϵ-*Approximate Database Reconstruction (ϵ-ADR)*, where the adversary's goal is to find the value of every record up to an (additive) error of ϵN rather than exactly. For small ϵ, such an attack is still extremely effective: imagine learning all salaries in a database up to an error of 1%. LMP gave the first algorithm for ϵ-ADR. Their algorithm achieves ϵ-ADR from access pattern leakage on only $\mathcal{O}(N \cdot \log \epsilon^{-1})$ queries. However, it still requires a density assumption and its analysis is highly complex.

1.2 Overview of Our Contributions

None of the aforementioned attacks exploiting range query leakage is fully satisfactory: FDR is too expensive for large N, while the only ADR algorithm we have (from LMP's work [LMP18]) relies on a density assumption and its query cost still scales with N. This presents a potentially misleading picture of the impact of leakage for range queries, one which may lead to underestimating the potential damage. Additionally, leakage from other kinds of queries has received little attention; nor have other attack settings of practical importance such as known-query attacks.

In this work, we show how statistical learning theory effectively addresses the problem of database reconstruction, yielding new results across a range of settings. A common thread through all of our results is the analysis of *concept spaces* over the set of all queries. The results we apply from learning theory rely on the *VC dimension* of the concept space, intuitively a measure of how complex it is. (See Section 2 for a short primer on statistical learning theory.)

PAC learning and known-query attacks. In Section 3, we show that database reconstruction given a set of *known* queries can be recast as an instance of Probably Approximately Correct (PAC) learning, and standard results from that field can predict how many queries are needed to achieve reconstruction. We present this reduction to PAC learning as an introduction to how we view database reconstruction as a learning problem, as well as an illustration of the power of this viewpoint. While the attack model here is rather powerful, it is considered realistic in some recent literature [ZKP16, GSB+17, GMN+16]; our analysis largely resolves the question of how damaging such attacks can be. In the remainder, we no longer assume queries are known, aligning our setting with most prior work.

Sacrificial ϵ-ADR. In Section 4, we present two new ϵ-ADR algorithms for range queries. These attacks are *scale-free*: their query complexity depends not on the number of possible values N, but only the precision ϵ. They accommodate any number of queries, as opposed to the "all-or-nothing" attacks of KKNO and LMP. To obtain scale-freeness, we must *sacrifice* recovering some records near the endpoints. As we explain in Section 4.1, scale-free ϵ-ADR is impossible in general – for example, $\mathcal{O}(N)$ uniformly random range queries are necessary to recover values near the endpoints 1 and N, so we must sacrifice these.

The first algorithm (Section 4.2), whose analysis is somewhat simpler, achieves sacrificial ϵ-ADR using $\mathcal{O}(\epsilon^{-4} \log \epsilon^{-1})$ uniformly random range queries. Setting $\epsilon = 1/N$ yields an FDR attack with the same complexity as KKNO's original FDR attack. Indeed, our algorithm can be seen as generalizing the ideas of KKNO to the ADR setting, and making it scale-free. In Section 4.3, we introduce our second attack, the ApproxValue algorithm, which achieves sacrificial ϵ-ADR using only $\mathcal{O}(\epsilon^{-2} \log \epsilon^{-1})$ uniformly random queries, but under the additional, mild requirement that the database contains a record whose value is in the range $[0.2N, 0.3N]$ (or its reflection). Setting $\epsilon = 1/N$ in our algorithm again yields a FDR algorithm with complexity $\mathcal{O}(N^2 \log N)$ that works whether data is sparse or dense, assuming only a single favorably-located record. Our proof techniques for both attacks are rooted in learning theory, using VC dimension and the concept of ϵ-samples. Both attacks also come with general lower bounds showing that they are optimal in the number of queries needed within a logarithmic factor.

In order to assess the effectiveness of these algorithms and the tightness of our bounds, we implement our attacks and experiment on synthetic data (Section 4.4). For example, if $N = 10^6$ and the condition of the ApproxValue algorithm is met, KKNO's FDR attack would require about 10^{26} queries. We found experimentally that only 500 queries (or 24 orders of magnitude fewer than KKNO) are needed to approximate almost all records to within 5% error.

Lifting requirements on the query distribution. In view of the previous attacks, it may seem that the topic of analyzing leakage from range queries is mostly closed, but these attacks require that the adversary knows the query distribution, and crucially rely on the assumption that queries are drawn

independently and uniformly at random. This second requirement especially makes little sense for real-world queries, and we contend that practical attacks should not require it. In this regard, we view KKNO's work and our aforementioned results as important indicators of what is possible in principle, and valuable warnings regarding the power of range query leakage, but not as practical, ready-for-use attacks.

The question, then, is what an attacker can hope to learn in practice, given only the access pattern leakage of some range queries. We investigate this question in Section 5, focusing on attacks that do not rely on any assumptions regarding the query distribution: whereas the attack algorithms so far required a distribution that is independently and identically distributed (i.i.d.), uniform, and known to the adversary, in Section 5 we remove all three assumptions. The LMP results already represent a step in this direction, since their algorithms are not distribution-dependent. However, they do require that the database is dense, whereas we wish to investigate this question in the general setting.

Sacrificial ϵ-AOR. In Section 5, we introduce the attack target of *sacrificial ϵ-Approximate Order Reconstruction* (sacrificial ϵ-AOR). This asks that the order of all records should be recovered, except for records that are within ϵN of each other (which the algorithm groups), and the sacrificed records whose values are within ϵN of 1 or N. Thus, save for sacrificed values, sacrificial ϵ-AOR reveals the order of any two records as soon as they are at least ϵN apart.

As our main result in Section 5, we introduce the ApproxOrder algorithm, which takes as input the access pattern leakage of some range queries, builds a *PQ-tree*, and extracts from it approximate order information. The algorithm does not use any knowledge on the query distribution. In particular, it does not require uniformly random queries, nor even i.i.d. queries. If, for the sake of analyzing the algorithm, we assume a uniform query distribution, ApproxOrder achieves sacrificial ϵ-AOR after only $\mathcal{O}(\epsilon^{-1} \log \epsilon^{-1})$ queries. Once again, our analysis relies on learning theory, more specifically the concept of an ϵ-net, and shows that the attack is scale-free. We also prove that the query complexity of our algorithm is optimal within a constant factor.

For $\epsilon = 1/N$, ϵ-AOR yields exact order reconstruction for all records. If the database is dense, then recovering order directly implies recovering values, so we obtain full database reconstruction in $\mathcal{O}(N \log N)$ queries, recovering as a special case the main result of LMP.

The ApproxOrder algorithm is not merely theoretical, but highly practical. There is no barrier (such as an i.i.d. query assumption) to running it on real data. Our experiments in Section 5.4 show that the attack behaves as predicted by the theory. As an example, for $N = 10^6$, after only 500 queries, the attack is able to fully order records, except for records whose difference in value is less than 2% of the support size.

From ϵ-AOR to ϵ-ADR. A crucial question remains: what are the implications of the AOR attack? That is, what does learning approximate order reveal to the attacker? It is well known that leaking record order is highly damaging, if only because it can be closely correlated to record values using an auxiliary distribution [NKW15, GSB+17]. In fact, the severe implications of order leakage is one of the main motivations behind the development of second-generation encrypted databases schemes that attempt to hide that leakage, as argued in [LMP18]. To concretize that point, in Section 5.5 we present an attack showing how approximate database values can be reconstructed from approximate order information: we extend our sacrificial ϵ-AOR attack to a sacrificial ϵ-ADR attack using an auxiliary model of the database distribution. (As per [NKW15, LMP18], such distributions are often available.)

We conduct experiments with real datasets of US ZIP codes and public sector salaries in Section 5.5. The resulting sacrificial ϵ-ADR attack is effective: with only 50 queries, we can learn the first two digits of a ZIP code (often identifying a city) for a majority of records in the target database. With 100 queries on salaries, we can predict a majority of salaries to within 10000 USD. The table in Figure 1 compares our different attacks on range queries.

Beyond range queries. As illustration of the power of the viewpoint we have taken, in Section 6, we generalize approximate reconstruction to other query classes and analyze the resulting attacks using tools from learning theory. Using *generalization error* as a metric γ on the values in the database, we show that all query classes with finite VC dimension reveal the distance (in γ) between the underlying values of records, which allows an attacker to group records whose values are close. Further, we show how to use an ϵ-net to precisely analyze how many queries are needed to guarantee all groups of records have small diameter according to γ. We construct, analyze, and evaluate the first reconstruction attack on prefix queries. We conclude the section with a general lower bound, via a reduction to PAC learning, relating the query class's VC dimension, attack accuracy, and number of queries needed for any reconstruction attack using access pattern leakage. In addition to being of theoretical interest, this suggests VC dimension or similar concepts from learning theory could be a useful way to compare different techniques which leak access patterns.

Notation. Throughout $[n]$ denotes the set of integers $\{1, \ldots, n\}$; $[a, b]$ denotes the set of integers within the given interval; and open brackets such as $[a, b[$ denote that the corresponding endpoint is excluded. (If $b \leq a$, $[a, b[$ is empty.) We model a database as a set of R records where each record has a single attribute that takes an integer value in $[N]$. We let $\mathsf{val}(r) \in [N]$ denote the value of the record r.

Assumptions. We assume the adversary knows the number of possible values N, and the set of all possible queries. We do not assume that the adversary knows the set of all records in advance, or even their number. We do not assume that every value appears in at least one record (no density assumption).

The ApproxValue algorithm in Section 4 further assumes that queries are drawn i.i.d. and uniformly at random. The attack and its analysis can be generalized to other query distributions. As explained earlier, we then introduce the ApproxOrder algorithm in Section 5 precisely to do away with assumptions on the query distribution. Likewise all algorithms in Section 6 function without such assumptions. When it comes to analyzing the query complexity of those algorithms, we are forced to make a hypothesis on the query distribution. In that case, we choose an assumption about the query distribution that helps provide insight into a typical behavior of the algorithm. We stress that that hypothesis is in no way required for the algorithm to function and succeed.

1.3 Related Work

Dautrich Jr. and Ravishankar [JR13] introduced the use of PQ-trees in revealing the order of records in a database with access pattern leakage. They experimentally measured, in some special cases, how quickly the number of orders contained in the tree decreases as more queries are gathered. We use also use PQ-trees for revealing order from range queries, but otherwise our aims are distinct from theirs—their paper focuses primarily on heuristic measures of security after some ordering information is revealed.

Kellaris *et al.* (KKNO) [KKNO16] described the first exact reconstruction attack on range queries with access pattern leakage; Lacharité *et al.* (LMP) [LMP18] improved the results of KKNO in the dense setting, obtaining an $\mathcal{O}(N \log N)$ exact reconstruction attack; see Section 1.1 for more detail. Kornaropoulos et al. [KPT18] gave an approximate reconstruction attack for access pattern leakage from k-nearest-neighbor queries. Other papers attacking encrypted databases include [GSB+17, NKW15, BGC+18, GMN+16, CGPR15]; these mostly analyze so-called "property-revealing encryption" schemes, which leak strictly more than what we assume.

Attack	Goal	Data density	Query complexity		Source
			Proposed attack	**Generic lower bound**	
KKNO	FDR	any	$\mathcal{O}(N^4 \log N)$	$\Omega(N^4)$	[KKNO16]
KKNO	FDR	dense	$\mathcal{O}(N^2 \log N)$	–	[KKNO16]
LMP	FDR	dense	$N \log N + \mathcal{O}(N)$	$\frac{1}{2} N \log N - \mathcal{O}(N)$	[LMP18]
LMP	ϵ-ADR	dense	$\frac{5}{4} N \log \epsilon^{-1} + \mathcal{O}(N)$	$N \log \epsilon^{-1} - \mathcal{O}(N)$	[LMP18]
GeneralizedKKNO	sacrificial ϵ-ADR	any	$\mathcal{O}(\epsilon^{-4} \log \epsilon^{-1})$	$\Omega(\epsilon^{-4})$	Section 4.2
ApproxValue	sacrificial ϵ-ADR	any*	$\mathcal{O}(\epsilon^{-2} \log \epsilon^{-1})$	$\Omega(\epsilon^{-2})$	Section 4.3
ApproxOrder	sacrificial ϵ-AOR	any*	$\mathcal{O}(\epsilon^{-1} \log \epsilon^{-1})$	$\Omega(\epsilon^{-1} \log \epsilon^{-1})$	Section 5

Figure 1: Comparison of database reconstruction attacks that use access pattern leakage from range queries chosen uniformly at random. N is the number of possible plaintext values. All attacks are up to global reflection. Generic lower bounds are for any attack targeting the same goal under the same assumptions. "*" denotes some additional but mild requirements on the existence of records with particular values.

2 Statistical Learning Theory Primer

We begin with a brief introduction to some elements of statistical learning theory that will play a central role in our work. We use terminology from a recent textbook [MU17].

2.1 Concept Spaces, ϵ-Nets, ϵ-Samples

Let X be some set of (base) elements. In this work, X is always finite (although our scale-free reconstruction bounds extend to infinite sets, e.g. continuous ranges, as is). A *concept* (also called *event* or *range*) C is a subset of X. Given a probability distribution $\mathcal{D}$ on the set X, let $f_{\mathcal{D}}$ represent the probability mass function. The probability $\Pr_{\mathcal{D}}(C)$ of a concept C is equal to the probability that a single element of X sampled according to $\mathcal{D}$ is in C, i.e., $\Pr_{\mathcal{D}}(C) = \sum_{c \in C} f_{\mathcal{D}}(c)$.

A *concept space* (or *set system*) is a pair $(\mathsf{X}, \mathbb{C})$ where $\mathbb{C}$ is a set of concepts (subsets) of X. Any concept C can also be viewed as a function $\mathsf{X} \to \{0, 1\}$ (its characteristic function): the function's output on input $x \in \mathsf{X}$ is 1 if $x \in C$ and 0 otherwise. Given a concept space $(\mathsf{X}, \mathbb{C})$ and a sample S of elements drawn from X according to $\mathcal{D}$, we may ask the following questions:

- Does every concept in $\mathbb{C}$ with some not-too-small probability occur in the sample S?
- Is the relative occurrence of every concept of $\mathbb{C}$ in the sample S close to its expectation?

Answering these questions involves analyzing objects called ϵ-nets [HW86] and ϵ-samples.

Definition 2.1. *A subset $S \subseteq \mathsf{X}$ is an ϵ-net for the concept space $(\mathsf{X}, \mathbb{C})$ with respect to the distribution $\mathcal{D}$ if for every event $C \in \mathbb{C}$ with $\Pr_{\mathcal{D}}(C) \geq \epsilon$, the intersection $S \cap C$ is non-empty.*

Definition 2.2. *A subset $S \subseteq \mathsf{X}$ is an ϵ-sample (also called ϵ-approximation) for the concept space $(\mathsf{X}, \mathbb{C})$ with respect to the distribution $\mathcal{D}$ if for every concept $C \in \mathbb{C}$,*

$$\left| \frac{|S \cap C|}{|S|} - Pr_{\mathcal{D}}(C) \right| \leq \epsilon.$$

Informally, a sample S is an ϵ-sample when every concept's relative frequency in S is within ϵ of its true probability. It is an ϵ-net iff every concept of probability at least ϵ occurs in the sample.

One way to analyze when a set S is an ϵ-net or an ϵ-sample is to characterize the complexity of the concept space. We turn to this next.

2.2 Shattering, VC Dimension, Growth Functions

The critical measures in determining the complexity of a concept space are the growth function $m_{\mathbb{C}}(n)$ and the Vapnik-Chervonenkis (VC) dimension d, which are related. Given a concept space $(\mathsf{X}, \mathbb{C})$ and a finite sample $S \subseteq \mathsf{X}$, an important object is the *set of subsamples of S induced by $\mathbb{C}$* (also called *the projection of $\mathbb{C}$ on S*): $\mathbb{C}_S := \{C \cap S\}_{C \in \mathbb{C}}$. The size of this set is the *index of $\mathbb{C}$ with respect to S*:

$$\Delta_{\mathbb{C}}(S) := |\mathbb{C}_S| = |\{C \cap S : C \in \mathbb{C}\}| .$$

Clearly, the index of a concept space relative to a set S is at most $2^{|S|}$, and, when $\mathbb{C}$ is finite, it is at most $|\mathbb{C}|$.

The concept space $(\mathsf{X}, \mathbb{C})$ *shatters* the sample $S \subseteq \mathsf{X}$ if $\mathbb{C}$ induces all possible subsamples of S, i.e., $\Delta_{\mathbb{C}}(S) = 2^{|S|}$.

The *VC dimension* (also called *density* [Sau72] or *capacity* and denoted by d) of a concept space $(\mathsf{X}, \mathbb{C})$ is the largest cardinality (possibly infinite) of a set $S \subseteq \mathsf{X}$ that can be shattered by $\mathbb{C}$. (It is sufficient for only one set of this size to exist; not all sets of this size need to be shattered by $\mathbb{C}$.) VC dimension is an indicator of the complexity of a concept space. Related to VC dimension is the *growth function* of a concept space $(\mathsf{X}, \mathbb{C})$, which is the maximum index of $\mathbb{C}$ over all samples $S \subseteq \mathsf{X}$ of size n: $m_{\mathbb{C}}(n) := \max_{S \subseteq \mathsf{X}: |S|=n} \Delta_{\mathbb{C}}(S)$.

The VC dimension, then, is the largest value of n for which the growth function equals 2^n. Knowing the VC dimension of a concept space is sufficient to determine an upper bound on the growth function: either d is infinite or the growth function is bounded by $\sum_{i=0}^{d} \binom{n}{i}$.

Lemma 2.3 (Sauer's Lemma [Sau72])**.** *Let $(\mathsf{X}, \mathbb{C})$ be a concept space having finite VC dimension d. Then, the growth function satisfies $m_{\mathbb{C}}(n) \leq \sum_{i=0}^{d} \binom{n}{i}$.*

By induction on d, it is straightforward to show that the partial sum of binomial coefficients $\sum_{i=0}^{d} \binom{n}{i}$ is upper-bounded by n^d for $d \geq 2$.

2.3 Sufficient Conditions for ϵ-Nets and ϵ-Samples

In their groundbreaking paper, Vapnik and Chervonenkis established a lower bound [VC71, Thm. 2] on the probability that a sample S is an ϵ-sample, i.e., that the relative frequencies of events in $\mathbb{C}$ are all within ϵ of their true probabilities.

Theorem 2.4 (Sufficient conditions for ϵ-sample [VC71])**.** *Let $(\mathsf{X}, \mathbb{C})$ be a concept space with growth function $m_{\mathbb{C}}(n)$ and VC dimension d. Let $\mathcal{D}$ be a probability distribution on X and let S be a set of size n drawn from X according to $\mathcal{D}$. Then, for any $\epsilon > 0$, the probability that S is an ϵ-sample is at least $1 - 4 \cdot m_{\mathbb{C}}(2n) \cdot e^{-\epsilon^2 n/8}$. In particular, there is an n that is*

$$\mathcal{O}\left(\frac{d}{\epsilon^2} \log \frac{d}{\epsilon} + \frac{1}{\epsilon^2} \log \frac{1}{\delta}\right)$$

such that a sample S of size at least n is an ϵ-sample with probability at least $1 - \delta$.

More precisely, [MU17, Thm. 14.15] establishes that a sample of size at least $\frac{32d}{\epsilon^2}\log\frac{64d}{\epsilon^2}+\frac{16}{\epsilon^2}\log\frac{2}{\delta}$ is an ϵ-sample with probability at least $1-\delta$.

Inspired by Vapnik and Chervonenkis's work on ϵ-samples, Haussler and Welzel introduced ϵ-nets and derived a lower bound in the case where the distribution over X is uniform [HW86, Thm. 3.7]. Later work extended this bound to arbitrary distributions:

Theorem 2.5 (Sufficient conditions for ϵ-net [MU17])**.** *Let $(\mathsf{X},\mathbb{C})$ be a concept space with growth function $m_{\mathbb{C}}(n)$ and VC dimension d. Let $\mathcal{D}$ be a probability distribution on X and let S be a set of size n drawn from X according to $\mathcal{D}$. Then, for any $\epsilon>0$, the probability that S is an ϵ-net is at least $1-2\cdot m_{\mathbb{C}}(2n)\cdot e^{-\epsilon n/2}$. In particular, there is an n that is*

$$\mathcal{O}\left(\frac{d}{\epsilon}\log\frac{d}{\epsilon}+\frac{1}{\epsilon}\log\frac{1}{\delta}\right)$$

such that a sample of at least this size is an ϵ-net with probability at least $1-\delta$. Specifically, a random sample of size at least $\max\{\frac{8d}{\epsilon}\log\frac{16d}{\epsilon},\frac{4}{\epsilon}\log\frac{2}{\delta}\}$ is an ϵ-net with probability at least $1-\delta$.

Ehrenfeucht *et al.* prove a lower bound [EHKV89, Cor. 5] on the number of samples needed to obtain an ϵ-net with probability at least $1-\delta$. Since every ϵ-sample is an ϵ-net, this lower bound also applies to ϵ-samples.

Theorem 2.6 (Necessary conditions for ϵ-net [EHKV89, MU17])**.** *Let $(\mathsf{X},\mathbb{C})$ be a concept space of VC dimension d. Let $\mathcal{D}$ be a probability distribution on X and let S be sample drawn from X according to $\mathcal{D}$. Let $\epsilon>0$ and $\delta>0$. Suppose S is an ϵ-net with probability at least $1-\delta$. Then $|S|=\Omega\left(\frac{d}{\epsilon}+\frac{1}{\epsilon}\log\frac{1}{\delta}\right)$.*

2.4 PAC Learning

Introduced by Valiant [Val84], PAC learning is concerned with algorithms that learn from labelled examples. (We restrict our attention to *realizable* and *consistent* PAC learning; for a more general treatment see [KVV94].) Using the terminology above, a learner $\mathcal{L}$ is an algorithm which takes as input a transcript $\{(s_i,C(s_i))\}_{i=1}^m$ of elements from X (where each $s_i \leftarrow\!\!\$\ \pi$ for some distribution π on X) along with their labels according to the unknown concept C. A learner outputs a *hypothesis* $H\in\mathbb{C}$ representing its guess for C.

The learner $\mathcal{L}$ is a *PAC learner* for $\mathbb{C}$ if for any $C\in\mathbb{C}$, distribution π on X, and $0<\epsilon,\delta<1/2$, for any sample (or transcript) of size m in $\mathcal{O}(\mathrm{poly}(\frac{1}{\epsilon},\frac{1}{\delta}))$ drawn according to π, the *generalization error* $\Pr_\pi[\,\{x\in\mathsf{X}\,|\,H(x)\neq C(x)\}\,]$ is less than ϵ with probability at least $1-\delta$. In words, the generalization error is the probability under π that an element is labelled differently by H and C. A central result [BEHW86] in learning theory states that if $\mathcal{C}$ has finite VC dimension, then there exists a (not necessarily efficient) PAC learner for $\mathcal{C}$. In particular, the following holds [MU17].

Theorem 2.7. *Let $(\mathsf{X},\mathbb{C})$ be a concept space with finite VC dimension d. Then for any $0<\epsilon,\delta<1/2$, there exists a PAC learner $\mathcal{L}$ for $(\mathsf{X},\mathbb{C})$ that uses a sample of size*

$$m=\mathcal{O}\left(\frac{d}{\epsilon}\log\frac{d}{\epsilon}+\frac{1}{\epsilon}\log\frac{1}{\delta}\right).$$

That is, for any concept $C\in\mathbb{C}$, with probability at least $1-\delta$, $\mathcal{L}$ achieves generalization error less than ϵ using a sample of size m.

3 PAC Learning and Database Reconstruction Attacks

In this section, we begin exploring the connection between learning theory and database reconstruction attacks. Concretely, we demonstrate a connection between approximate database reconstruction and "Probably Approximately Correct" (PAC) learning [Val84] in the setting where the attacker has access pattern leakage from some known queries. For a brief introduction to PAC learning, see Section 2.4.

Reconstruction via PAC learning. The attack setting we consider here is one in which an attacker has observed the access pattern leakage from a number of *known* queries drawn i.i.d. from a fixed query distribution (which the adversary does not need to know). The assumption of known queries is somewhat stronger than has been considered previously in this literature; however, some recent works have argued that it is realistic [GSB+17, GMN+16] on the grounds that the adversary is able to make some queries or has compromised an honest user.

A crucial question is the relationship between the *number* of known queries and the amount of information the adversary can learn about the database itself, cf. Section 1. We will see that this question is largely resolved via a simple reduction to PAC learning in the known query setting.

We can think of a database DB with R records having values in $[N]$ as being a vector of length R with values in $[N]$; the value of record j is $DB[j]$. We construct a concept space $\mathcal{C} = (\mathcal{Q}, \mathbb{C})$ as follows. The points in the ground set are the possible queries $q \in \mathcal{Q}$. We write $q(i) = 1$ when value $i \in [N]$ matches query q. We set $C_i = \{q \in \mathcal{Q} \,|\, q(i) = 1\}$. We then define $\mathbb{C} = \{C_i : i \in [N]\}$. With this set-up, we have the following result.

Theorem 3.1. *Let $\mathcal{Q}$ be a class of queries and $\mathcal{C} = (\mathcal{Q}, \mathbb{C})$ be the concept space constructed as above. Let π_q be any distribution over $\mathcal{Q}$. Let d be the VC dimension of $\mathcal{C}$, and assume d is finite. Then, there is an adversary such that for any database DB, given as input $m \in \mathcal{O}(\frac{d}{\epsilon} \log \frac{d}{\epsilon\delta})$ queries sampled from π_q and their access pattern leakage on DB, the adversary outputs a database DB' such that $\Pr_{\pi_q}[\, q(DB[j]) \neq q(DB'[j]) \,] \leq \epsilon$ holds simultaneously for all $j \in [R]$, with probability at least $1 - R\delta$.*

This theorem requires some explanation. In the statement, we chose to use the generalization error $\Pr_{\pi_q}[\, q(DB[j]) \neq q(DB'[j]) \,]$ as the accuracy measure. This is intended to surface the core points without adding unnecessary detail, but it may also make the result hard to interpret. Section 6 studies in more detail how generalization error relates to traditional notions of attack accuracy.

The proof proceeds via a natural reduction to PAC learning. The adversary gets as input m known queries along with their access pattern leakage (i.e. which records match the query) for each of the R records in the database. The core observation is that the access pattern is a binary classification of each database element; further, each database value is a concept in $\mathbb{C}$. This means that the task of reconstructing each database element can be seen as R PAC learning experiments for the concept space $\mathcal{C}$ defined above. The adversary simply runs the PAC learner from Theorem 2.7 R times, invoking it once for each record j. For each invocation, the adversary gives the learner as input the m queries and their access patterns (i.e. the 0/1 labellings) for record j. Each time the learner is run, it outputs a hypothesis $H_{\mathsf{ind}} \in \mathbb{C}$ corresponding to an element of $[N]$. The adversary's complete output is then of the form $[H_1, H_2, \ldots, H_R]$, which we denote by DB'. Each independent invocation of the learner outputs a hypothesis H_j such that $\Pr_{\pi_q}[\, q(H_j) \neq q(DB[j]) \,] > \epsilon$ with probability at most δ, and a union bound over the R elements completes the proof.

Remark. Here, we obtain a probability bound that depends on R. While this may look discouraging, the sample complexity of PAC learning is only logarithmic in δ, so the resulting loss in tightness is small. Further, in Appendix A, we show that the dependency on R can be removed. The proof in Appendix A does not use a generic reduction to PAC learning; instead it uses a fixed learner and applies the ϵ-net

theorem. The approach we chose in this section is simpler and directly highlights the connection with PAC learning.

Extensions. The above result can be extended in several ways. First, a symmetric and nearly identical result can be proven about *query* reconstruction attacks in the presence of known records. Such a result would proceed as above except flipping the role of the queries and values in the concept space. An interesting extension of the above result (and its twin for known database elements) is a setting where the adversary has both known and unknown queries (or database elements). One question which our PAC learning approach can address is how much information the adversary learns about the *unknown* information elements by PAC learning with its known information.

Some authors have argued recently that the even stronger setting of *chosen* query attacks is a realistic threat model for encrypted databases. With chosen queries, the corresponding learning setting is not PAC learning, but active learning [DL]. Active learning is similar to PAC learning except the learner can adaptively query an oracle which labels points in X according to the unknown concept. Interestingly, both the folklore binary-search attack on order-revealing encryption and the Zhang *et al.* [ZKP16] document-injection attack can be viewed as active learning algorithms.

We note that lower bounds on the sample complexity of PAC learning can be used to prove that certain kinds of security guarantees hold even in the presence of some known or chosen queries (or database elements). In Section 6 we state and prove one such result which does not assume any records or queries are known to the adversary.

Closing remark. In the encrypted database literature, it has become apparent that known- and chosen-query attacks are damaging. However, the *quantitative* question ("How severe a risk is a known- or chosen-plaintext attack?") has not been fully explored. We posit that extending the above result using techniques from learning theory will fully resolve this question. Rather than developing this theme further here, we leave it to future work and focus the remainder of this work on more challenging attack settings.

4 Sacrificial Approximate Database Reconstruction

In this section, we turn to range queries, and introduce *sacrificial ϵ-approximate database reconstruction* (sacrificial ϵ-ADR). Sacrificial ϵ-ADR asks to successfully recover the value of every record in the database within ϵN, for some target precision ϵ, save for records whose value lies within ϵN of 1 or N. The term *ϵ-approximate* means that reconstruction is within an error of ϵN, as in [LMP18]. The term *sacrificial* means the attack "sacrifices" records whose value lies within ϵN of the endpoints. We explain the need for this and provide a full definition in Section 4.1. The rest of the section presents two results.

In Section 4.2, we extend KKNO's database reconstruction attack [KKNO16] to sacrificial ϵ-ADR. A direct application of the ϵ-sample theorem from learning theory shows the dependency on N vanishes: the required number of queries becomes $\mathcal{O}(\epsilon^{-4} \log \epsilon^{-1})$, making the attack *scale-free*.

In Section 4.3, we introduce a new algorithm, ApproxValue, for sacrificial ϵ-ADR with a mild additional hypothesis h_1: the database contains at least one record whose value lies in $[0.2N, 0.3N] \cup [0.7N, 0.8N]$. Under this hypothesis, ApproxValue achieves sacrificial ϵ-ADR within only $\mathcal{O}(\epsilon^{-2} \log \epsilon^{-1})$ queries. The analysis also uses the ϵ-sample theorem, but is somewhat more involved. This attack shows the pathological nature of KKNO's lower bounds on query complexity for FDR. An experimental validation in Section 4.4 supports the analysis, and shows that the constants in the $\mathcal{O}$ notation are empirically very small.

As noted in the introduction, the previous two results also imply full database reconstruction within

$\mathcal{O}(N^4 \log N)$ queries in general, and $\mathcal{O}(N^2 \log N)$ when h_1 is satisfied. In Appendix E, we show that both attacks are optimal in data within a logarithmic factor—any adversary achieving sacrificial ϵ-ADR for all databases (resp. databases satisfying h_1) must require $\Omega(\epsilon^{-4})$ (resp. $\Omega(\epsilon^{-2})$) queries.

4.1 Definition of Sacrificial ϵ-ADR

We now formally define sacrificial ϵ-approximate database reconstruction (sacrificial ϵ-ADR). Let $\epsilon > 0$ be the desired precision. Let $\mathsf{est\text{-}val}(r)$ denote the value predicted by the algorithm for record r. Sacrificial ϵ-ADR is said to succeed iff one of the following two events occur:

1. For every record r such that $\epsilon N \leq \mathsf{val}(r) \leq N + 1 - \epsilon N$, $|\mathsf{est\text{-}val}(r) - \mathsf{val}(r)| < \epsilon N$.
2. For every record r such that $\epsilon N \leq \mathsf{val}(r) \leq N + 1 - \epsilon N$, $|\mathsf{est\text{-}val}(r) - (N + 1 - \mathsf{val}(r))| < \epsilon N$.

The fact that reconstruction is only possible up to reflection is inherent to this setting, as seen in [KKNO16, LMP18]. It is required that for all values (except those within ϵN of the extrema), either the estimated value is within ϵN of the correct value, or its reflection. But whichever case it is holds *simultaneously* for all values. In other words, only one bit of information is missing *globally* regarding the reflection symmetry. Note that setting $\epsilon = 1/N$ yields full database reconstruction (FDR), i.e. exact value reconstruction for all records.

Finally, we come to explaining why our attack needs to be *sacrificial*. Sacrificing values that are close to 1 and N is inherent to a *scale-free* attack under a uniform query assumption. Intuitively, these values are harder to recover because fewer range queries touch them. The probability of hitting records with values 1 and N with a uniform range query is $2/(N+1) = \mathcal{O}(1/N)$. This remains true for any record whose value is within $\mathcal{O}(1)$ of 1 or N: hitting one of these records requires $\Omega(N)$ queries. If they are not hit, then it is impossible for the algorithm to differentiate them or determine which records are on the same side of $N/2$—reflection symmetry cannot be determined *globally* for those values. Note that if the set of all records is known our algorithms can infer that these records have values close to either 1 or N because they were not hit by a query.

If a query on some range $[1, x]$ for some $x \in [\epsilon N, N + 1 - \epsilon N]$ is ever issued, then the attacker is trivially able to break the reflection symmetry between the values within ϵN of 1 and N (since the query will hit records with values near one of the endpoints, but not the other). The problem is that with uniform queries, the probability of such a query is $\mathcal{O}(1/N)$, so requiring such a query to occur is not scale-free. In practice, though, a query of that form seems likely, since endpoints are generally "interesting" to query. For that reason, we view the sacrificial aspect of the attack as more of an artefact of the analysis than a practical issue. Nevertheless, it must be addressed in a formal treatment of the attack.

4.2 Generalizing the KKNO Attack

We now present our generalization of the KKNO attack to sacrificial ϵ-ADR. Our algorithm proceeds in two steps. The first step is to (approximately) recover the value of each record up to reflection individually: for each record, we recover an approximation of its *symmetric value*, defined as $\mathsf{symval}(r) \stackrel{\text{def}}{=} \min\big(\mathsf{val}(r), N + 1 - \mathsf{val}(r)\big)$. The second step of the algorithm is to determine which values are on the same side of $N/2$ so that, in the end, the value of records is recovered up to reflection *globally*, as discussed above.

We focus here on the first step of the attack because it suffices to highlight the main ideas. For the second step and its analysis, we refer the reader to Appendix B. Note that the first step does not sacrifice any values: this is necessary only to break the reflection symmetry.

The idea underpinning the attack is natural: a given query distribution (in this case, the uniform distribution) induces a distribution on the probability that each value is hit by a range query. By measuring that probability empirically, the value of a record can be inferred. More precisely, for a value $k \in [1, N]$, let A_k denote the set of ranges in $[1, N]$ that contain k. Observe that there are $|[1, k] \times [k, N]| = k(N + 1 - k)$ such ranges.

We also assimilate A_k with the event that a uniform range falls within A_k, i.e. contains the value k. Since there are $N(N + 1)/2$ possible (non-empty) ranges in total, we have:

$$p(k) \stackrel{\text{def}}{=} \Pr(A_k) = \frac{2}{N(N+1)} k(N + 1 - k). \tag{1}$$

We note that $x \mapsto p(x)$ is quadratic and reaches its maximum at $x = (N + 1)/2$. It is symmetric about that value, as implied by the reflection symmetry of the setting.

The algorithm simply measures $p(x)$ empirically for each record by counting how many times that record is hit by a query, and dividing by the number of queries. It then infers the symmetric value of the record by choosing k such that $p(k)$ is as close as possible to the empirical measurement. Pseudo-code is provided in Algorithm 1.

Algorithm 1 Estimating symval.

Input: $\{\mathcal{M}_i\}_{1 \le i \le Q}$.
Output: $\mathsf{est\text{-}symval} : \mathcal{R} \to \{1, \ldots, N/2\}$.
1: **for all** $r \in \mathcal{R}$ **do**
2: $\quad c \leftarrow |\{i : r \in \mathcal{M}_i\}| / Q$
3: $\quad \mathsf{est\text{-}symval}(r) \leftarrow \arg\min_{k \in \{1,\ldots,N/2\}} |p(k) - c|$
4: **end for**
5: **return** $\mathsf{est\text{-}symval}$

We now turn to the analysis of the algorithm: how many queries are required to achieve sacrificial ϵ-ADR? Because the function p used to infer record values is quadratic and flat around $(N + 1)/2$, getting an error of ϵ on the *input* of p, i.e., on record values, requires an error bounded by $\mathcal{O}(\epsilon^2)$ on the *output* of p. That is, for ϵ-ADR to succeed, the difference between the empirical estimate c/Q for a record r and its expectation $p(\mathsf{val}(r))$ should be $\mathcal{O}(\epsilon^2)$. See Appendix B for the formal proof.

Hence what we want is that the empirical probability of each event A_k should be within $\mathcal{O}(\epsilon^2)$ of its expected value, for all values k. If we were to naively apply a union bound, since there are N distinct values k, we would get a dependency on N. Instead, a direct application of VC theory shows that $\mathcal{O}(\epsilon^{-4} \log \epsilon^{-1})$ queries suffice, with no dependency on N. To see this, the idea is to define the ground set X as the set of all ranges in $[1, N]$, and the concept set $\mathbb{C}$ as the A_k's, i.e., each A_k is a concept. Then what we want is exactly a $\Omega(\epsilon^2)$-sample on that concept class.

Proposition 4.1. *The growth function of* $(\mathsf{X}, \mathbb{C})$ *is* $2n$*, and its VC dimension is 2.*

The proof of Proposition 4.1 is given in Appendix B. As a direct consequence, we can apply the ϵ-sample theorem (Theorem 2.4), with $\Omega(\epsilon^2)$ playing the role of the ϵ in the statement of the theorem, to conclude that

$$\mathcal{O}\left(\epsilon^{-4} \log \epsilon^{-1} + \epsilon^{-4} \log \delta^{-1}\right)$$

queries suffice for Algorithm 1 to recover the symmetric value of all records within ϵN, except with probability at most δ. Thus for any fixed probability of success $\eta = 1 - \delta < 1$, Algorithm 1 succeeds within $\mathcal{O}(\epsilon^{-4} \log \epsilon^{-1})$ queries.

For the full attack, see Appendix B. The rest of the attack uses similar ideas, and the first step is representative of the techniques involved. The final query complexity remains $\mathcal{O}(\epsilon^{-4} \log \epsilon^{-1})$.

4.3 The ApproxValue Attack

Next we introduce the ApproxValue algorithm, which achieves sacrificial ϵ-ADR within $\mathcal{O}(\epsilon^{-2}\log\epsilon^{-1})$ queries, saving roughly a square root factor over the generalization of KKNO's algorithm presented above. In particular, setting $\epsilon = 1/N$ yields full database reconstruction within $\mathcal{O}(N^2 \log N)$ queries, significantly improving on KKNO's original $\mathcal{O}(N^4 \log N)$ bound. This comes at the cost of the attack requiring a mild hypothesis about the database. The hypothesis h_1 asks that there exist at least one record in the database with a value in $[0.2N, 0.3N] \cup [0.7N, 0.8N]$. The constants here are for concreteness; others would work as well. The record does not need to be known to the adversary, it suffices that it exist in the database.

We note that an hypothesis such as h_1 is necessary to achieve a query complexity of $\mathcal{O}(\epsilon^{-2}\log\epsilon^{-1})$: in Appendix E, we prove that any algorithm achieving sacrificial ϵ-ADR in full generality requires $\Omega(\epsilon^{-4})$ queries. With h_1 we prove a similar lower bound of $\Omega(\epsilon^{-2})$, so ApproxValue is optimal in data within a logarithmic factor. We believe that $\mathcal{O}(\epsilon^{-2})$ (resp. N^2) better captures the actual cost of ϵ-ADR (resp. full reconstruction) on a real database, assuming a uniform query distribution: the $\mathcal{O}(\epsilon^{-4})$ cost of the original attack was because the bound had to account for pathological cases where h_1 is not satisfied. Such databases have all records concentrated around 1, $N/2$ and N. None of the real datasets used in our experiments had this property.

Next we explain how we gain a square factor over the previous attack, which required $\mathcal{O}(\epsilon^{-4}\log\epsilon^{-1})$ queries. Above we saw that the main ϵ^4 term comes as a result of ϵ being squared *twice*: first, to move from the estimate c/Q of $p(\mathsf{val}(r))$ to the symmetric value of the record, and second by applying the ϵ-sample theorem to uniformly approximate p across all values.

The second squaring is inherent: even if we wanted to approximate p on a single value, we would still need to approximate p within ϵ, which requires $\Omega(\epsilon^2)$ queries. (See Lemma E.1 in Appendix E.) In contrast, the first squaring ultimately comes from the fact that p is quadratic. We can avoid it if we use a linear function to approximate record values: this is exactly what happens in the ApproxValue algorithm.

Pseudo-code of the ApproxValue algorithm is provided in Algorithm 2. The first step is to identify a record whose (symmetric) value is as close as possible to $N/4$. Hypothesis h_1 implies a suitable record exists. We call the resulting record the *anchor* record. Because its value is close enough to $N/4$, identifying such a record does not incur the quadratic cost of the function p as in the previous section, because that cost only occurs near the extremum of the function at $(N+1)/2$. We then determine the value of every other record up to symmetry around the anchor by measuring the empirical probability that a query hits both the target record and the anchor record. Let v_A denote the value of the anchor record and k denote the value of a record we are trying to approximate. The expectation of the previous probability is

$$d(v_A, k) \stackrel{\text{def}}{=} \frac{2}{N(N+1)} \cdot \begin{cases} k(N+1-v_A) & \text{if } k \le v_A \\ v_A(N+1-k) & \text{if } k > v_A. \end{cases}$$

The second step of the ApproxValue algorithm is to use the function $x \mapsto d(v_A, x)$ exactly as we used $x \mapsto p(x)$ in the previous section to estimate a record's value. The crucial point is that while $x \mapsto p(x)$ was quadratic, $x \mapsto d(v_A, x)$ is piecewise linear (with a single bend at v_A). This avoids the first squaring discussed earlier.

The third and final step of ApproxValue is to use p once again to break the symmetry around v_A inherent to the mapping $x \mapsto d(v_A, x)$ used in the previous step. However, this limited use of p does not incur a new square factor. In the end, we obtain the following result.

Algorithm 2 Approximate database reconstruction algorithm.

ApproxValue($\mathcal{Q}$):
Input: Set of queries $\mathcal{Q}$.
Output: Function est-val approximating val.

$Q \leftarrow |\mathcal{Q}|$
for each record r **do**
  $c(r) \leftarrow |\{q \in \mathcal{Q} : r \in q\}|$
  $\widetilde{v}(r) \leftarrow \arg\min_k |c(r)/Q - p(k)|$
end for
$r_A \leftarrow \arg\min_r |\widetilde{v}(r) - N/4|$ ▷ Anchor record
$\widetilde{v_A} \leftarrow \widetilde{v}(r_A)$ ▷ Est. anchor value
for each record r **do**
  $c'(r) \leftarrow |\{q \in \mathcal{Q} : r_A, r \in q\}|$
  $\widetilde{w_L} \leftarrow \arg\min_{k \in [1, \widetilde{v_A}]} |d(\widetilde{v_A}, k) - c'(r)/Q|$
  $\widetilde{w_R} \leftarrow \arg\min_{k \in [\widetilde{v_A}, N]} |d(\widetilde{v_A}, k) - c'(r)/Q|$
  if $c(r)/Q < (p(\widetilde{w_L}) + p(\widetilde{w_R}))/2$ **then**
    est-val$(r) \leftarrow \widetilde{w_L}$
  else
    est-val$(r) \leftarrow \widetilde{w_R}$
  end if
end for

Theorem 4.2. *Let $\epsilon < 1/4$. Assume the database under attack satisfies hypothesis h_1. Then after*

$$\mathcal{O}\Big(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\Big).$$

queries, ApproxValue *achieves sacrificial ϵ-ADR with probability of success at least $1 - \delta$.*

A formal proof is given in Appendix C. Given any constant probability of success $\eta < 1$, ApproxValue achieves sacrificial ϵ-ADR within $\mathcal{O}(\epsilon^{-2}\log\epsilon^{-1})$ queries.

4.4 Experimental Results

The ApproxValue attack achieves ϵ-ADR within $\mathcal{O}(\epsilon^{-2}\log\epsilon^{-1})$ queries (for any given constant probability of success $\eta < 1$). We experimentally evaluate the tightness of this bound for a fixed number of records, R, and various numbers of possible values, N, so that we generate both dense and sparse databases. Record values are sampled uniformly at random, so hypothesis h_1 was satisfied with high probability. Our results are averaged over 500 databases, each with 500 randomly sampled queries.

For the attack to succeed, the difference $|\mathsf{est\text{-}val}(r) - \mathsf{val}(r)|$ should be at most ϵN for records at least ϵN away from the endpoints. The records whose values are near the endpoints may have been placed on the wrong side of $N/2$ relative to the anchor record. The bottom group of lines in Figure 2 shows, after every 10 queries, the maximum symmetric value of such misclassified records. As discussed in Section 4.1, sacrificing reconstruction of some records is necessary. Nevertheless, we see that our practical results are even better than Theorem 4.2 suggests: the upper bound on the maximum symmetric value of a sacrificed record still holds when we take it *with all constants set to 1* – in particular, taking the VC dimension to be 1, not taking into account the success probability, and taking any multiplicative constant hidden by the $\mathcal{O}()$ notation to be 1.

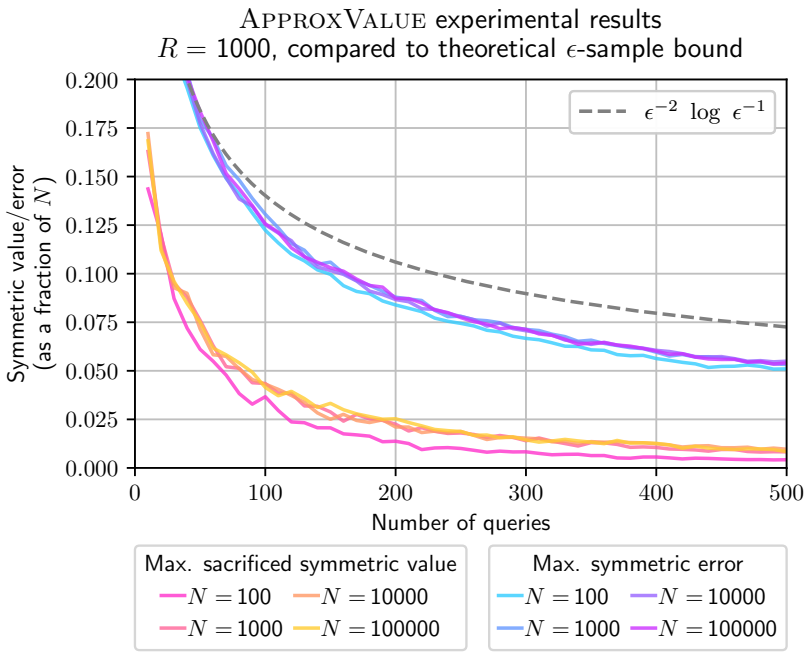

Figure 2: Maximum symmetric errors of all records and maximum symmetric values of records that were sacrificed. Results averaged over 500 databases satisfying h_1 for each value of N.

The primary reason this "no-constants" bound holds is because the bound in Theorem 4.2 inherits the looseness of the ϵ-sample theorem (cf. Theorem 2.4): while VC theory is a good predictor of asymptotic behavior, constants are notoriously loose. In particular, one point where loss of tightness arises in the proof of the ϵ-sample theorem (e.g., as in [MU17, Lemma 14.17]) is when using the growth function to upper bound the number of subsamples induced in the so-called double sample. Tightening this bound is possible, for instance, with sample-based growth functions [STW99]. A benefit of running experiments is that they allow us to estimate the constant in practice: in our experiments, simply setting all constants to 1 provided a reasonable estimation of the attack's success.

In addition to limiting the sacrificed values' distance from the endpoints, a successful ϵ-ADR attack must correctly estimate the other records' values within ϵN, up to global reflection. The upper group of lines in Figure 2 is the maximum error of the *symmetric* values, i.e., the maximum difference $|\min\{\mathsf{est\text{-}val}(r), N + 1 - \mathsf{est\text{-}val}(r)\} - \mathsf{symval}(r)|$ over all records r, as a fraction of N. The reason we plot the symmetric error rather than the absolute error is that it allows us to present results for all records at once—even sacrificed records. It also gives an upper bound on the errors $|\mathsf{est\text{-}val}(r) - \mathsf{val}(r)|$ for records that were not sacrificed. Overall, we see that experimental results behave in the manner predicted by the theory, including scale-freeness, and that the $\mathcal{O}()$ upper bound derived by the theory holds in practice, even when setting the hidden multiplicative constant to just 1.

5 Approximate Order Reconstruction

Attack algorithms in the previous section strongly relied on the query distribution being known to the adversary, and on queries being drawn independently and uniformly at random. Of course, real-world queries are hardly independent, let alone i.i.d. uniform. In this section, we propose and analyze an algorithm that forgoes any such assumption.

As a starting point, observe that, absent any kind of frequency information, access pattern leakage on range query still reveals something about the *order* of records. To see this, consider the following simple example. Say we have three records a, b, c, and observe the leakage of two queries: the first matches records a, b, and the second matches records b, c. Then it is easy to convince oneself that the only possible orders of records are abc, and its reflection cba. For any other order, ab or bc would not be adjacent. Thus, we see that even raw access pattern leakage, by itself, reveals information about the order of records.

This gives rise to two questions: (1) how to *extract* order information from access pattern leakage; and (2) how to *quantify* the speed at which we learn order information. For the first point, most of the heavy lifting will be done by *PQ-trees*, a data structured tailored to solving precisely this problem, presented in Section 5.2. To tackle the second point, in Section 5.1, we introduce the notion of *sacrificial ϵ-approximate order reconstruction* (sacrificial ϵ-AOR). In Section 5.3, we present our ApproxOrder algorithm, which shows how PQ-trees can be used to target sacrificial ϵ-AOR; and analyze its query complexity using VC theory. In Section 5.4, we experimentally evaluate the bounds. In Section 5.5 we show how the attack can be extended to recover approximate record values, rather than just their order, and present experimental results.

Remark. Taking a step back, we note that what range query leakage reveals *by itself* is the order of records. For sparse databases, numerical information on *values* can be recovered when some form of additional *frequency* information is available to the adversary: such information can be the query distribution (as in Section 4), or an approximation of the database distribution (as in Section 5.5).

5.1 Definition of Sacrificial ϵ-AOR

Sacrificial ϵ-approximate order reconstruction (sacrificial ϵ-AOR) asks to recover the order of records, except for records that are within ϵN of each other ("approximate" recovery), and for records within ϵN of the endpoints 1 and N ("sacrificed" records).

We first introduce some notation: if A is a set of records, then the *diameter* of A is the largest difference between the values of any two records in A, i.e.: $\mathsf{diam}(A) \stackrel{\text{def}}{=} \max\{\mathsf{val}(b) - \mathsf{val}(a) : a, b \in A\}$. We let $<$ denote the order on records induced by their values, i.e. $r < s$ iff $\mathsf{val}(r) < \mathsf{val}(s)$. For two sets of records A and B, we write $A < B$ to denote $\forall a \in A, b \in B, a < b$.

An algorithm is said to achieve sacrificial ϵ-AOR iff it outputs disjoint subsets of records $A_1, \ldots, A_k$ such that:

1. $\forall i, \mathsf{diam}(A_i) < \epsilon N$.
2. $A_1 < \cdots < A_k$ holds up to reflection.
3. For all $r \notin \bigcup A_i$, $\mathsf{val}(r) \in [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$.

The definition implies that the algorithm reveals the order of the values of any two records, as soon as they are at least ϵN apart; except possibly for records whose value is within ϵN of 1 or N. If we set $\epsilon = 1/N$, sacrificial ϵ-AOR is equivalent to recovering the exact order of all records.

5.2 PQ-Trees

Our attack makes use of *PQ-trees* [BL76], a special structure that makes it possible to compactly represent the set of all orders on records compatible with a given access pattern leakage.

PQ-trees were introduced in [BL76], and are typically used as tools to solve other algorithmic problems (such as planarity testing for graphs). Given a ground set X with an unknown order, together

with a set $\mathcal{I}$ containing intervals of X for that order, a PQ-tree succinctly represents (i.e. in size linear in $|X|$) the set of all orderings of X that are compatible with $\mathcal{I}$. Moreover a PQ-tree can be updated on the fly: given a PQ-tree and a new interval I not previously contained in $\mathcal{I}$, the PQ-tree can be updated in linear time to only contain those orders that are compatible with $\mathcal{I} \cup \{I\}$. (Details of the update procedure are not relevant to our work.)

The structure of a PQ-tree is simple: the leaves of the tree are labelled by the ground set X, each element of X appearing in exactly one leaf. Internal nodes of the tree consist in two types of nodes: P-nodes and Q-nodes. Both types of nodes can have any number of children leaf or non-leaf nodes. P-nodes denote that the children of the node can be ordered in any way. For example, if $X = \{a, b, c\}$, then the tree $P(a, b, c)$ represents the set of all permutations of X. Q-nodes denote that the children of the node can only be ordered either as they appear in the tree, or as the reverse order (a.k.a. its reflection). For example, if $X = \{a, b, c\}$, then the tree $Q(a, b, c)$ represents the orders abc and cba. Nodes combine in the natural way: for example, the tree $Q(a, b, P(c, d))$ represents the possible orders $abcd$, $abdc$, $cdba$, $dcba$.

Initially, when no information whatsoever is known about the order, the PQ-tree consists in a single P-node, with all elements of the ground set as leaves. Conversely, once the order is fully determined, the tree consists in a single Q-node, whose leaves are either in the correct order or its reflection. This corresponds to the usual reflection symmetry, which cannot be broken by learning intervals (as replacing the order on X by its reflection leaves the set of intervals invariant).

In our setting, it is possible that the ground set X, which corresponds to the set of all record IDs, is not known in advance. However PQ-trees can be easily extended to handle that case: to do so, we start with a tree formed of a P-node with a single leaf labelled by a special element $\star$. Whenever the tree is updated with a new set I, if I contains new elements not already among the leaves of the current tree, these new elements are first added as siblings of $\star$. The tree is then updated with I as normal.

5.3 The ApproxOrder Attack

We now present the ApproxOrder algorithm, which targets sacrificial ϵ-AOR. Pseudo-code is given in Algorithm 3. The pseudo-code uses the following notation: if S is a node of a PQ-tree $\mathcal{T}$, then the leaves of S are defined as the leaves of the subtree rooted at S, and denoted $\mathsf{leaf}(S)$; and $\mathsf{root}(\mathcal{T})$ denotes the root of $\mathcal{T}$.

The idea is to first build the PQ-tree induced by the query access pattern leakage. The algorithm then locates the deepest node T in the tree such that the leaves below T contain a strict majority of all records. The algorithm returns as its output the set A_i of leaves below each child C_i of T, in the order of the children of T. Thus, the order between two records is learned by the adversary iff they appear below distinct children of T, and the order between the two records matches the order of the children of T below which they appear.

Analytically, our main result is as follows. The theorem assumes hypotheses h_2 and h_3, which will be presented below, and a uniform query distribution.

Theorem 5.1. *Let $\epsilon < 1/4$. Assume the database under attack satisfies h_2 and h_3. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon} + \frac{1}{\epsilon}\log\frac{1}{\delta}\right).$$

queries, ApproxOrder *achieves sacrificial ϵ-AOR with probability of success at least $1 - \delta$.*

The proof of Theorem 5.1 is given in Appendix D. For any fixed constant probability of success, the algorithm succeeds using only $\mathcal{O}(\epsilon^{-1}\log\epsilon^{-1})$ queries. As a direct corollary (setting $\epsilon = 1/N$), the

Algorithm 3 The ApproxOrder Attack for sacrificial ϵ-AOR.

ApproxOrder($\mathcal{Q}$):
Input: Set of queries $\mathcal{Q}$.
Output: Disjoint subsets of records $A_1, \ldots, A_k$.
$\mathcal{T} \leftarrow$ PQ-tree built from $\mathcal{Q}$.
$T \leftarrow$ FindNodeT($\mathcal{T}$, root($\mathcal{T}$))
$C_1, \ldots, C_k \leftarrow$ children of T (in order)
return leaf(C_1), ..., leaf(C_k)

FindNodeT($\mathcal{T}, S$):
Input: PQ-tree $\mathcal{T}$ and node S of $\mathcal{T}$.
Output: Deepest node $S' \leq S$ with $> R/2$ leaves.
$R \leftarrow |\text{leaf}(\text{root}(\mathcal{T}))|$
for each child C of S **do**
  if $|\text{leaf}(C)| > R/2$ **then**
    return FindNodeT($\mathcal{T}, C$)
  end if
end for
return S

expected number of queries before the PQ-tree collapses into a single Q-node, thus completely revealing the order up to reflection, is $\mathcal{O}(N \log N)$.

The overall idea is that after that number of queries, with high probability there exist certain queries whose endpoints partition $[1, N]$ into sufficiently small buckets while revealing the order between these buckets. By properties of PQ-trees, this situation implies the existence of a node within the PQ-tree that essentially directly reveals ϵ-approximate order (and that node can be easily located as the deepest node covering a majority of records). Moreover, the existence of the aforementioned queries inducing the partition is implied by an ϵ-net, so that ultimately the query complexity required for those queries to exist is directly derived from the ϵ-net theorem of VC theory.

The result does require some assumptions about the existence of records having certain values in the database, namely hypotheses h_2 and h_3. Hypothesis h_2 requires that there exist two records with values a and b such that $a, b \in [N/4, 3N/4]$, and $b - a \geq N/3$ (what really matters for the proof to go through is that a, b should be $\Omega(N)$ away from 1, N and each other); and additionally that there exist at least three records with values within $[\epsilon N, N+1-\epsilon N]$ that are more than ϵN away from each other (note that a and b can be two of these values). Hypothesis h_3 requires that a strict majority of all records have a value within $[\epsilon N, N+1-\epsilon N]$, and that no range of length ϵN contains the values of a (strict) majority of all records. On the face of it, h_2 and h_3 seem like they are restrictive in that they make several requirements on the database. But those requirements are quite mild. Both hypotheses essentially ask that the database should not be too concentrated over a few values. We have not encountered a real-world dataset that failed to satisfy those requirements. Further, only h_2 is actually required for the T node to exist and leak sacrificial ϵ-approximate order as claimed. The only point of hypothesis h_3, which is more demanding, is to ensure that that node is the deepest node covering a majority of records, so that it can be easily located. But that is a theoretical concern: in our experiments, the desired T node was usually in the first two levels of the tree. Thus, the practically relevant hypothesis is h_2, which only requires that the database should not be entirely concentrated near the endpoints.

In Appendix E.2, we prove that the query complexity of our algorithm is optimal within a constant factor. More precisely, any (unbounded) adversary achieving sacrificial ϵ-AOR for all databases must require $\Omega(\epsilon^{-1} \log \epsilon^{-1})$ queries.

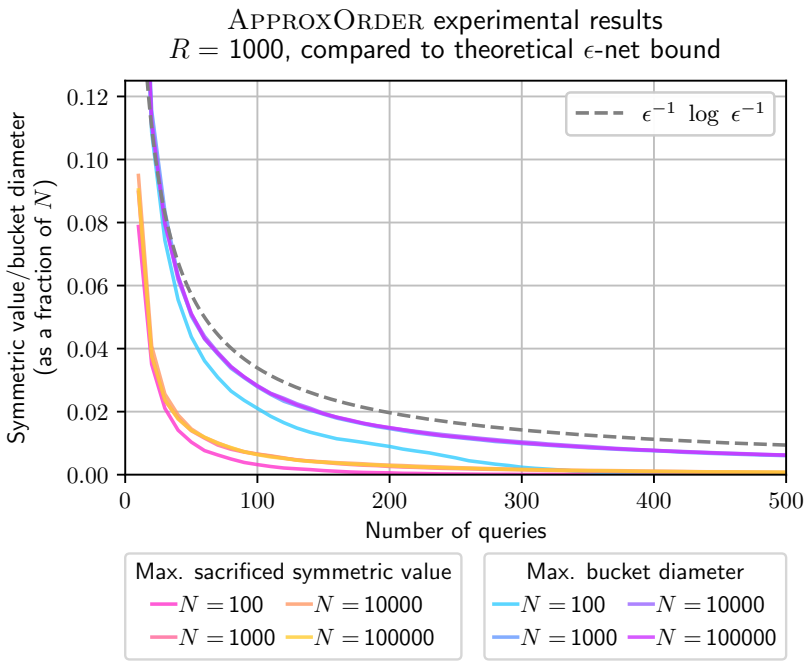

Figure 3: Maximum symmetric values of records not in buckets and maximum bucket diameters. Results averaged over 500 databases for each value of N.

5.4 Experimental Results

Assuming uniform queries, the ApproxOrder attack succeeds within $\mathcal{O}(\epsilon^{-1} \log \epsilon^{-1})$ queries (for any given constant probability of success $\eta < 1$). We experimentally evaluate the tightness of this bound for a fixed number of records R, and various numbers of possible values, N, so that we generate both dense and sparse databases. Record values are sampled uniformly at random, so hypotheses h_2 and h_3 were satisfied with high probability. Our results are averaged over 500 databases, each with 500 randomly sampled queries. We measured the results after every 10 queries, and therefore sometimes needed a heuristic to identify a likely candidate for the Q node when the number of queries is very small. When the root node was not a Q node, our experiments chose the first child Q node that contained at least a third of the records. As our results indicate, this node usually contained an overwhelming majority of the records.

The bottom group of lines in Figure 3 shows the maximum symmetric value (as a fraction of N) of any record that was not in one of the Q node's children buckets. When the ApproxOrder attack succeeds, the only records that are not necessarily in buckets are those with values in $[1, \epsilon N[$ or $]N + 1 - \epsilon N, N]$. If all records have been placed into buckets below the Q node, the maximum excluded symmetric value is set to 0. These results show that the theoretical upper bound holds, even when taking it with *all constants set to 1*, like in Section 4.4. The attack also behaves in the predicted *scale-free* way: changing N has little effect on empirical results.

The upper group of lines in Figure 3 shows the maximum diameter (as a fraction of N) of the Q node's child buckets. We compare this to the expected maximum diameter dictated by the ϵ-net bound, and see that convergence happens as quickly as predicted by the bound taken with *all constants set to 1*, as in the previous case. Again, results are scale-free.

Another way of interpreting these results is to ask, after a certain number of queries, for what ϵ have we achieved sacrificial ϵ-approximate order reconstruction? Our results indicate that the bottleneck is

the maximum bucket diameter, not the sacrificed values, so the upper group of lines in Figure 3 could be interpreted in this way.

Although our theoretical analysis for the ApproxOrder attack assumes a uniform query distribution, this assumption was *only* for the analysis and the attacker does not need to know the query distribution to carry out the attack. We consider now another more realistic distribution on queries, namely fixed-width range queries. Such queries are widespread in practice: for example, the industry-standard TPC-H contains six explicit fixed-width range queries. For a given number of possible values N and width $W \leq N$, there are $N + 1 - W$ such ranges: $[1, W], [2, W + 1], \ldots, [N + 1 - W, N]$. We experimentally evaluate how well the ApproxOrder attack performs for a dataset of $R = 1000$ records, $N = 10000$ possible values, and range queries of different widths. The results are in Figure 4. Unlike the case of uniform range queries, the limiting factor here in attaining ϵ-AOR is initially the too-high symmetric values of the sacrificed records. For small range widths (relative to the domain size, N), these results are to be expected: when only a few queries have been observed, the total number of possible values that have matched any query so far is limited, and thus the maximum symmetric value of a record that is not in a bucket may be high. After this initial period, the attack's performance follows the results of the uniform range query case and reflects the behaviour of $\epsilon^{-1} \log \epsilon^{-1}$.

5.5 From AOR to ADR

We now show how our approximate ordering attack can be combined with a model of the database distribution π (commonly called an *auxiliary distribution*) to mount powerful ϵ-ADR attacks. That is, we leverage our approximate ordering attack above to achieve approximate database recovery. Our attack is somewhat reminiscent of the LMP auxiliary distribution attack, with two major differences: (1) it does not require the additional rank leakage used by LMP, and (2) we can study its performance analytically.

We implemented the resulting ϵ-ADR attack and conducted experiments with several datasets representative of practical use cases of encrypted databases. As in the analysis of approximate order reconstruction, we will assume here that the query distribution is uniform *only* to make the exposition simpler—no part of our attack requires queries to be uniformly distributed. Our attack takes as input the output of any algorithm achieving ϵ-AOR. It also takes a model of the database distribution π (which needs only to approximate the true database distribution), the query distribution π_q, and the domain size N. It outputs an estimate for the underlying value of every record in the database. The pseudocode for the attack is given in Algorithm 4.

Attack intuition. Briefly, the attack uses the observation that, for every disjoint subset of records A_i ($i \in [1, \ldots, k]$) given by ϵ-AOR, some information about the *ranks* of the records (i.e. their positions in a sorted list of all records) in the subset is revealed. Because each sacrificed record could be before A_1 or after A_k the exact ranks are unknown, but lower and upper bounds can be computed.

There are three main questions to answer in building an attack from this observation. **(1)** How should the attack orient the A_i? **(2)** How many sacrificed records should go before A_1? **(3)** How should record values be estimated?

We now describe our attack and how it resolves these three questions. The first step, **record rank estimation** orients the A_i (question 1), estimates the number of sacrificed records less than A_1 (question 2), and produces an estimate of the range of ranks for each group. The second step, **partition estimation**, uses order statistics to estimate a range of *values* (i.e. a partition of $[N]$) for each range of ranks obtained in the previous step. The third step, **database estimation**, estimates a value for the records in each group given the estimated partition (question 3).

Record rank estimation. To recap, record rank estimation must (1) orient the A_is, (2) guess the

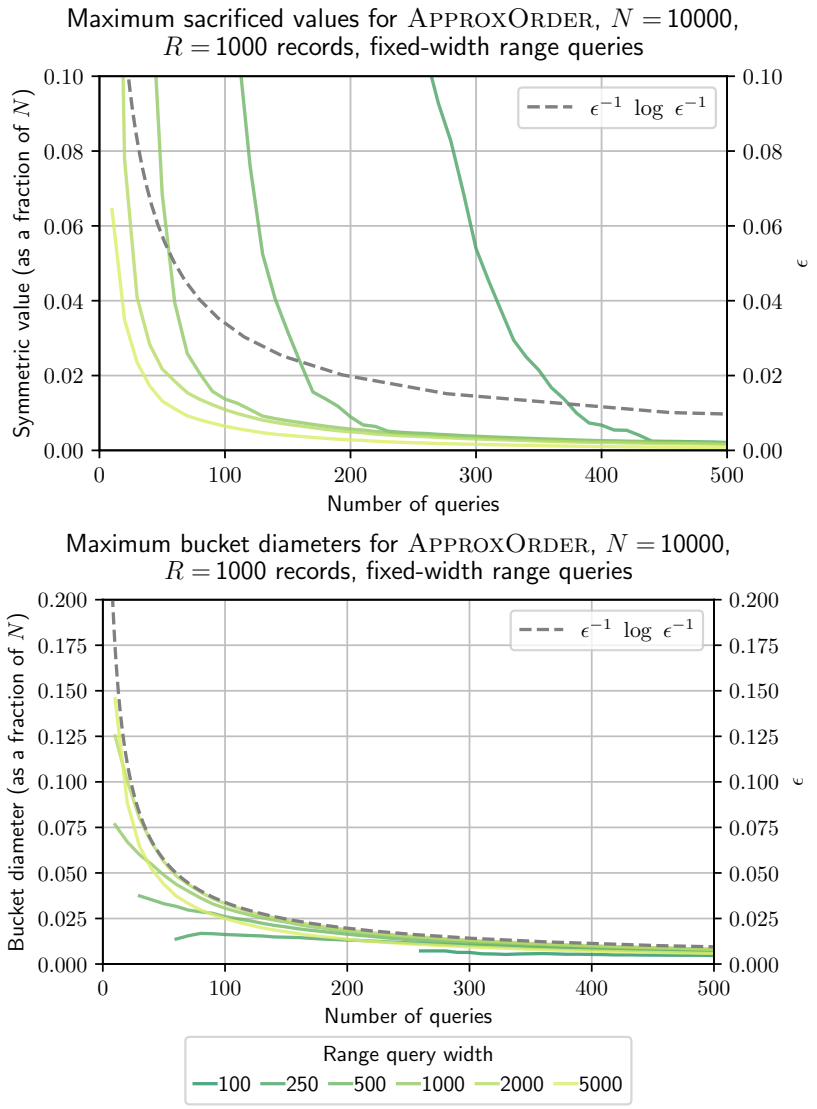

Figure 4: (**Top**) Maximum symmetric values of records not in buckets. (**Bottom**) Maximum bucket diameters. Results for fixed-width queries averaged over 500 databases for each value of range query width.

number of sacrificed records less than A_1, the first sorted group, and (3) estimate a range of ranks for each A_i. Our attack uses a heuristic for orienting the A_i (this is function OrientSubsets in Algorithm 4): first, measure the proportion of records above and below the middle group. Call these quantities $\hat{p}_a$ and $\hat{p}_b$, respectively. Then, compute the probability of a database value falling above (p_a) and below (p_b) the value $\lceil N/2 \rceil$. If $\hat{p}_a > \hat{p}_b$ and $p_a > p_b$, keep that orientation, else choose the other one. Though quite naive, below we will see this heuristic generally works well for real data distributions.

To guess the number of sacrificed records below the first sorted group (EstimateRank in Algorithm 4), we use a more principled approach. Observe that the sacrificed records are exactly those with values either lower than the value of the smallest left query endpoint (call this value $\ell_{\min}$) or higher than the value of the largest right endpoint (call this $r_{\max}$). Let $E_{ij} = (\ell_{\min} = i) \cap (r_{\max} = j)$ be the event that $\ell_{\min}$ is i and $r_{\max}$ is j. Let the number of sacrificed records be S and r_0 be a random variable denoting

the smallest rank for a record in A_1. The RV r_0 takes values in $[0, \ldots, S]$. Conditioned on E_{ij}, the distribution of the number of sacrificed records to the left of i and right of j is binomial with sample size S and probability of success $p_{ij} = \frac{\Pr[\,1,\ldots,i\,]}{\Pr[\,1,\ldots,i\,]+\Pr[\,j,\ldots,N\,]}$ where $\Pr[\,x,\ldots,y\,] = \sum_{k=x}^{y} \pi(k)$ and π is the auxiliary distribution. Thus, for any $r \in [0, \ldots, S]$,

$$\begin{aligned}\Pr[\,r_0 = r\,] &= \sum_{i\le j\in[N]} \Pr[\,r_0 = r \mid E_{ij}\,] \Pr[\,E_{ij}\,] \\ &= \sum_{i\le j\in[N]} \binom{S}{r} p_{ij}^{r} (1-p_{ij})^{S-r} \Pr[\,E_{ij}\,] \ .\end{aligned}$$

If the number of queries is Q and the query distribution is uniform, we can compute $\Pr[\,E_{ij}\,]$ via inclusion-exclusion as follows. First, define $f(x,y) = (x-y)(y-x+1)/N(N+1)$. Then

$$\begin{aligned}\Pr[\,E_{ij}\,] = {} & f(i,j)^Q - f(i,j-1)^Q \\ & - f(i+1,j)^Q + f(i+1,j-1)^Q \ .\end{aligned}$$

This is the only part of the attack that uses the uniform distribution on queries. If we let $\pi_q^{[i,j]}$ be the probability that a query is contained in the range $[i,j]$, with a non-uniform query distribution this expression would be the same except with $f(\cdot,\cdot)$ replaced by $\pi_q^{[\cdot,\cdot]}$. The value $\hat{r_0}$ output by function EstimateRank is then $\mathbb{E}[r_0] = \sum_{r=0}^{S} r \Pr[\,r_0 = r\,]$. The expression $\Pr[\,r_0 = r\,]$ has $\mathcal{O}(N^2)$ terms, which could make the attack scale poorly. Our implementation uses a heuristic to discard the terms for which $\Pr[\,E_{ij}\,]$ is very small, so computing $\mathbb{E}[r_0]$ (a one-time operation) takes only about eighty minutes in the worst case. Once we compute $\hat{r_0}$ we can find the lower and upper ranks for the A_i via addition; see the line assigning r_i in Algorithm 4.

Partition estimation. The output of the previous step is a lower and upper rank (call them $\mathrm{r_{lb}}$ and $\mathrm{r_{ub}}$) for each A_i. From this we will recover a lower and upper value ($\mathrm{ep_{lb}}$ and $\mathrm{ep_{ub}}$) used by the final step of the attack. To estimate values from ranks, we use order statistics. For a sample $X_1, \ldots, X_s$, the kth order statistic (denoted $X_{(k)}$, $k \in [1, \ldots, s]$) is the kth largest value in the sample. The probability $\Pr\left[\, X_{(k)} = u \,\right]$ has a simple formula: $\Pr\left[\, X_{(k)} = u \,\right] = \Pr\left[\, X_{(k)} \le u \,\right] - \Pr\left[\, X_{(k)} \le u-1 \,\right]$, where $\Pr\left[\, X_{(k)} \le u \,\right]$ is

$$\sum_{j=0}^{s-k} \binom{s}{j} (1 - \Pr[\,1,\ldots,u\,])^j \Pr[\,1,\ldots,u\,]^{s-j} \ .$$

Using this, we estimate $\mathrm{ep_{lb}} = \max_x \Pr\left[\, X_{(\mathrm{r_{lb}})} = x \,\right]$ and do the same for $\mathrm{ep_{ub}}$. For a fixed rank and varying x, the distribution of $\Pr\left[\, X_{(\mathrm{r_{lb}})} = x \,\right]$ converges to a Gaussian very quickly, so $\max_x \Pr\left[\, X_{(\mathrm{r_{lb}})} = x \,\right] \approx \mathbb{E}\left[X_{(\mathrm{r_{lb}})}\right]$.

Database estimation. This is the simplest step—the previous step outputs a partition $[1, \mathrm{ep}_1, \ldots, \mathrm{ep}_{|\mathcal{B}|}, N]$ of $[N]$ where the records in group b_i are between ep_i and ep_{i+1}, so we need only choose a value in $[\mathrm{ep}_i, \mathrm{ep}_{i+1}]$ to assign to the records in b_i. Since we are concerned with minimizing the absolute value of the difference between the true value and the guess, the natural choice is the median of the database distribution π, conditioned on the range $[\mathrm{ep}_i, \mathrm{ep}_{i+1}]$. In Algorithm 4 this is written as $\mathrm{RangeMedian}(\pi, \mathrm{ep}_i, \mathrm{ep}_{i+1})$.

Experiment setup and data. We implemented Algorithm 4 in Python 2.7 and ran all our experiments on an Ubuntu 16.04 desktop with an Intel Core i7-6700 CPU, clocked at 3.4GHz. We used an

Algorithm 4 Recovering values from approximate order.

Input: $\mathcal{T}_0$, R,π,π_q,N.
Output: $[x_1, x_2, \ldots, x_R]$ $(\forall i, x_i \in [N])$.

$\mathcal{T} \leftarrow \text{OrientSubsets}(\mathcal{T}_0)$
$\mathcal{B} \leftarrow \text{GetBuckets}(\mathcal{T})$
$\mathsf{e} \leftarrow \text{GetEnds}(\mathcal{T})$
$\hat{r_0} \leftarrow \text{EstimateRank}(\mathsf{e}, \pi_q, \pi)$
for all $b_i \in \mathcal{B}$ **do**
    $r_i \leftarrow r_{i-1} + |b_i|$
    $\text{ep}_i \leftarrow \arg\max_{k \in [N]} \Pr\left[X_{(r_i)} = k \right]$
    $\mathsf{med}_{b_i} \leftarrow \text{RangeMedian}(\pi, \text{ep}_{i-1}, \text{ep}_i)$
    for all $\mathsf{ind} \in b_i$ **do**
        $\text{cand}[\mathsf{ind}] = \mathsf{med}_{b_i}$
    end for
end for
return cand

existing C++ implementation [Gro11] of the PQ-tree data structure and used SWIG [swi18] to call it from Python.

We evaluate the attack on two datasets. The first is a database of registered pilots from the US government Federal Aviation Administration (FAA) [faa17]. It contains the ZIP code of residence for over 61,000 pilots nationwide. ZIP codes are five-decimal-digit numbers. The most significant digits reveal increasingly precise information about location—for example, the first three digits identify a neighborhood in large cities. For more information see [Wik18]. For this experiment we use US Census data about ZIP code population as an auxiliary model of the distribution. Though there are some high-probability ZIP codes, the distribution is overall fairly uniform. Further, the FAA ZIP codes are not well-modeled by the census data - their statistical distance is about 0.51.

The second is a database of California (CA) state public employee salaries from 2016. Salary data is sensitive both for cultural reasons and because of the possibility of blackmail. The database contains over 248,000 numbers between 0 and 762,000 US dollars. Most are salaries (i.e. at least 25,000 US dollars), but a sizeable fraction are in the low hundreds of dollars. We did not remove the low dollar amounts (as doing so could bias experiments in our favor) but we did truncate the few outliers over 500,000 US dollars. We used a database of around 120,000 New York (NY) state public employee salaries from the same year as auxiliary data for this experiment. Both NY and CA salary datasets are roughly Gaussian with means 73,000 and 67,000 respectively. Their statistical distance is about 0.19. Rather than use the full CA salary database, in this experiment we subsampled random databases of 10,000 salaries and averaged the results to better understand how the attack performs on smaller databases.

Results and discussion. Our attacks will measure accuracy as percent error, that is, if the true value of a record is u and the attack guesses v, (for $u, v \in [N]$ the error for that record is $|u - v|/N$. The baseline guessing attack for this accuracy measure is predicting the median of the database distribution for every record. Figure 5 shows the results of the ZIP code experiment averaged over 20 randomly-generated transcripts and the salary experiments averaged over 10 randomly subsampled databases each with 10 randomly-generated transcripts. We also show the baseline guessing accuracy. The variance was low in all our experiments with 25 or more queries. With only ten queries, the variance for the 75th percentile error is quite high, which we intuitively expect—with so few queries many groups of records will be large. The results in that table assume the A_i have been oriented correctly. Because ZIP

	Percent Error					
	25%		50%		75%	
# Queries	ZC	SAL	ZC	SAL	ZC	SAL
10	4	2	7	4	11	7
25	2	1	4	2	7	4
50	1	1	3	2	6	3
100	1	1	2	2	5	3
BL	15	2	27	5	37	9

Figure 5: Accuracy of Algorithm 4 on FAA ZIP codes ('ZC", $N = 9,999$) and CAL salaries ('SAL", $N = 500,000$): percentage of records recovered with error at most the listed percent of N. 'BL" refers to baseline guessing. Error is computed as $|(\text{actual}) - (\text{guessed})|/N$).

codes have a fairly flat distribution our heuristic procedure OrientSubsets chose the wrong orientation in about half of the experiments. The A_i were oriented correctly in all runs of the salary experiment. Since there are only two ways to orient the A_i an incorrect guess is mostly inconsequential. We do not include trials for which the PQ tree does not have a Q node at the first level. This happened only a few times in all experiments for ZIP codes. For salary experiments with 10 queries about one-quarter of the trials did not have a Q node at the first level. With 100 queries, about one-tenth of the trials did not. (The attacker can tell when there is no Q node and choose to see more queries before running the attack.)

The attack on ZIP codes performed extremely well. With only ten queries, we are able to guess the first digit correctly for over half the records on average. Concretely, about half the database records would have their state of residence partially revealed with only ten queries. With only one hundred queries, we recover the first two digits (or a small window with only a few possibilities) for a majority of the records in the database.

For the attack on salaries, the accuracy of the baseline guessing attack is artificially low because of the skew of the distribution—the max value (which we use as the denominator to compute percent error) is much larger than all but a tiny fraction of salaries. Thus, the baseline guessing attack having 5% error translates to 25,000 USD, but most salaries are within 25,000 USD of the median, so baseline guessing performs very poorly on most salaries. In contrast, our attack predicts a majority of the records in the database to within 2% error (10,000 USD) with only fifty queries, and with only 100 queries predicts a quarter with 1% error (5,000 USD).

6 Generalizing Approximate Reconstruction

We have seen how ϵ-nets and ϵ-samples can be used to build and analyze approximate reconstruction attacks on range queries. In this section, we abstract a core technical idea from those attacks – that records accessed the same way by most queries must be "close" – and show how it extends beyond range queries. We explore this in three ways: (1) by using learning theory to define a natural and general notion of distance relevant to access pattern attacks, (2) by showing how ϵ-nets are the right technical tool for analyzing the meaning of this distance for particular query classes, and (3) by using this distance notion to prove a general lower bound on the query complexity of any attack with access pattern leakage. To the best of our knowledge, our general lower bound is the first such proof ever given for this setting and illustrates a core finding of this work: the security impact of access pattern leakage for any class of queries is related to its VC dimension.

Distances induced by range queries. Let C_i be the set of range queries matching value i, and C_j, j. Then, the set of queries matching i XOR j (exactly one of i and j) is $\Delta(C_i, C_j)$, where Δ is the symmetric difference operator on sets, and the number of such queries is $\gamma(i,j) \stackrel{\text{def}}{=} |\Delta(C_i, C_j)|$. We can make three interesting observations about $\gamma(i,j)$. First, it is related to the numerical distance metric $|i-j|$ (though, importantly, they are not identical). Second, $\gamma(\cdot,\cdot)$ is itself a metric on $[N]$. Third, distance in this metric is approximately revealed by the access pattern leakage of range queries: if every query accesses either both or neither records i and j, then $\gamma(i,j)$ is likely to be small. These three properties were used extensively in our attacks on range queries, but they are not specific to range queries: we can abstract them using ideas from learning theory.

Distance, generally. Consider any class of queries $\mathcal{Q}$ on $[N]$ and distribution π over those queries, and consider the concept space $(\mathcal{Q}, \mathbb{C})$ with concepts $\mathbb{C} \stackrel{\text{def}}{=} \{C_i\}_{i\in[N]}$, where each $C_i \stackrel{\text{def}}{=} \{q \in \mathcal{Q} \,|\, q(i) = 1\}$. Each query is a point in this concept space, and there is a set corresponding to each possible value in $[N]$ containing the queries that match it. Now, define the *symmetric difference* concept space $(\mathsf{X}, \mathbb{C})^\Delta \stackrel{\text{def}}{=} (\mathsf{X}, \mathbb{C}^\Delta)$, where $\mathbb{C}^\Delta \stackrel{\text{def}}{=} \{\Delta(C_i, C_j)\}_{i,j\in[N]}$ and $\Delta(\cdot,\cdot)$ is the symmetric difference of the input sets. This new concept space contains, for each pair i, j, the queries which return exactly one of i, j. By Lemma A.2, the VC dimension of C^Δ is at most twice the VC dimension of $\mathbb{C}$. Next, define the function $\gamma_\pi(i,j) \stackrel{\text{def}}{=} \Pr_\pi[\,\Delta(C_i, C_j)\,]$. As above for range queries, where implicitly π was the uniform distribution, this defines a metric on $[N]$. To see that the triangle inequality holds, observe that for any i, j, and k, any query in $\Delta(C_i, C_j)$ is in $\Delta(C_i, C_k)$ or $\Delta(C_k, C_j)$. This allows us to generalize the use of ϵ-nets in ApproxOrder to *arbitrary* query classes. If the adversary observes a set of queries that is an ϵ-net for the symmetric difference concept space, then it must be the case that for any subset S of records with identical access pattern, the underlying values V of those records satisfy $\mathsf{diam}_\gamma(V) \stackrel{\text{def}}{=} \max_{i,j\in V} \gamma_\pi(i,j) \le \epsilon$.

Thus, if we simply group together records that have the same access pattern, then the existence of an ϵ-net provides an upper bound on the distance (with respect to the measure γ) of records in the same group. Essentially, access pattern leakage from *any* query class reveals a kind of approximate equality between the underlying values of the records in the database. This approximate equality depends both on the query class and the query distribution. For range queries, we used this approximate equality to build the ApproxOrder attack and reveal a great deal of information with few queries. However, closeness in the metric γ may not be practically interesting for all query classes and distributions: for example, access pattern leakage from the "query class" which is sampled uniformly at random from $2^{[N]}$ is unlikely to reveal anything interesting. Nevertheless, for many query classes used in practice, closeness in this distance metric can lead to serious privacy breaches. For example, for prefix queries, two values being close in this metric implies they have a common prefix. We will show a simple attack that allows an adversary to reveal which records in the database are approximately equal according to the distance metric γ_π.

Approximate equality attack. Consider a set of queries $\mathcal{Q}$, possible record values $[N]$, and resulting concept space $(\mathcal{Q}, \mathbb{C})$, whose VC dimension d we assume is finite and ≥ 2. Let π_q be any distribution over $\mathcal{Q}$. The attack takes as input records $\{r_1, r_2, \ldots, r_R\}$ along with a 0-1 matrix AP with R rows and Q columns, where $\mathsf{AP}_{ij} = 1$ iff query j returns record i. The attack views each row of the matrix as a number in $[0, 2^Q - 1]$ and outputs a partition by grouping all records with the same number. Let $g_i = \{r_1^i, \ldots, r_k^i\}$ be any such group, and let $V = \{v_1, \ldots, v_k\}$ be the underlying values of these records. An application of the ϵ-net theorem lets us immediately conclude that $\Pr_{\pi_q}[\,\mathsf{diam}_\gamma(V) \le \epsilon\,] > 1 - (2Q)^d 2^{-\epsilon Q/2}$, and this bound holds for all groups simultaneously.

6.1 Prefix and Suffix Queries

Next, we show how to instantiate the approximate equality attack for a practically relevant query class. For a set $\Sigma^{\leq \ell}$ of all strings with length $\leq \ell$ from some alphabet Σ, define a *prefix query* q to be a string in the set $\cup_{j=1}^{\ell} \Sigma^j$. In text search, prefix queries are usually indicated by a trailing asterisk "*". For any element $j \in \Sigma^{\leq \ell}$, define the predicate $q(j)$ to be 1 if either $q = j$ or q is a prefix of j, and 0 otherwise. As an example, take the database $\{\text{cat}, \text{carbon}\}$. A prefix query "c*" on these two values would return both, but "carb*" would return only the second one.

Although prefix queries are technically a subset of range queries, there are three crucial differences which obviate the use of previous attacks on range queries: prefix queries do not reveal order, they cannot overlap without one query being contained in the other, and the number of queries matching any fixed string is constant. (Replacing "prefix" with "suffix" in the discussion above gives an identical query class that matches strings based on a suffix instead of a prefix. Our discussion and attacks easily translate to suffix queries, so we dispense with a separate discussion for them.)

In the the symmetric difference concept space for prefix queries, the concepts $\Delta(C_i, C_j)$ for $i, j \in \Sigma^{\leq \ell}$ are the queries that are prefixes of exactly one of i or j. If i and j themselves have a common prefix, though, some prefix queries will match both i and j. More precisely, if i and j have a length-k common prefix, then $|\Delta(C_i, C_j)| = (|i| - k) + (|j| - k)$. Informally, if the adversary notices that two records are always accessed together or not at all, then it can infer that they share a long common prefix. We will describe how to formalize this intuition with ϵ-nets. Further, if the adversary has a model of the database distribution, it can use frequency analysis to learn the characters of each record, one at a time (reminiscent of the climax of the science-fiction movie *WarGames*).

A *WarGames* attack on prefix search. Most modern text and web search systems support prefix queries on unstructured data [es18], and they are ubiquitous in software-as-a-service (SaaS) products like Salesforce, ServiceNow, and Dropbox [sal18, sno18, dro18]. A common [dro18, sal18] design pattern for these systems is to send a prefix query for every character the user types in the search bar. Since users may find their desired result without finishing their query, the distribution of queries is heavily biased towards shorter prefixes.

Our attack in this setting is simple. First, the adversary runs the approximate equality attack described above, obtaining a partition of the records in the database. Then, for each record, it takes the union of all query results containing that record. Here is where we apply the generalized distance notion discussed earlier: with an ϵ-net, we can ensure that each group in the partition contains records with at least a length-one common prefix, and that the unions we form afterwards are exactly the sets of records with the same first character. The first character of each record is then recovered via frequency analysis, and the attack is iterated to learn the second character, then the third, etc.

Analyzing the attack. We model the queries as being sampled via a two-step process. First, a prefix length ℓ_q is sampled from a Zipf distribution on $[\ell]$. (Recall that the standard Zipf distribution on ℓ elements has $\Pr[\, i \,] = (1/i)/H_\ell$, where $H_\ell = \sum_{m=1}^{\ell} 1/m$ is the ℓth harmonic number.) Then, the query is sampled as a uniformly random element of Σ^{ℓ_q}. Call this distribution over queries π_{ts}.

We first consider, for two words $i, j \in \Sigma^{\leq \ell}$, how the length of i and j's common prefix relates to $\Pr_{\pi_{ts}}[\, \Delta(C_i, C_j) \,]$. If i and j have different lengths and share a length-k prefix, then $\Pr_{\pi_{ts}}[\, \Delta(C_i, C_j) \,] = \frac{1}{H_\ell}\left(\sum_{m=k+1}^{|i|} 1/(m|\Sigma|^m) + \sum_{m=k+1}^{|j|} 1/(m|\Sigma|^m)\right)$. Let $\ell_{\min}$ be the length of the shortest string. If the queries observed by the adversary are an ϵ-net for the symmetric difference concept space and for $\epsilon = \frac{1}{H_\ell}\sum_{m=1}^{\ell_{\min}} 1/(m|\Sigma|^m)$, then, for all i, j having no common prefix, we have the distance $\gamma_{\pi_{ts}}(i, j) > \epsilon$ and a query accessing i and j differently must have occurred. The VC dimension of this concept space is at most 4, so $\mathcal{O}(\frac{1}{\epsilon}\log\frac{1}{\epsilon\delta})$ queries suffice for this attack to recover the first character of every record

with probability at least $1 - \delta$. This same analysis can be iterated for the rest of the characters.

Experiments. We implemented the attack using last name data from the Fraternal Order of Police (FOP) database dump, posted online in 2016. It contains the personal information of over 600,000 law enforcement officers in the United States. For auxiliary data, we used public US Census statistics [Bur16] on last name frequencies. We also ran the attack on the FAA ZIP code dataset from the experiments in Section 5.5, but it performed quite poorly, primarily due to the auxiliary data being a poor model of the ZIP code distribution.

In 9 out of 10 trials with only 500 prefix queries sampled according to the distribution described above, we were able to partition the records into groups with at least a one-character prefix in common. The mean number of queries required to do this was 315. Once we obtain this partition, we recovered the first character for over 70% of the last-name records. With the same number of trials for 40,000 queries, we recovered the first and second characters of over 55% of the last-name records. With 3 million queries, we recovered the first three characters for over 40% of last-name records, and we recovered roughly 1,500 three-character last names exactly. The sample complexities given by the ϵ-net theorem above are 1,491, 120,000, and 6 million for recovering 1, 2, and 3 characters—much higher than our experiments indicated. As we saw above, applying these results can give loose bounds but the "true" constants are usually small.

This attack on prefix queries can be improved. Our goal was not simply to construct an accurate reconstruction attack for prefix queries, but to demonstrate the power of the learning-theoretic approach in building and analyzing reconstruction attacks. We can generalize the prefix attack to obtain the three basic steps for this approach: (1) define a concept space and a metric, (2) use an ϵ-net to analyze the number of queries needed to learn approximate equality, then (3) perform an attack on the information about values revealed by approximate equality. We note also that standard results [MU17] on intersections and unions of concept classes can extend this approach to *composite* query classes (e.g. a SQL query which intersects the result of a range query on one column and a prefix query on another).

6.2 A General Lower Bound on Attacks

The metric γ is defined for *any* query class, and in many cases this leads to privacy implications: for range queries, it is closely related to the distance between record values; for prefix queries, the length of the longest common prefix. A general approximate reconstruction attack should recover values that are close (for γ) to the actual record values, and lower bounds on closeness (for γ) should imply lower bounds on the accuracy of any approximate reconstruction attack. The following theorem gives one such lower bound on the number of queries necessary for any approximate reconstruction attack on any query class, as a function of the desired accuracy ϵ and the VC dimension d of the query class.

Theorem 6.1. *Let $\mathcal{Q}$ be a class of queries on $[N]$, π_q a query distribution, and $\mathcal{C} = (\mathcal{Q}, \mathbb{C})$ the associated concept space with VC dimension $d > 1$. Let $\gamma(i,j) \stackrel{\text{def}}{=} \Pr_{\pi_q}[\,\Delta(C_i, C_j)\,]$ be the distance metric induced on $[N]$ by $\mathcal{Q}$ and π_q. Consider any algorithm that takes as input a database of size R with elements in $[N]$, together with the access pattern leakage of m queries sampled from π_q, and outputs an approximation DB' such that $\gamma(DB[i], DB'[i]) \leq \epsilon$ for all $i \in [1, \ldots, R]$, with probability of success at least $1 - \delta$ (over the choices of queries from π_q). Then m is in $\Omega(\frac{d}{\epsilon} + \frac{1}{\epsilon}\log\frac{1}{\delta})$.*

This result is a direct application of PAC learning theory: an algorithm that takes any database as input and outputs a DB' satisfying the stated condition is a PAC learner for the concept space $\mathcal{C}$ defined in the theorem statement. We can thus apply a general lower bound [EHKV89] on the sample complexity of PAC learning to conclude that m must be in $\Omega(\frac{d}{\epsilon} + \frac{1}{\epsilon}\log\frac{1}{\delta})$. With a smaller number of queries m, there will be, with probability at least δ, two values in $[N]$ whose distance γ is strictly greater than ϵ, but which every query given to the algorithm accessed in the same way.

This result is not easy to interpret, so we briefly reflect on its implications. First, it holds even if the adversary knows the exact query distribution and database distribution. Next, note that the same lower bound holds for the existence of an ϵ-net: if the queries fail to form an ϵ-net for the metric γ, then some records that are more than ϵ apart in γ cannot be separated based on access pattern. Since any approximate attack should be able to distinguish such records, in some sense this approximate equality attack is a *minimal* approximate attack. For example, consider both our sacrifical ϵ-ADR and -AOR attacks from Sections 4 and 5. Recovering approximate values or a partition into buckets with small diameters implies we are able to group together approximately-equal records. From this perspective, the lower bound on the existence of an ϵ-net for γ may be interpreted as a lower bound on the number of queries necessary for *any* form of approximate attack for which γ is a relevant notion of distance—not only an approximate attack attempting to recover values, as in Theorem 6.1.

7 Conclusions

This work initiates the application of learning theory to attacks on encrypted databases which leak access patterns. Our learning-theoretic viewpoint lets us build and analyze approximate reconstruction attacks which are both nearly-optimal in query complexity and effective on real data. We believe this work represents an exciting first step towards building a cohesive theory of security in the presence of access pattern leakage. Towards this, we recommend two main research directions for future work to pursue: first, extend our attacks to other query types of practical importance like edit distance, wildcard, and substring queries. Second, study and apply other results from learning theory, such as active or online learning, to access pattern leakage attacks and defenses.

Acknowledgments

Portions of this work were written while the first author was visiting Royal Holloway, University of London. This work was supported by NSF Graduate Research Fellowship DGE-1650441, the European Union's Horizon 2020 grant ECRYPT-NET (H2020 643161), ERC Project aSCEND (H2020 639554), and EPSRC Grant EP/M013472/1.

References

[BEHW86] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred Warmuth. Classifying learnable geometric concepts with the Vapnik-Chervonenkis dimension. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, pages 273–282, New York, NY, USA, 1986. ACM.

[BGC+18] Vincent Bindschaedler, Paul Grubbs, David Cash, Thomas Ristenpart, and Vitaly Shmatikov. The tao of inference in privacy-protected databases. *Proc. VLDB Endow.*, 11(11):1715–1728, July 2018.

[BL76] Kellogg S. Booth and George S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *J. Comput. Syst. Sci.*, 13(3):335–379, 1976.

[Bur16] US Census Bureau. US Census Bureau name statistics. https://www.ssa.gov/OACT/babynames/, 2016.

[CGPR15] David Cash, Paul Grubbs, Jason Perry, and Thomas Ristenpart. Leakage-abuse attacks against searchable encryption. In *CCS*, 2015.

[DL] Sanjoy Dasgupta and John Langford. A tutorial on active learning. http://hunch.net/~active_learning/.

[dro18] Dropbox, 2018. https://www.dropbox.com.

[EHKV89] Andrzej Ehrenfeucht, David Haussler, Michael Kearns, and Leslie Valiant. A general lower bound on the number of examples needed for learning. *Inf. Comput.*, 82(3):247–261, September 1989.

[es18] ElasticSearch, 2018. https://www.elastic.co/.

[faa17] Federal Aviation Administration pilot database, 2017. https://www.faa.gov/regulations_policies/pilot_records_database/.

[FVY+17] Benjamin Fuller, Mayank Varia, Arkady Yerukhimovich, Emily Shen, Ariel Hamlin, Vijay Gadepally, Richard Shay, John Darby Mitchell, and Robert K. Cunningham. SoK: Cryptographically protected database search. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 172–191, May 2017.

[GMN+16] Paul Grubbs, Richard McPherson, Muhammad Naveed, Thomas Ristenpart, and Vitaly Shmatikov. Breaking web applications built on top of encrypted data. In *CCS*, 2016.

[Gro11] Greg Grothaus. General implementation of the PQ-tree algorithm, 2011. https://github.com/Gregable/pq-trees.

[GSB+17] Paul Grubbs, Kevin Sekniqi, Vincent Bindschaedler, Muhammad Naveed, and Thomas Ristenpart. Leakage-abuse attacks against order-revealing encryption. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 655–672, May 2017.

[HW86] David Haussler and Emo Welzl. Epsilon-nets and simplex range queries. In *Proceedings of the Second Annual Symposium on Computational Geometry*, SCG '86, pages 61–71, New York, NY, USA, 1986. ACM.

[JR13] Jonathan L. Dautrich Jr. and Chinya V. Ravishankar. Compromising privacy in precise query protocols. In *Joint 2013 EDBT/ICDT Conferences, EDBT '13 Proceedings*, pages 155–166. ACM, 2013.

[KKNO16] Georgios Kellaris, George Kollios, Kobbi Nissim, and Adam O'Neill. Generic attacks on secure outsourced databases. In *CCS*, 2016.

[KPT18] Evgenios M. Kornaropoulos, Charalampos Papamanthou, and Roberto Tamassia. Data recovery on encrypted databases with k-nearest neighbor query leakage. Cryptology ePrint Archive, Report 2018/719, 2018. https://eprint.iacr.org/2018/719.

[KVV94] Michael J. Kearns, Umesh Virkumar Vazirani, and Umesh Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

[LMP18] Marie-Sarah Lacharité, Brice Minaud, and Kenneth G. Paterson. Improved reconstruction attacks on encrypted data using range query leakage. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 297–314, May 2018.

[MU17] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomization and Probabilistic Techniques in Algorithms and Data Analysis*. Cambridge University Press, New York, NY, USA, 2nd edition, 2017.

[NKW15] Muhammad Naveed, Seny Kamara, and Charles V Wright. Inference attacks on property-preserving encrypted databases. In *CCS*, 2015.

[sal18] Salesforce.com, 2018. https://www.salesforce.com.

[Sau72] N. Sauer. On the density of families of sets. *Journal of Combinatorial Theory, Series A*, 13(1):145–147, 1972.

[sno18] ServiceNow, 2018. https://www.servicenow.com.

[STW99] John Shawe-Taylor and Robert C. Williamson. Generalization performance of classifiers in terms of observed covering numbers. In *Computational Learning Theory*, pages 274–285, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[swi18] Simplified wrapper and interface generator (SWIG), 2018. http://www.swig.org/.

[Val84] Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 1984.

[VC71] V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971. Translation by B. Seckler.

[Wik18] Wikipedia contributors. ZIP code — Wikipedia, the free encyclopedia. https://en.wikipedia.org/wiki/ZIP_Code, 2018.

[ZKP16] Yupeng Zhang, Jonathan Katz, and Charalampos Papamanthou. All your queries are belong to us: The power of file-injection attacks on searchable encryption. In *25th USENIX Security Symposium, USENIX Security 16*, pages 707–720, 2016.

A Query Complexity for Reconstruction with Known Queries

In this section, we prove that the dependency on the number of records R can be removed from Theorem 3.1. This is formalized in Theorem A.1.

Recall the following notation from Section 3. We can think of a database DB with R records having values in $[N]$ as being a vector of length R with values in $[N]$; the value of record j is $DB[j]$. We construct a concept space $\mathcal{C} = (\mathcal{Q}, \mathbb{C})$ as follows. The points in the ground set are the possible queries $q \in \mathcal{Q}$. We write $q(i) = 1$ when value $i \in [N]$ matches query q. We set $C_i = \{q \in \mathcal{Q} \,|\, q(i) = 1\}$. We then define $\mathbb{C} = \{C_i : i \in [N]\}$.

Theorem A.1. *Let $\mathcal{Q}$ be a class of queries and $\mathcal{C} = (\mathcal{Q}, \mathbb{C})$ be the concept space constructed as above. Let π_q be any distribution over $\mathcal{Q}$. Let d be the VC dimension of $\mathcal{C}$, and assume d is finite. Then, there is an adversary such that for any database DB, given as input $m \in \mathcal{O}(\frac{d}{\epsilon} \log \frac{d}{\epsilon\delta})$ queries sampled from π_q and their access pattern leakage on DB, the adversary outputs a database DB' such that $\Pr_{\pi_q}[\, q(DB[j]) \neq q(DB'[j]) \,] \leq \epsilon$ holds simultaneously for all $j \in [R]$, with probability at least $1 - \delta$.*

Remark. The statement above is identical to Theorem 3.1, except the probability of success is $1 - \delta$ instead of $1 - R\delta$. In reality, there is small price to pay: the required number of queries is

increased, essentially by a factor (at most) two. This is not reflected in the statement since the factor two disappears into the $\mathcal{O}()$ notation.

Before proving Theorem A.1, we start with a short lemma. Given a concept class $(\mathsf{X}, \mathbb{C})$, $\mathbb{C}^\Delta$ denotes the set of symmetric differences of elements of $\mathbb{C}$; that is:

$$\mathbb{C}^\Delta \stackrel{\text{def}}{=} \{\Delta(A, B) : A, B \in \mathbb{C}\}$$

where $\Delta(\cdot, \cdot)$ denotes the symmetric difference of the input sets.

Lemma A.2. *Let $(\mathsf{X}, \mathbb{C})$ be an arbitrary concept class with finite VC dimension. Then the VC dimension of $(\mathsf{X}, \mathbb{C}^\Delta)$ is also finite. Moreover it is at most twice the VC dimension of $(\mathsf{X}, \mathbb{C})$.*

Proof. The proof of the lemma is identical to standard proofs of the same result for e.g. unions. Namely, let d denote the VC dimension of $(\mathsf{X}, \mathbb{C})$. Then the growth function of $(\mathsf{X}, \mathbb{C})$ is $\mathcal{O}(n^d)$ as a function of the number of points n (see Section 2). Hence given n points in X, $\mathbb{C}$ induces $\mathcal{O}(n^d)$ subsamples, hence symmetric differences of two elements in $\mathbb{C}$ can only induce $\mathcal{O}(n^{2d})$ subsamples, hence the VC dimension of $\mathbb{C}^\Delta$ is at most $2d$. □

Remark. This bound is tight. This can be seen by starting from an arbitrary concept space $(\mathsf{X}, \mathbb{C})$ of VC dimension d, and forming the concept space $(\mathsf{X}_2, \mathbb{C}_2)$ with $\mathsf{X}_2 = \mathsf{X} \times \{0, 1\}$ and $\mathbb{C}_2 = \{C \times \{i\} : C \in \mathbb{C}, i \in \{0, 1\}\}$. It can be checked that the VC dimension of $(\mathsf{X}_2, \mathbb{C}_2)$ is d, and the VC dimension of $(\mathsf{X}_2, \mathbb{C}_2^\Delta)$ is $2d$.

We now turn to the proof of Theorem A.1. The proof is no longer a "generic" reduction to PAC learning. Instead, the proof uses the ϵ-net theorem (Theorem 2.5) directly.

Proof of Theorem A.1. We choose as adversary any adversary that outputs any database DB' that is consistent with the observed leakage. Such a database exists, since DB must be consistent with its own leakage. (As with general PAC learning, there is no guarantee that the adversary is efficient.) Next we pick a sample size m large enough to ensure that the sample is an ϵ-net for the concept space $(\mathsf{X}, \mathbb{C}^\Delta)$. By Lemma A.2, the VC dimension of that concept space is at most $2d$. By the ϵ-net theorem, it follows that $m \in \mathcal{O}(\frac{d}{\epsilon} \log \frac{d}{\epsilon\delta})$ suffices to ensure that the sample forms an ϵ-net with probability at least $1 - \delta$. We claim that this choice of adversary and m satisfies the conclusion of the theorem.

To see this, assume the sample is an ϵ-net, which holds with probability at least $1 - \delta$. Let DB' be the database output by the adversary. Assume towards contradiction that there exists $j \in [R]$ such that $\Pr_{\pi_q}[\, q(DB[j]) \neq q(DB'[j])\,] > \epsilon$. This last expression is equivalent to saying that the measure of $\Delta(C_{DB[j]}, C_{DB'[j]})$ according to π_q is greater than ϵ. Since the sample is an ϵ-net for $\mathbb{C}^\Delta$, it must contain a point in $\Delta(C_{DB[j]}, C_{DB'[j]})$. This point is a (known) query that matches record j in DB but not in DB', or conversely. Hence the database DB' output by the adversary is not consistent with the sample, a contradiction. □

B Query Complexity of GeneralizedKKNO

B.1 Algorithm

Throughout, we assume ϵN is a strictly positive integer[1]. For simplicity (to avoid the proliferation of rounding) we also assume $N/4$ is an integer. Recall that for $k \in [1, N]$, A_k denotes the set of ranges in

[1]This is without loss of generality, because the relevant quantity in the definition of sacrificial ϵ-ADR is $\lceil \epsilon N \rceil$. That is, ϵN can be replaced by $\lceil \epsilon N \rceil$ everywhere in the definition, and this does not affect whether it is satisfied.

$[1, N]$ that contain the value k. The probability that a uniformly random range contains the value k, i.e. falls within A_k, is:

$$p(k) \stackrel{\text{def}}{=} \Pr(A_k) = \frac{2}{N(N+1)} k(N+1-k).$$

We have:

$$p(k+\delta) - p(k) = \frac{2}{N(N+1)} \delta(N+1-2k-\delta). \tag{2}$$

The full GeneralizedKKNO algorithm is given in Algorithm 5. Note that arg min is computed over the integers. On the other hand, throughout the analysis, we extend the functions p and d to be defined over rationals rather than just integers, using the same polynomial expressions.

Algorithm 5 ADR Algorithm GeneralizedKKNO.

GeneralizedKKNO($\mathcal{Q}$):
Input: Set of queries $\mathcal{Q}$.
Output: Function est-val approximating val.
1: est-symval $\leftarrow$ GetEstsymval($\mathcal{Q}$) ▷ Step 1
2: est-val $\leftarrow$ GetEstval($\mathcal{Q}$, est-symval) ▷ Step 2
3: **return** est-val

GetEstsymval($\mathcal{Q}$):
1: **for** each record r **do**
2: $c(r) \leftarrow |\{q \in \mathcal{Q} : r \in q\}|/|\mathcal{Q}|$
3: est-symval$(r) \leftarrow \arg\min_{k \in [N/2]} |p(k) - c(r)|$
4: **end for**
5: **return** est-symval

GetEstval($\mathcal{Q}$, est-symval):
1: $Q \leftarrow |\mathcal{Q}|$
2: $r_A \leftarrow \arg\min_r |\text{est-symval}(r) - N/4|$ ▷ Anchor record
3: est-val$(r_A) \leftarrow$ est-symval(r_A)
4: **for** each record $r \neq r_A$ **do**
5: $c'(r) \leftarrow |\{q \in \mathcal{Q} : r_A, r \in q\}|/|\mathcal{Q}|$
6: **if** $c'(r) > \min\{\text{est-val}(r_A), \text{est-symval}(r)\}/N$ **then**
7: est-val$(r) \leftarrow$ est-symval(r)
8: **else**
9: est-val$(r) \leftarrow N + 1 -$ est-symval(r)
10: **end if**
11: **end for**
12: **return** est-val

Finally, we recall that *sacrificial ϵ-approximate database reconstruction* (sacrificial ϵ-ADR) is said to succeed iff one of the following two events occur:

1. For every record r such that $\text{val}(r) \in [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$, $|\text{est-val}(r) - \text{val}(r)| < \epsilon N$.
2. For every record r such that $\text{val}(r) \in [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$, $|\text{est-val}(r) - (N + 1 - \text{val}(r))| < \epsilon N$.

We now state our main result.

Theorem B.1. *Let $\epsilon < 1/4$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^4}\log\frac{1}{\epsilon}+\frac{1}{\epsilon^4}\log\frac{1}{\delta}\right).$$

queries, GeneralizedKKNO *achieves sacrificial ϵ-ADR with probability of success at least $1-\delta$.*

The proof of the theorem is given in the next section.

B.2 Analysis

Throughout, $\epsilon < 1/4$ denotes the target precision, and $\mathcal{Q}$ the set of queries. We assume that the anchor record has a value less than $N/2$. If not, it means we will ultimately recover the reflection of the record values, instead of the record values themselves, but it does not affect the success of the algorithm. Indeed it simply means we will ultimately satisfy condition (2) in the definition of sacrificial ϵ-ADR instead of condition (1). (This assumption is where the two possible success conditions are separated.)

Our goal in this section is to prove Theorem B.1. The overall structure of GeneralizedKKNO is as follows. In Step 1 of the algorithm, GetEstsymval approximates the symmetric value of every record. In Step 2, GetEstval determines, for each record, on which side of the reflection symmetry around $(N+1)/2$ the record falls (relative to a chosen anchor record). We analyze each of these two steps in turn.

Step 1. *Analysis of* GetEstsymval. First, we introduce the following concept space, which will play a crucial role in the analysis. Define the ground set X as the set of all ranges in $[1,N]$, and the concept set $\mathbb{C}$ as the set of all A_k's, i.e. $\mathbb{C} \stackrel{\text{def}}{=} \{A_k : k \in [N]\}$.

Lemma B.2. *The growth function of $(\mathsf{X},\mathbb{C})$ is $2n$, and its VC dimension is 2.*

Proof. Let S be any sample of n ranges, say $S = \{[a_1,b_1],\ldots,[a_n,b_n]\}$. Consider the set of points Y defined as follows:

$$Y := \{1, N+1\} \cup \{a_i\}_{1\le i\le n} \cup \{b_i+1\}_{1\le i\le n}.$$

It contains some $\ell \le 2n+2$ elements, $y_1 \le \ldots \le y_\ell$, where $y_1 = 1$ and $y_\ell = N+1$. Notice that whenever two distinct integers k_1 and k_2 lie in the same interval $[y_i, y_{i+1}[$, they must match exactly the same subset of queries in S, since all queries match both or neither of them. Formally, $S \cap A_{k_1} = S \cap A_{k_2}$. Given that ℓ points create $\ell - 1$ intervals, the growth function is at most $\ell - 1$.

If $\ell \le 2n+1$, then $\ell - 1 \le 2n$, so we are done. If $\ell = 2n+2$, then all left endpoints a_i must be strictly greater than $y_1 = 1$ and all right endpoints $b_i + 1$ must be strictly less than $y_\ell = N+1$. In that case, for any $k_1 \in [y_1, y_2[$ and $k_2 \in]y_{\ell-1}, y_\ell]$, $S \cap A_{k_1} = S \cap A_{k_2} = \varnothing$, so the growth function is again at most $2n$.

We now show this bound is tight by constructing a set S of n ranges such that $S \cap A_k$ takes $2n$ distinct values as k spans $[N]$. Consider the set of range queries $S = \{[1,n],[2,n+1],\ldots,[n,2n-1]\}$. For $N \ge 2n$, the resulting set Y is $\{1,2,\ldots,2n,N+1\}$, with $|Y| = \ell = 2n+1$. Each of the sets A_k for $k = 1,\ldots,2n$ induces a distinct subsample of S: for k in $\{1,\ldots,n\}$, the induced subsample is $\{[1,n],\ldots,[k,n+k-1]\}$, while for $k \in \{n+1,\ldots,2n-1\}$, it is $\{[k-n+1,k],\ldots,[n,2n-1]\}$, and for $k = 2n$, it is the empty subsample $\varnothing$.

Hence, the growth function of $(\mathsf{X},\mathbb{C})$ is $2n$ (assuming $N \ge 2n$). Since $\mathbb{C}$ induces at most $2n$ subsamples in a sample of size n, and shattering a sample requires 2^n subsamples, the size of the largest sample that can be shattered, and thus the VC dimension of this concept space, is 2. □

We now turn to analyzing GetEstsymval. In the main body of this work, we have argued that GetEstsymval succeeds within $\mathcal{O}(\epsilon^{-4}\log\epsilon^{-1})$ queries. We now formalize and prove that statement.

Lemma B.3. *Let $\delta > 0$. Let $c_1 > 0$ be a constant. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^4}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^4}\log\frac{1}{\delta}\right).$$

queries, the following event simultaneously *holds for all records r, except with probability at most δ:*

$$|\mathsf{est\text{-}symval}(r) - \mathsf{symval}(r)| < c_1\epsilon N$$

where all quantities involved are computed as in Algorithm 5.

Proof. Let us consider a record r with value $v = \mathsf{val}(r)$. Recall that $\mathsf{est\text{-}symval}(r) = |\arg\min_{k\in[N/2]}(c(r) - p(k))|$. Note that the probability p that a value is hit by a uniform query is monotone increasing over $[1, N/2]$. It follows that for the event $|\arg\min_{k\in[N/2]}(c(r) - p(k))| \leq c_1\epsilon N$ to hold for r, it is enough that:

$$p(v - c_1\epsilon N) \leq c(r) \leq p(v + c_1\epsilon N).$$

Since $c(r)$ converges towards $p(v)$, the previous equation holds iff $c(r)$ does not deviate from its expected value $p(v)$ by more than:

$$\max\left(p(v) - p(v - c_1\epsilon N), p(v + c_1\epsilon N) - p(v)\right).$$

Elementary analysis shows that the previous quantity is minimized (thus maximizing the constraint on the precision) for $v = N/2$, yielding:

$$\begin{aligned} & p(N/2) - p(N/2 - c_1\epsilon N) \\ = & \frac{2}{N(N+1)} \cdot c_1\epsilon N \cdot (1 + c_1\epsilon N) \\ = & \Theta(\epsilon^2) \end{aligned}$$

where the second line uses Equation (2).

Thus we only need the empirical approximation $c(r)$ of $p(v)$ to be correct within an (additive) factor $\Theta(\epsilon^2)$. Using the concept space $(\mathsf{X}, \mathbb{C})$, a direct application of the ϵ-sample theorem (with $\Theta(\epsilon^2)$ in place of the "ϵ" of the theorem) yields the result. □

Before moving on to the next step, we refine the previous analysis to show that within the same query complexity, GetEstsymval is able to approximate the value of each record r to within $\epsilon \cdot \mathsf{symval}(r)$ (which is better than the ϵN of Lemma B.3). The price to pay is that we sacrifice some records with values within $\mathcal{O}(\epsilon N)$ of the endpoints 1 and N. This improvement will be useful later on.

Lemma B.4. *Let $\delta > 0$. Let $c_2 > 0$ and $c_3 > 0$ be two constants. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^4}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^4}\log\frac{1}{\delta}\right).$$

queries, the following event simultaneously *holds for all records r such that $\mathsf{symval}(r) \geq c_3\epsilon N$, except with probability at most δ:*

$$|\mathsf{est\text{-}symval}(r) - \mathsf{symval}(r)| < c_2\epsilon \cdot \mathsf{symval}(r)$$

where all quantities involved are computed as in Algorithm 5.

Proof. Fix a record r, and let $k = \mathsf{symval}(r) \in [N/2]$. We separate two cases based on k.

- Case 1: $k > N/4$. In that case $k = \Theta(N)$, so Lemma B.3 suffices to conclude.
- Case 2: $k \leq N/4$. In that case we follow exactly the same reasoning as in the proof of Lemma B.3, with $c_2\epsilon k$ playing the role of $c_1\epsilon N$, up until the point where we seek to minimize the quantity:

$$\max\left(p(k) - p(k - c_2\epsilon k), p(k + c_2\epsilon k) - p(k)\right).$$

This time around, because $k \leq N/4$, and because p is concave, the minimum is reached for $k = N/4$, which yields:

$$\begin{aligned} & p(N/4) - p(N/4 - c_2\epsilon k) \\ = & \frac{2}{N(N+1)} \cdot c_2\epsilon k \cdot (N/2 + 1 + c_2\epsilon k) \\ = & \Theta(\epsilon k/N) \\ = & \Omega(\epsilon^2) \end{aligned}$$

where the second line uses Equation (2), and the last line uses the assumption $k \geq c_3\epsilon N$. At this point we can conclude exactly like in Lemma B.3. □

Step 2. *Analysis of* GetEstval. In this second step, our goal is to show that GetEstval places all records on the correct side of $N/2$, relative to the chosen anchor record r_A, except for records close to 1, $N/2$ and $(N+1)/2$, which will be handled separately.

We will make use of the concept space $(\mathsf{X}, \mathbb{C}')$ where the ground set X is the set of all ranges, and each concept in $\mathbb{C}'$ is the set of queries hitting a given value k together with a fixed anchor value $v_A = \mathsf{val}(r_A)$ (i.e. each value k yields one concept $\{[x, y] \subseteq [N] : v_A, k \in [x, y]\}$). A straightforward adaptation of the proof of Lemma B.2 shows that $(\mathsf{X}, \mathbb{C}')$ has VC dimension 2.

First, we treat the case where $\epsilon = 1/N$, which requires special attention. Recall that sacrificial $1/N$-ADR is equivalent to full database reconstruction.

Lemma B.5. *Let $\delta > 0$. After*

$$\mathcal{O}\left(N^4 \log N + N^4 \log \frac{1}{\delta}\right).$$

queries, GeneralizedKKNO *achieves full database reconstruction with probability of success at least $1 - \delta$.*

Proof. Using Lemma B.3 with $\epsilon = 1/N$ and $c_1 = 1$, we can ensure $\mathsf{est\text{-}symval}(r) = \mathsf{symval}(r)$ for all r. It remains to show that for each record r, GetEstval correctly determines which side of $N/2$ the record r is relative to r_A, so that GetEstval returns the correct choice for $\mathsf{est\text{-}val}$.

Fix a record r. Let $a = \min(\mathsf{symval}(r), \mathsf{symval}(r_A))$ and $b = \max(\mathsf{symval}(r), \mathsf{symval}(r_A))$. Assume that r and r_A are on opposite sides of $(N+1)/2$ (the other case is very similar). Observe that $c'(r)$ converges towards $2ab/(N(N+1))$, and the algorithm succeeds if $c'(r)$ is less than a/N. Let us compute:

$$\begin{aligned} \frac{a}{N} - \frac{2ab}{N(N+1)} &= \frac{a}{N}\left(1 - \frac{2b}{N+1}\right) \\ &\geq \frac{1}{N}\left(1 - \frac{N}{N+1}\right) \\ &= \Omega\left(\frac{1}{N^2}\right). \end{aligned}$$

It follows that for the algorithm to succeed, it is enough that $c'(r)$ be within $\Omega(1/N^2)$ of its expected value. Using the concept space $(\mathsf{X}, \mathbb{C}')$ (where we recall that each concept corresponds to the set of ranges containing a given value k together with the anchor value $v_A = \mathsf{val}(r_A)$), a direct application of the ϵ-sample theorem with $\epsilon = \Omega(1/N^2)$ yields the result. □

We now remove the assumption $\epsilon N = 1$ and treat the general case. We distinguish two cases, based on whether there exists a "good" anchor record or not within the database. First, we will assume that every record in the database has values within $\epsilon N/4$ of 1, N and $(N+1)/2$. This is the case where no good anchor record exists. Note that the adversary does not know that fact, but that is immaterial: what we will show is that the same algorithm GeneralizedKKNO succeeds whether a good anchor record exists or not. The following lemma handles this first case.

Lemma B.6. *Let $\delta > 0$. Assume all records r satisfy $\mathsf{val}(r) \in [1, 1+\epsilon N/4] \cup [(N+1)/2 - \epsilon N/4, (N+1)/2 + \epsilon N/4] \cup [N - \epsilon N/4, N]$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^4}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^4}\log\frac{1}{\delta}\right).$$

queries, GeneralizedKKNO *achieves sacrificial ϵ-ADR with probability of success at least $1-\delta$.*

Proof. First, note that the definition of sacrificial ϵ-ADR makes no claim about records within $\epsilon N/4$ of 1 or N, so we can focus purely on those records whose values lie within $\epsilon N/4$ of $(N+1)/2$. Fix a record r such that $|(N+1)/2 - \mathsf{val}(r)| \leq \epsilon N/4$. Then $|(N+1)/2 - \mathsf{symval}(r)| \leq \epsilon N/4$. We apply Lemma B.3 with $c_1 = 1/4$. We get $|(N+1)/2 - \mathsf{est\text{-}symval}(r)| \leq \epsilon N/2$. Hence, whether GetEstval returns $\mathsf{est\text{-}symval}(r)$ or $N+1-\mathsf{est\text{-}symval}(r)$, we always have $|(N+1)/2 - \mathsf{est\text{-}val}(r)| \leq \epsilon N/2$. Since we also have $|(N+1)/2 - \mathsf{val}(r)| \leq \epsilon N/4$, we get $|\mathsf{val}(r) - \mathsf{est\text{-}val}(r)| < \epsilon N$. □

We are now free to assume that there exists a record r with $\mathsf{symval}(r) \in]1+\epsilon N/4, (N+1)/2 - \epsilon N/4[$. This is the case where a good anchor record exists, and it is handled by the following lemma.

Lemma B.7. *Let $\delta > 0$. Assume there exists a record r such that $\mathsf{symval}(r) \in]1+\epsilon N/4, (N+1)/2 - \epsilon N/4[$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^4}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^4}\log\frac{1}{\delta}\right).$$

queries, GeneralizedKKNO *achieves sacrificial ϵ-ADR with probability of success at least $1-\delta$.*

Proof. Let r be as in the assumption of the lemma. Then $|\mathsf{symval}(r) - N/4| < N/4 + 1/2 - \epsilon N/4$. Using Lemma B.3 with $c_1 = 1/16$, we get $|\mathsf{est\text{-}symval}(r) - N/4| < N/4 + 1/2 - (1/4 - 1/16)\epsilon N$. Since r_A is chosen such that $\mathsf{est\text{-}symval}(r_A)$ is closest to $N/4$, the same inequality applies to r_A. Finally, because we have applied Lemma B.3 with $c_1 = 1/16$, we can deduce that $|\mathsf{symval}(r_A) - N/4| < N/4 + 1/2 - \epsilon N/8$.

First, consider the case that $\epsilon N \leq 16$. In that case $\epsilon = N/16$, so $\mathcal{O}(\epsilon^{-4}\log\epsilon^{-1} + \epsilon^{-4}\log\delta^{-1})$ is the same as $\mathcal{O}(N^4\log N + N^4\log\delta^{-1})$, and we can apply Lemma B.5 to achieve full database reconstruction within that query complexity. In particular we achieve sacrificial ϵ-ADR, so we are done. Hence in the remainder we are free to assume $\epsilon N > 16$. In that case the inequality $|\mathsf{symval}(r_A) - N/4| < N/4 + 1/2 - \epsilon N/8$ from earlier implies

$$1 + \Omega(\epsilon N) \leq \mathsf{symval}(r_A) \leq \frac{N}{2} - \Omega(\epsilon N). \tag{3}$$

Fix a record r. We want to show that the algorithm succeeds for that record. Let us first consider the case that $|(N+1)/2 - \mathsf{val}(r)| \leq \epsilon N/4$. In that case we can apply Lemma B.3 with $c_1 = 1/4$, and

apply exactly the same reasoning as in the proof of Lemma B.6: whichever value GetEstval returns of $\mathsf{est\text{-}symval}(r)$ or $N+1-\mathsf{est\text{-}symval}(r)$, we always have $|(N+1)/2-\mathsf{est\text{-}val}(r)| \leq \epsilon N/2$. Since we also have $|(N+1)/2-\mathsf{val}(r)| \leq \epsilon N/4$, we get $|\mathsf{val}(r)-\mathsf{est\text{-}val}(r)| < \epsilon N$, so the algorithm succeeds for that record r. Hence we are free to assume $|(N+1)/2-\mathsf{val}(r)| > \epsilon N/4$. Further, since we are targeting sacrificial ϵ-ADR, we are also free to assume $\mathsf{symval}(r) \geq \epsilon N$. In combination (and using $\epsilon N > 16$), the previous two observations yield:

$$1+\Omega(\epsilon N) \leq \mathsf{symval}(r) \leq \frac{N}{2} - \Omega(\epsilon N). \tag{4}$$

By Equation (3), the same holds for the anchor r_A.

We can now tackle the main part of the proof. We want to show is that GetEstval correctly determines which side of $(N+1)/2$ the record r is relative to r_A, so that GetEstval returns the correct choice for $\mathsf{est\text{-}val}$. Let:

$$\begin{aligned} a &= \min(\mathsf{symval}(r), \mathsf{symval}(r_A)) \\ b &= \max(\mathsf{symval}(r), \mathsf{symval}(r_A)) \\ a' &= \min(\mathsf{est\text{-}symval}(r), \mathsf{est\text{-}symval}(r_A)) \\ b' &= \max(\mathsf{est\text{-}symval}(r), \mathsf{est\text{-}symval}(r_A)). \end{aligned}$$

An important remark is that because the same equations Equations (3) and (4) hold for both r and r_A, they also hold for a and b. Assume that r and r_A are on opposite sides of $(N+1)/2$ (the other case is very similar). Observe that $c'(r)$ converges towards $2ab/(N(N+1))$, and the algorithm succeeds if $c'(r)$ is less than a'/N. Because $a = \Omega(\epsilon N)$, we can use Lemma B.4 to ensure $\mathsf{est\text{-}symval}(r) \geq \mathsf{symval}(r)(1-c_2\epsilon)$, for a constant $c_2 > 0$ of our choice; and likewise for r_A. It follows that $a' \geq a(1-c_2\epsilon)$. Let us compute:

$$\begin{aligned} \frac{a'}{N} - \frac{2ab}{N(N+1)} &\geq \frac{a}{N}\left(1 - c_2\epsilon - \frac{2b}{N+1}\right) \\ &\geq \frac{a}{N}\left(1 - c_2\epsilon - 1 + \Omega(\epsilon)\right) \\ &\geq \Omega(\epsilon)\left(\Omega(\epsilon) - c_2\epsilon\right). \end{aligned}$$

Since we are free to pick $c_2 > 0$ arbitrarily, we choose it to be small enough that the above quantity is $\Omega(\epsilon^2)$.

As a result, for the algorithm to succeed, it is enough that $c'(r)$ be within $\Omega(\epsilon^2)$ of its expected value. Using the concept space $(\mathsf{X}, \mathbb{C}')$ (where we recall that each concept corresponds to the set of ranges containing a given value k together with the anchor value $v_A = \mathsf{val}(r_A)$), a direct application of the ϵ-sample theorem (with $\Omega(\epsilon^2)$ playing the role of the "ϵ" in the theorem statement) yields the result. □

It is clear that Lemmas B.6 and B.7 together imply Theorem B.1, so we are done.

C Query Complexity of ApproxValue

C.1 Algorithm

The general setup and notation are the same as for the analysis of GeneralizedKKNO. Throughout, we assume ϵN is a strictly positive integer[2]. For simplicity (to avoid the proliferation of rounding) we

[2]This is without loss of generality, because the relevant quantity in the definition of sacrificial ϵ-ADR is $\lceil \epsilon N \rceil$. That is, ϵN can be replaced by $\lceil \epsilon N \rceil$ everywhere in the definition, and this does not affect whether it is satisfied.

also assume $N/4$ is an integer. Recall that the probability that a given value $k \in [1, N]$ is hit by a uniform query is:

$$p(k) \stackrel{\text{def}}{=} \frac{2}{N(N+1)} k(N+1-k).$$

We have:

$$p(k+\delta) - p(k) = \frac{2}{N(N+1)} \delta(N+1-2k-\delta). \tag{5}$$

Similarly, for a given *anchor value* $v_A \in [1, N]$, the probability that a given value k is hit together with the anchor value by a uniform query is:

$$d(v_A, k) \stackrel{\text{def}}{=} \frac{2}{N(N+1)} \cdot \begin{cases} k(N+1-v_A) & \text{if } k \leq v_A \\ v_A(N+1-k) & \text{if } k > v_A. \end{cases}$$

Intuitively, $d(v_A, k)$ measures a type of distance between the anchor value v_A and the target value k. Indeed the closer the two values are, the more likely they are to be hit by the same query. Note that $d(k, k) = p(k)$.

During the analysis, we will make use of the concept space $(\mathsf{X}, \mathbb{C})$ (resp. $(\mathsf{X}, \mathbb{C}')$), where the ground set X is the set of all ranges in $[N]$, and the concept set $\mathbb{C}$ (resp. $\mathbb{C}'$) is the set of all ranges containing a given value k (resp. containing a given value k together with a fixed anchor value $v_A = \mathsf{val}(r_A)$). That is, each value k yields the concept $\{[x, y] \subseteq [N] : k \in [x, y]\} \in \mathbb{C}$ (resp. $\{[x, y] \subseteq [N] : v_A, k \in [x, y]\} \in \mathbb{C}'$). In the analysis of GeneralizedKKNO, we have seen that both concept spaces have VC dimension 2.

For convenience, we recall the ApproxValue algorithm in Algorithm 6. Note that arg min is computed over the integers. On the other hand, throughout the analysis, we extend the functions p and d to be defined over rationals rather than just integers, using the same polynomial expressions.

Algorithm 6 ADR Algorithm ApproxValue.

ApproxValue($\mathcal{Q}$):
Input: Set of queries $\mathcal{Q}$.
Output: Function $\mathsf{est\text{-}val}$ approximating val.

for each record r **do**
  $c(r) \leftarrow |\{q \in \mathcal{Q} : r \in q\}|/|\mathcal{Q}|$
  $\widetilde{v}(r) \leftarrow \arg\min_k |c(r) - p(k)|$
end for
$r_A \leftarrow \arg\min_r |\widetilde{v}(r) - N/4|$ ▷ Anchor record
$\widetilde{v_A} \leftarrow \widetilde{v}(r_A)$ ▷ Est. anchor value
for each record r **do**
  $c'(r) \leftarrow |\{q \in \mathcal{Q} : r_A, r \in q\}|/|\mathcal{Q}|$
  $\widetilde{w_L} \leftarrow \arg\min_{k \in [1, \widetilde{v_A}]} |d(\widetilde{v_A}, k) - c'(r)|$
  $\widetilde{w_R} \leftarrow \arg\min_{k \in [\widetilde{v_A}, N]} |d(\widetilde{v_A}, k) - c'(r)|$
  if $c(r) < (p(\widetilde{w_L}) + p(\widetilde{w_R}))/2$ **then**
    $\mathsf{est\text{-}val}(r) \leftarrow \widetilde{w_L}$
  else
    $\mathsf{est\text{-}val}(r) \leftarrow \widetilde{w_R}$
  end if
end for

Finally, we recall that *sacrificial ϵ-approximate database reconstruction* (sacrificial ϵ-ADR) is said to succeed iff one of the following two events occur:

1. For every record r such that $\mathsf{val}(r) \in [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$, $|\mathsf{est\text{-}val}(r) - \mathsf{val}(r)| < \epsilon N$.
2. For every record r such that $\mathsf{val}(r) \in [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$, $|\mathsf{est\text{-}val}(r) - (N + 1 - \mathsf{val}(r))| < \epsilon N$.

We now state our main result.

Theorem C.1. *Let $\epsilon < 1/4$. Assume there exists a record r_0 such that $0.2N < \mathsf{symval}(r_0) < 0.3N$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, ApproxValue *achieves sacrificial ϵ-ADR with probability of success at least $1 - \delta$.*

The proof of the theorem is given in the next section.

C.2 Analysis

Our goal in this section is to prove Theorem C.1. Throughout this section, we assume the condition of Theorem C.1 is satisfied, i.e. there exists a record r_0 such that $0.2N < \mathsf{symval}(r_0) < 0.3N$.

At a high level, Algorithm 6 proceeds by first choosing an anchor record r_A, and approximating its value using the empirical probability $c(r_A)$ that it is hit by a query. This yields an approximation $\widetilde{v_A}$ of the value of the anchor record. The anchor record is then used to approximate the value of all other records using the empirical probability $c'(r)$ that a query hits both the anchor record r_A and the target record r whose value we are approximating. This yields two candidate approximate values $\widetilde{w_L}$ and $\widetilde{w_R}$ for $\mathsf{val}(r)$, depending on whether $\mathsf{val}(r)$ is to the left or right of the anchor—we say that the two values are "reflections" around the anchor, by analogy with what happens in the GeneralizedKKNO Algorithm. Finally the empirical probability $c(r)$ that record r is hit by a query is once again used to decide which of the values $\widetilde{w_L}$ or $\widetilde{w_R}$ is the correct one, by checking which of the two choices predicts $c(r)$ more accurately.

The success of Algorithm 6 depends on successive approximations holding with sufficient precision, so that the final output achieves a precision of ϵN (within the claimed number of queries). The formal proof below proceeds in three main steps. In a nutshell, the first step is to show that the approximation $\widetilde{v_A}$ of the value of the anchor record r_A is sufficiently precise. The second step is to deduce that for every record r, one of the two values $\widetilde{w_L}$ and $\widetilde{w_R}$ is an approximation of the target record value $\mathsf{val}(r)$ with sufficient precision. The third step is to show that the condition on Line 11 of the algorithm correctly determines which of the two previous values is the correct one (except possibly for records whose value is within ϵN of either 1 or N). Finally we will put all three steps together to conclude the proof.

We now move on to the actual proof. Throughout, $\epsilon < 1/4$ denotes the target precision, and $\mathcal{Q}$ the set of queries. We assume that the anchor record has a value less than $N/2$. If not, it means we will ultimately recover the reflection of the record values, instead of the record values themselves, but it does not affect the success of the algorithm. Indeed it simply means we will ultimately satisfy condition (2) in the definition of sacrificial ϵ-ADR instead of condition (1). (This assumption is where the two possible success conditions are separated.)

Step 1. *Approximation of the anchor record value (lines 1-6 of Algorithm 6).* Let r_A denote the anchor record as computed in Line 5 of Algorithm 6, and let $v_A \stackrel{\text{def}}{=} \mathsf{val}(r_A)$ denote its value. In Algorithm 6 this value is approximated by $\widetilde{v_A}$. Our first goal is to show that $\widetilde{v_A}$ is a "good enough" approximation of v_A. This will be the result of two sub-steps. In Step 1a, we show that v_A is sufficiently close to $N/4$. In Step 1b, we deduce that $\widetilde{v_A}$ is sufficiently close to v_A.

Recall that we assume $v_A \leq N/2$. As a result, we focus our analysis on values in $[1, N/2]$.

Step 1a. In this sub-step, we first require that for all records r (with values in $[1, N/2]$), when approximating $\mathsf{val}(r)$ by $\widetilde{v}(r)$, the approximation is wrong by no more than $N/20$. More precisely we prove the following lemma.

Lemma C.2. *Let $\delta > 0$. Then $\mathcal{O}(\log \delta^{-1})$ queries suffice for the following event:*

$$\left|\widetilde{v}(r) - \mathsf{val}(r)\right| \leq N/20$$

to simultaneously hold for every record r such that $\mathsf{val}(r) \leq N/2$, except with probability at most δ.

Proof. This proof is essentially the same as in the first step of the GeneralizedKKNO algorithm, where we have set $\epsilon = 1/20$. For the sake of completeness we recall it here. Let us consider a record r with value $v = \mathsf{val}(r)$. Recall that $\widetilde{v}(r) = |\arg\min_k(c(r) - p(k))|$. Note that the probability p that a value is hit by a uniform query is monotone increasing over $[1, N/2]$. It follows that for the event $|\arg\min_k(c(r) - p(k))| \leq N/20$ to hold for r, it is enough that:

$$p(v - N/20) \leq c(r) \leq p(v + N/20).$$

Since $c(r)$ converges towards $p(v)$, the previous equation holds iff $c(r)$ does not deviate from its expected value $p(v)$ by more than:

$$\max\left(p(v) - p(v - N/20), p(v + N/20) - p(v)\right).$$

Elementary analysis shows that the previous quantity is minimized (thus maximizing the constraint on the precision) for $v = N/2$, yielding:

$$\begin{aligned} & p(N/2) - p(N/2 - N/20) \\ = & \frac{2}{N(N+1)} \cdot \frac{N}{20} \cdot (1 + N/20) \\ = & \Theta(1) \end{aligned}$$

where the second line uses Equation (5).

Thus we only need the empirical approximation $c(r)$ of $p(v)$ to be correct within a constant (additive) factor. Using the concept space $(\mathsf{X}, \mathbb{C})$ (where we recall that each concept corresponds to the set of ranges containing a given value k), a direct application of the ϵ-sample theorem with constant ϵ yields the result. □

Since by hypothesis there exists a record r with (symmetric) value within $[0.2N, 0.3N]$, it follows from Lemma C.2 that after $\mathcal{O}(\log \delta^{-1})$ queries, for that particular record r, we have $\widetilde{v}(r) \in [0.15N, 0.35N]$. Since by construction the anchor record r_A minimizes $|\widetilde{v}(r_A) - N/4|$, it follows that $\widetilde{v}(r_A) \in [0.15N, 0.35N]$. By Lemma C.2 again, after $\mathcal{O}(\log \delta^{-1})$ queries, we can deduce $v_A \in [0.1N, 0.4N]$. That makes the anchor record "good enough" for our purpose, and concludes Step 1a of the proof.

Step 1b. Our aim in this step is to show that $\widetilde{v_A}$ is a good enough approximation of v_A. More precisely, we want to prove the following.

Lemma C.3. *Let $\delta > 0$. Fix c_1 to be any constant with $0 < c_1 < 1/2$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event holds except with probability at most δ:

$$|\widetilde{v_A} - v_A| \leq c_1 \epsilon N.$$

Proof. Recall we are assuming the value v_A of the anchor record is less than $N/2$. In fact, using Lemma C.2, by paying $\mathcal{O}(\log \delta^{-1})$ queries we can assume $v_A \in [0.1N, 0.4N]$.

Note that the probability p that a value is hit by a uniform query is monotone increasing over $[1, N/2]$. It follows that to ensure $|\widetilde{v_A} - v_A| \leq c_1 \epsilon N$, it is enough to have:

$$p(v_A - c_1\epsilon N) \leq c(r_A) \leq p(v_A + c_1\epsilon N).$$

As in the proof of Lemma C.2, since $c(r_A)$ converges towards $p(v_A)$, the previous equation holds iff $c(r_A)$ does not deviate from its expected value $p(v_A)$ by more than:

$$\max\left(p(v_A) - p(v_A - c_1\epsilon N), p(v_A + c_1\epsilon N) - p(v_A)\right).$$

Because p is concave and increasing over $[1, N/2]$ and all previous values lie within that range, the maximum is reached for $v_A = 0.4N$ with:

$$\begin{aligned} & p(0.4N + c_1\epsilon N) - p(0.4N) \\ = & \frac{2}{N(N+1)} c_1\epsilon N(N + 1 - 0.8N - c_1\epsilon N) \\ = & \Omega(\epsilon). \end{aligned}$$

where the second line uses Equation (5), and the last line uses $c_1\epsilon < 1/2 \cdot 1/4 < 0.2$ (and that c_1 is constant).

Thus it is enough that the empirical probability $c(r_A)$ be within $\Omega(\epsilon)$ of its expected value $p(v_A)$. Even though we need the approximation to hold only for a single value, that value was not fixed and known in advance. On the contrary, the anchor record was chosen based on the empirical probability it is hit by a range query, which biases its distribution. As a result a simple bound such as a Hoeffding bound does not directly apply. We circumvent that problem by requiring that the approximation hold for all values. To that end, we once again use the concept space $(\mathsf{X}, \mathbb{C})$. A direct application of the ϵ-sample theorem yields the result. □

Step 2. *Approximation of the target record value, up to reflection around the anchor (lines 7-10 of Algorithm 6).* Informally, our goal in this step is to prove that for every record r with value $v \stackrel{\text{def}}{=} \mathsf{val}(r)$, if v is to the *left* (resp. *right*) of the anchor value, i.e. $v \leq v_A$ (resp. $v \geq v_A$), then w_L (resp. w_R) is a good enough approximation of v. In the remainder we focus on the case $v \geq v_A$; the other case is very similar. Hence our goal is to show that $\widetilde{w_R}$ is a good enough approximation of v.

To that end, recall that $v_A = \mathsf{val}(r_A)$, and let $w_R \stackrel{\text{def}}{=} \arg\min_{k \in [v_A, N]} |d(v_A, k) - c'(r)|$. That is, w_R is computed in exactly the same way as $\widetilde{w_R}$, except using the actual value v_A of the anchor record rather than its approximation $\widetilde{v_A}$. We proceed in two lemmas. The first lemma shows that v is close to w_R. The second lemma shows that w_R is close to $\widetilde{w_R}$. As in the previous step, "close enough" is defined as being within $c_2\epsilon$ of each other, for some constant c_2.

Lemma C.4. *Let $\delta > 0$. Fix c_2 to be any constant with $0 < c_2 \leq 1$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event holds simultaneously *for all records r such that $\mathsf{val}(r) \geq v_A$, except with probability at most δ:*

$$|\mathsf{val}(r) - w_R| \leq c_2\epsilon N$$

where $w_R = \arg\min_{k \in [v_A, N]} |d(v_A, k) - c'(r)|$, and $v_A = \mathsf{val}(r_A)$.

Proof. Fix a target record r with $v = \mathsf{val}(r) \geq v_A$. The main point is that $c'(r)$ converges towards $d(v_A, k)$. Using the same reasoning as in the proof of Lemma C.2, a sufficient condition for our goal $|v - w_R| \leq c_2\epsilon N$ to hold is that:

$$d(v_A, v - c_2\epsilon N) \leq c'(r) \leq d(v_A, v + c_2\epsilon N) \tag{6}$$

as long as $v_A + c_2\epsilon N \leq v \leq N - c_2\epsilon N$. In the case where $v_A \leq v < v_A + c_2\epsilon N$ (resp. $N - c_2\epsilon N < v \leq N$), the first (resp. second) inequality in Equation (6) is unnecessary. For simplicity we focus on the case $v_A + c_2\epsilon N \leq v \leq N - c_2\epsilon N$ where both inequalities are necessary; the analysis easily extends to the other two cases.

By Step 1, after $\mathcal{O}(\log \delta^{-1})$ queries, we are free to assume $v_A \in [0.1N, 0.4N]$. On the other hand, by definition of d, and because $v_A + c_2\epsilon N \leq v \leq N - c_2\epsilon N$, we have:

$$\begin{aligned} d(v_A, v - c_2\epsilon N) - d(v_A, v) &= d(v_A, v) - d(v_A, v + c_2\epsilon N) \\ &= \frac{2}{N(N+1)} v_A c_2 \epsilon N \\ &= \Omega(\epsilon) \end{aligned}$$

where the last line uses $v_A \geq 0.1N = \Omega(N)$.

Using the concept space $(\mathsf{X}, \mathbb{C}')$ (where we recall that each concept corresponds to the set of ranges containing a given value k together with the anchor value v_A), a direct application of the ϵ-sample theorem yields the result. □

Lemma C.5. *Let $\delta > 0$. Fix c_3 to be any constant with $0 < c_3 \leq 1$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event holds simultaneously *for all records r such that $\mathsf{val}(r) \geq v_A$, except with probability at most δ:*

$$|\widetilde{w_R} - w_R| \leq c_3\epsilon N$$

where $w_R = \arg\min_{k \in [v_A, N]} |d(v_A, k) - c'(r)|$, $v_A = \mathsf{val}(r_A)$, and $\widetilde{v_A}$, $\widetilde{w_R}$ are computed as in Algorithm 6.

Proof. Since the mapping $x \mapsto d(v_A, x)$ is affine, the output of $\arg\min_{x \in [v_A, N]} |d(v_A, x) - c'(r)|$ can be easily computed over the rationals. Indeed if we let $\gamma \stackrel{\text{def}}{=} c'(r)$, then it suffices to solves the equation:

$$\frac{2}{N(N+1)} v_A (N + 1 - x) = \gamma$$

which yields:

$$x = N + 1 - \frac{N(N+1)\gamma}{2v_A}.$$

For ease of exposition, we will first ignore integer rounding issues, and pretend that the above quantity is exactly the output of $\arg\min_{x \in [v_A, N]} |d(v_A, x) - c'(r)|$, so that

$$w_R = N + 1 - \frac{N(N+1)\gamma}{2v_A}.$$

Likewise for $\widetilde{w_R}$. We will examine the impact of integer rounding on the reasoning later on. We get:

$$\begin{aligned}|\widetilde{w_R} - w_R| &= \frac{N(N+1)\gamma}{2}\Big|\frac{1}{\widetilde{v_A}} - \frac{1}{v_A}\Big| \\ &\leq \frac{N(N+1)}{2} \cdot \frac{|\widetilde{v_A} - v_A|}{\widetilde{v_A} v_A}\end{aligned}$$

where the second line uses $\gamma \leq 1$.

Applying the results in Step 1, after $\mathcal{O}\left(\epsilon^{-2}\log\epsilon^{-1} + \epsilon^{-2}\log\delta^{-1}\right)$ queries, with arbitrarily high constant probability we have $|\widetilde{v_A} - v_A| \leq c_1\epsilon N$ (with the assumption $c_1 \leq 1/2$) and $\widetilde{v_A}, v_A \in [0.1N, 0.4N]$. It follows that the previous quantity is $\mathcal{O}(c_1\epsilon)$. Hence it is less than $C \cdot c_1\epsilon$ for some constant C, so choosing $c_1 = \min(1/2, c_3/C)$ yields the desired result.

It remains to discuss rounding issues. In reality the arg min in the definition of w_R and $\widetilde{w_R}$ is computed over integers. Adapting the previous reasoning to take this into account ultimately yields a bound $|\widetilde{w_R} - w_R| \leq c_3\epsilon N + 1$ instead of $|\widetilde{w_R} - w_R| \leq c_3\epsilon N$ as desired. That issue can be avoided as follows. We consider two cases, depending on the value of $c_3\epsilon N$. Essentially the dichotomy is that in the first case, $c_3\epsilon N$ is large enough that rounding errors don't matter; and in the second case it is small enough that we can afford to trivialize the lemma and get rid of rounding errors entirely.

- Case 1: $c_3\epsilon N \geq 2$. Then we can apply the previous reasoning, replacing c_3 with the value $c_3' \stackrel{\text{def}}{=} c_3/2$. We obtain a bound $c_3'\epsilon N + 1 \leq c_3\epsilon N$, so we are done.
- Case 2: $c_3\epsilon N < 2$. In that case we can apply Lemma C.3 with $\epsilon = 1/N$ and $c_1 < 1$. This costs $\mathcal{O}(N^2 \log N)$ queries, which is also $\mathcal{O}(\epsilon^{-2}\log\epsilon^{-1})$ since $c_3\epsilon N < 2$ and c_3 is a constant. This yields $|\widetilde{v_A} - v_A| < 1$, which in turn implies $\widetilde{v_A} = v_A$ since they are both integers. It follows immediately that $\widetilde{w_R} = w_R$ and we are done.

This concludes the proof of Lemma C.5. □

Putting the previous two proofs together we get the following lemma.

Lemma C.6. *Let $\delta > 0$. Fix c_4 to be any constant with $0 < c_4 \leq 1$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event holds simultaneously *for all records r, except with probability at most δ:*

$$\begin{aligned}|\mathsf{val}(r) - \widetilde{w_L}| &\leq c_4\epsilon N \quad \text{if } \mathsf{val}(r) \leq v_A \\ |\mathsf{val}(r) - \widetilde{w_R}| &\leq c_4\epsilon N \quad \text{if } \mathsf{val}(r) \geq v_A\end{aligned}$$

where w_L, $\widetilde{w_L}$, w_R, $\widetilde{w_R}$, v_A, $\widetilde{v_A}$, are defined as previously.

Proof. The second inequality comes directly from combining Lemmas C.4 and C.5. Since we are aiming to show a bound on the number of queries sufficient to succeed except with probability at most δ, and the previous two lemmas held in the same setting, the bounds can be added in a straightforward manner, by requiring a probability of failure at most $\delta/2$ in each lemma and using a union bound. The first inequality can be proven in a similar way, and combined with the first as previously. □

Step 3. *Correctly determining reflection around the anchor (lines 11-15 of Algorithm 6).* At the outcome of Step 2, we have essentially shown that for every record, one of $\widetilde{w_L}$ or $\widetilde{w_R}$ is a good approximation

of the record value (whp after the required number of queries). Now it remains to show that the test on line 11 of Algorithm 6 is able to correctly determine which of the two values is the correct approximation (except for values within ϵN of 1 or N; or for values near the anchor, for which both approximations are close anyway). We proceed in two steps, with Lemmas C.9 and C.10 being the main lemmas. The first lemma shows that if $\widetilde{w_L}$ (resp. $\widetilde{w_R}$) is the correct approximation, then $c(r)$ is sufficiently close to $p(\widetilde{w_L})$ (resp. $p(\widetilde{w_R})$); the second lemma shows that $p(\widetilde{w_R})$ and $p(\widetilde{w_R})$ are far enough apart for the relevant value ranges. We will then put both lemmas together to reach the desired conclusion.

Lemma C.7. *Let $\delta > 0$. Fix c_5 to be any constant with $0 < c_5 \leq 1$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event simultaneously *holds for all records r, except with probability at most δ:*

$$|c(r) - p(\mathsf{val}(r))| \leq c_5\epsilon$$

where all quantities involved are computed as in Algorithm 6.

Proof. The is a direct application of the ϵ-sample theorem to the concept space $(\mathsf{X}, \mathbb{C})$. □

Lemma C.8. *Let $\delta > 0$. Fix c_6 to be any constant with $0 < c_6 \leq 1$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event holds simultaneously *for all records r such that $\mathsf{val}(r) \geq v_A$, except with probability at most δ:*

$$\left|p(\widetilde{w_R}) - p(\mathsf{val}(r))\right| \leq c_6\epsilon$$

where all quantities involved are computed as in Algorithm 6.

Proof. Applying Lemma C.6 with $c_4 = c_6$, the claimed number of queries suffices to ensure $\left|\widetilde{w_R} - \mathsf{val}(r)\right| \leq c_6\epsilon N$. Using Equation (5), we get:

$$\begin{aligned}
&\left|p(\widetilde{w_R}) - p(\mathsf{val}(r))\right| \\
&\leq \frac{1}{N(N+1)} c_6\epsilon N|N + 1 - \mathsf{val}(r) - \widetilde{w_R}| \\
&\leq c_6\epsilon
\end{aligned}$$

which is what we wanted. □

Combining Lemmas C.7 and C.8 yields the following corollary.

Lemma C.9. *Let $\delta > 0$. Fix c_7 to be any constant with $0 < c_7 \leq 1$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event holds simultaneously *for all records r, except with probability at most δ:*

$$\begin{aligned}
|c(r) - p(\widetilde{w_L})| &\leq c_7\epsilon \quad \textit{if } \mathsf{val}(r) \leq v_A \\
|c(r) - p(\widetilde{w_R})| &\leq c_7\epsilon \quad \textit{if } \mathsf{val}(r) \geq v_A
\end{aligned}$$

where all quantities involved are computed as in Algorithm 6.

Proof. The proof follows in a straightforward manner from combining Lemmas C.7 and C.8, using the same approach as in the proof of Lemma C.6. □

Now we want to show that $p(\widetilde{w_R})$ is far apart from $p(\widetilde{w_L})$ (except for specific choices of $\mathsf{val}(r)$ that will be handled later).

Lemma C.10. *Let $\delta > 0$. Fix c_8 to be any constant with $0 < c_8 \leq 1$. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event holds simultaneously *for all records r, except with probability at most δ:*

$$\begin{aligned} &\textit{if } \mathsf{val}(r) \notin [1, \epsilon N[\cup]N+1-\epsilon N, N] \\ &\qquad\qquad \cup [\widetilde{v_A} - \epsilon N/4, \widetilde{v_A} + \epsilon N/4] \\ &\textit{then } |p(\widetilde{w_R}) - p(\widetilde{w_L})| \geq c_8\epsilon \end{aligned}$$

where all quantities involved are computed as in Algorithm 6.

Proof. For ease of exposition, we ignore rounding issues and regard arg min as operating over rationals rather than integers. Rounding issues can then be handled as in the proof of Lemma C.5. Likewise, we ignore the case where $c'(r)$ does not fall into the range of $x \mapsto d(\widetilde{v_A}, x)$ over the domains $[1, \widetilde{v_A}]$ and $[\widetilde{v_A}, N]$: this is without loss of generality, as those cases imply that $\mathsf{val}(r)$ is either 1, $\widetilde{v_A}$ or N (whp after $\mathcal{O}(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon})$ queries, by Lemma C.6), and all those cases are excluded in the statement of the current lemma.

As shown in Step 1, we can assume $\widetilde{v_A} \in [0.15N, 0.35N]$, so in particular $\widetilde{v_A} \leq N/2$, from which it follows that $N+1-\widetilde{w_R} \geq \widetilde{w_L}$, which implies that $p(\widetilde{w_R}) \geq p(\widetilde{w_L})$. Hence, in order to lower-bound $|p(\widetilde{w_R}) - p(\widetilde{w_L})|$, it will be enough to lower-bound $p(\widetilde{w_R}) - p(\widetilde{w_L})$.

In the remainder we assume $\mathsf{val}(r) \geq \widetilde{v_A}$. The case $\mathsf{val}(r) \leq \widetilde{v_A}$ is very similar.

If we let $\widetilde{w_L}$ (resp. $\widetilde{w_R}$) be defined as in line 9 (resp. line 10) of Algorithm 6, except computed over the rationals rather than the integers, then we get:

$$\begin{aligned} & & d(\widetilde{v_A}, \widetilde{w_L}) &= d(\widetilde{v_A}, \widetilde{w_R}) \\ \implies & & \widetilde{w_L}(N+1-\widetilde{v_A}) &= \widetilde{v_A}(N+1-\widetilde{w_R}) \\ \implies & & \widetilde{w_L} &= \frac{\widetilde{v_A}}{N+1-\widetilde{v_A}}(N+1-\widetilde{w_R}) \end{aligned}$$

Let $\gamma \stackrel{\text{def}}{=} \widetilde{v_A}/(N+1-\widetilde{v_A})$. Then by Equation (5):

$$\begin{aligned} & p(\widetilde{w_R}) - p(\widetilde{w_L}) \\ &= \frac{2}{N(N+1)}(\widetilde{w_R} - \widetilde{w_L})(N+1-\widetilde{w_L} - \widetilde{w_R}) \\ &= \frac{2}{N(N+1)}(1-\gamma)(N+1-\widetilde{w_R})(\widetilde{w_R} - \widetilde{w_L}). \end{aligned}$$

We now want to prove that the above expression is $\Omega(\epsilon)$. We begin with the term $1-\gamma$:

$$1 - \gamma = \frac{N+1-2\widetilde{v_A}}{N+1-\widetilde{v_A}}.$$

Using the results of Step 1 as earlier, we can assume $\widetilde{v_A} \in [0.15N, 0.35N]$. We get $1 - \gamma = \Theta(1)$.

We now turn to the rest of the expression, namely:

$$\frac{2}{N(N+1)}(N + 1 - \widetilde{w_R})(\widetilde{w_R} - \widetilde{w_L}).$$

Our goal is to show that that expression is $\Omega(\epsilon)$. We distinguish two cases.

- Case 1: $\widetilde{w_R} \leq 3N/4$. In that case it is clear that $N + 1 - \widetilde{w_R} = \Omega(N)$, so it suffices to show $\widetilde{w_R} - \widetilde{w_L} = \Omega(\epsilon N)$. Recall we are assuming $\mathsf{val}(r) \geq \widetilde{v_A}$, and since the statement of the lemma only considers records for which $|\mathsf{val}(r) - \widetilde{v_A}| > \epsilon N/4$, it follows that $\mathsf{val}(r) \geq \widetilde{v_A} + \epsilon N/4$. On the other hand using Lemma C.3, we are free to assume $|\widetilde{v_A} - v_A| \leq \epsilon N/4$, which implies $\mathsf{val}(r) \geq v_A$. This allows us to apply Lemma C.6 to get $|\mathsf{val}(r) - \widetilde{w_R}| \leq \epsilon N/8$. In turn, this implies $\widetilde{w_R} \geq \widetilde{v_A} + \epsilon N/8$. Since $\widetilde{w_L} \leq \widetilde{v_A}$ by construction, we get $\widetilde{w_R} - \widetilde{w_L} = \Omega(\epsilon N)$ as desired.
- Case 2: $\widetilde{w_R} > 3N/4$. In that case since $\widetilde{w_L} \leq \widetilde{v_A}$, and as previously we can use the results of Step 1 to get $\widetilde{v_A} \leq 0.4$, it is clear that $\widetilde{w_R} - \widetilde{w_L} = \Omega(N)$. Hence it suffices to show that $N + 1 - \widetilde{w_R} = \Omega(\epsilon N)$. Like in the previous case we can enforce $|\mathsf{val}(r) - \widetilde{w_R}| \leq \epsilon N/8$, and since the statement of the lemma only considers records such that $\mathsf{val}(r) \leq N + 1 - \epsilon N$, it follows that $N + 1 - \widetilde{w_R} = \Omega(\epsilon N)$ as desired.

From the previous reasoning, we conclude that $p(\widetilde{w_R}) - p(\widetilde{w_L}) = \Omega(\epsilon)$. Since c_8 is a constant the result follows. □

Putting all the lemmas from Step 3 so far together, we get the following result.

Lemma C.11. *Let $\delta > 0$. After*

$$\mathcal{O}\left(\frac{1}{\epsilon^2}\log\frac{1}{\epsilon} + \frac{1}{\epsilon^2}\log\frac{1}{\delta}\right).$$

queries, the following event holds simultaneously *for all records r, except with probability at most δ:*

$$\begin{aligned} &\textit{if } \mathsf{val}(r) \notin [1, \epsilon N[\cup]N + 1 - \epsilon N, N] \\ &\qquad \cup [\widetilde{v_A} - \epsilon N/4, \widetilde{v_A} + \epsilon N/4] \\ &\textit{then Algorithm 6 returns } \widetilde{w_L} \textit{ iff } \mathsf{val}(r) \leq \widetilde{v_A}. \end{aligned}$$

Proof. The proof follows directly from combining Lemmas C.9 and C.10, where we choose any constants c_7 and c_8 such that $c_8 > 2c_7$. □

Putting everything together. We now combine the previous three steps to prove Theorem C.1. We have already explained at the start of the section how to handle the reflection symmetry, by assuming $v_A \leq N/2$. To finish the proof, we need to show that condition (1) in the definition of sacrificial ϵ-approximate reconstruction holds (whp after the claimed number of queries). To that end let us consider a record r with $\mathsf{val}(r) \in [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$. We need to ensure $|\mathsf{est\text{-}val}(r) - \mathsf{val}(r)| < \epsilon N$. We distinguish two cases.

- Case 1: $|\mathsf{val}(r) - \widetilde{v_A}| \leq \epsilon N/4$. By Lemma C.6, we can ensure that one of $x = \widetilde{w_L}$ or $x = \widetilde{w_R}$ satisfies $|\mathsf{val}(r) - x| \leq \epsilon N/4$. We get $|x - \widetilde{v_A}| \leq \epsilon N/2$.

On the other hand, using the same reasoning as in the proof of Lemma C.10, we can show:

$$\widetilde{w_L} - \widetilde{v_A} = -\gamma(\widetilde{w_R} - \widetilde{v_A})$$

where $\gamma = \widetilde{v_A}/(N+1-\widetilde{v_A}) = \Theta(1)$. It follows that $|x - \widetilde{v_A}| = \mathcal{O}(\epsilon N)$ actually implies both $|\widetilde{w_L} - \widetilde{v_A}| = \mathcal{O}(\epsilon N)$ and $|\widetilde{w_R} - \widetilde{v_A}| = \mathcal{O}(\epsilon N)$. Hence $|\widetilde{w_R} - \widetilde{w_L}| = \mathcal{O}(\epsilon N)$. Since $|\mathsf{val}(r) - x| = \mathcal{O}(\epsilon N)$, we deduce that both $\widetilde{w_L}$ and $\widetilde{w_R}$ are within $\mathcal{O}(\epsilon N)$ of $\mathsf{val}(r)$. Hence whichever of the two values Theorem C.1 returns is correct within $\mathcal{O}(\epsilon N)$. In particular by setting small enough constants c_i's it is correct within ϵN.

- Case 2: $|\mathsf{val}(r) - \widetilde{v_A}| > \epsilon N/4$. We assume $\mathsf{val}(r) \geq \widetilde{v_A}$; the other case is very similar. By Lemma C.3 we can enforce $|\widetilde{v_A} - v_A| \leq \epsilon N/4$, so we can deduce $\mathsf{val}(r) \geq v_A$. By Lemma C.6, choosing small enough c_4, we get $|\mathsf{val}(r) - \widetilde{w_R}| < \epsilon N$. On the other hand by Lemma C.11, we have that Algorithm 6 returns $\widetilde{w_R}$, so we are done.

This concludes the proof of Theorem C.1.

D Query Complexity of ApproxOrder

Preliminaries

General notation.

- If S is a set and f is a mapping whose domain includes S, then $f(S) \stackrel{\text{def}}{=} \{f(s) : s \in S\}$.
- For a set or multiset S, $\#S$ denotes the cardinality of S. If S is a multiset, cardinality includes multiplicity. In particular, whenever we write something of the form $\#\{[x,y] \in \mathcal{Q}_R : P(x,y)\}$ for some property P, multiplicity should be taken into account.

Records and values.

- $\mathcal{Q}_R$ denotes the sequence of queried ranges. $\mathcal{Q}$ is the sequence of access pattern leakage for all queries, i.e. $\mathcal{Q}$ is the sequence of record sets that match each range in $\mathcal{Q}_R$. While both $\mathcal{Q}$ and $\mathcal{Q}_R$ are defined as sequences, we will often regard them as multisets, and write e.g. $[x,y] \in \mathcal{Q}_R$. We will also (somewhat abusively) call $\mathcal{Q}$ the set of queries.
- For V a set of values, the *diameter* of V is defined as:

  $$\mathsf{diam}(V) \stackrel{\text{def}}{=} \max\{y - x : x, y \in V\}.$$

  The diameter of a set of records S is defined as $\mathsf{diam}(\mathsf{val}(S))$.

Order on records.

- If r, s are records, we write $r < s$ iff $\mathsf{val}(r) < \mathsf{val}(s)$, and likewise for $\leq$. We call $<$ the *real* order on records (as in: the order induced by the actual record values). (This is technically a small abuse of terminology, since two distinct records with the same value satisfy $r \leq s$ and $s \leq r$, so $\leq$ is a total preoder and not an order, but this is irrelevant for our purpose: for simplicity, we will call total preorders orders throughout this work.)
- The notation $\preceq$ will be used to denote orders on records, with $\prec$ being the corresponding strict order. In particular $\prec$ will always denote the strict variant of the order. Since this notation avoids any ambiguity regarding the strictness of the order, we will also call $\prec$ an order.

- We say that an order $\prec$ on records is *compatible* with $\mathcal{Q}$ iff all elements of $\mathcal{Q}$ are intervals for $\prec$. That is, for every $q \in \mathcal{Q}$, there exist two records a_q, b_q such that $q = \{r : a_q \preceq r \preceq b_q\}$.
- We say that an order $\prec$ on records *matches the real order up to reflection* on a set of records $r_1, \ldots, r_k$ iff either: (1) $\forall i \neq j, r_i < r_j \Leftrightarrow r_i \prec r_j$; or (2) $\forall i \neq j, r_i < r_j \Leftrightarrow r_i \succ r_j$.
- If A, B are sets of values, we write $A < B$ iff:

$$\forall a \in A, \forall b \in B, a < b.$$

  The notation is naturally extended to sets of records via val: $S < T$ iff $\mathsf{val}(S) < \mathsf{val}(T)$. Note that if either set is empty, the statement is vacuously true.

PQ-trees.

- If $\mathcal{T}$ is a PQ-tree, and T is a node of $\mathcal{T}$, then the leaves of T are defined as the leaves of the subtree rooted at T, and denoted $\mathsf{leaf}(T)$.
- If $\mathcal{T}$ is a PQ-tree, and $T_1, \ldots, T_k$ are nodes of $\mathcal{T}$, then the *meet* of $T_1, \ldots, T_k$, denoted $\mathsf{meet}(T_1, \ldots, T_k)$, is the deepest node of $\mathcal{T}$ whose descendants contain all T_i's. If A is a set of nodes, $\mathsf{meet}(A)$ is the meet of the elements of A.
- If $\mathcal{T}$ is a PQ-tree, $\mathsf{root}(\mathcal{T})$ is the root of $\mathcal{T}$.
- For S, T two nodes of a tree, $S \leq T$ means that S is a descendant of T. In general, we view trees as having their root at the top; and so if $S \leq T$ we may say that S is lower (or deeper) than T.

D.1 Algorithm

Throughout, we assume ϵN is a strictly positive integer[3]. For convenience, we recall the ApproxOrder attack in Algorithm 7.

Below, we restate the success condition of sacrificial ϵ-approximate order reconstruction (sacrificial ϵ-AOR) in terms of the output of Algorithm 7.

Definition D.1. *Given a target precision $\epsilon \in]0, 1/4]$, we say that the* ApproxOrder *procedure defined in Algorithm 7* succeeds *iff its output $A_1, \ldots, A_k$ satisfies the following three properties:*

1. $\forall i, \mathsf{diam}(A_i) < \epsilon N$.
2. $A_1 < \cdots < A_k$ *holds up to reflection.*
3. *For all $r \notin \bigcup A_i$, $\mathsf{val}(r) \in [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$.*

Recall that we require two hypotheses, h_2 and h_3, for the success of the algorithm. These hypotheses are restated below for convenience.

- Hypothesis h_2 requires that there exist two records with values a and b such that $a, b \in [N/4, 3N/4]$, and $b - a \geq N/3$ (what really matters for the proof to go through is that a, b should be $\Omega(N)$ away from 1, N and each other); and additionally that there exist at least three records with values within $[\epsilon N, N + 1 - \epsilon N]$ that are more than ϵN away from each other (note that a and b can be two of these values).

[3]This is without loss of generality, because the relevant quantity in the definition of sacrificial ϵ-AOR is $\lceil \epsilon N \rceil$ (much like sacrificial ϵ-ADR). That is, ϵN can be replaced by $\lceil \epsilon N \rceil$ everywhere in the definition, and this does not affect whether it is satisfied.

Algorithm 7 Approximate order algorithm.

ApproxOrder($\mathcal{Q}$):
Input: Set of queries $\mathcal{Q}$.
Output: Disjoint subsets of records $A_1, \ldots, A_k$.
1: $\mathcal{T} \leftarrow$ PQ-tree built from $\mathcal{Q}$.
2: $T \leftarrow$ FindNodeT($\mathcal{T}$, root($\mathcal{T}$))
3: $C_1, \ldots, C_k \leftarrow$ children of T (in order)
4: **return** leaf(C_1), ..., leaf(C_k)

FindNodeT($\mathcal{T}, S$):
Input: PQ-tree $\mathcal{T}$ and node S of $\mathcal{T}$.
Output: Deepest node $S' \leq S$ with $> R/2$ leaves.
1: $R \leftarrow |\mathsf{leaf}(\mathsf{root}(T))|$
2: **for** each child C of S **do**
3: **if** $|\mathsf{leaf}(\mathcal{T}, C)| > R/2$ **then**
4: **return** FindNodeT($\mathcal{Q}, \mathcal{T}, C$)
5: **end if**
6: **end for**
7: **return** S

- Hypothesis h_3 requires that a strict majority of all records have a value within $[\epsilon N, N + 1 - \epsilon N]$, and that no range of length ϵN contains the values of a (strict) majority of all records.

We are now in a position to state our main result.

Theorem D.2. *Let $\epsilon < 1/4$. Assume the database under attack satisfies hypotheses h_2 and h_3. Then after*

$$\mathcal{O}\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon} + \frac{1}{\epsilon}\log\frac{1}{\delta}\right).$$

queries, Algorithm 7 succeeds with probability at least $1 - \delta$.

The proof of Theorem D.2 is given in the next section.

Remark. Intuitively what we are showing is that when building a PQ-tree from uniform ranges, assuming some fairly mild hypotheses expressed by h_2 and h_3 about the database not being overly sparse or overly concentrated on a few values, then quite quickly the PQ-tree contains a Q-node near the root that provides a considerable amount of information about the order (each of its children has small diameter, their order is known since it is a Q-node, and collectively they contain all values save a small fraction near 1 and N). Moreover that Q-node can be located easily because it is the deepest node containing a strict majority of records.

D.2 Analysis

Fix a target precision $\epsilon < 1/4$, a set of queries $\mathcal{Q}$ and let $\mathcal{T}$ denote the PQ-tree built from $\mathcal{Q}$. We assume hypotheses h_2 and h_3 are satisfied, in particular there exist two records r_a and r_b with respective values a, b such that $a, b \in [N/4, 3N/4]$ and $b - a \geq N/3$. We assimilate each record occuring in $\mathcal{Q}$ with the corresponding leaf of $\mathcal{T}$. Let $T \stackrel{\text{def}}{=} \mathsf{meet}(r_a, r_b)$.

The proof of Theorem D.2 will proceed as follows. First, we will define a collection of events whose intersection E implies that Algorithm 7 succeeds. Then we will show that E is satisfied after the claimed number of queries (except with probability δ). More precisely, the proof is composed of the following three claims.

Claim D.3. *If E occurs and h_2 holds, then letting $C_1, \dots, C_k$ be the children of T and $A_i = \mathsf{leaf}(C_i)$ for all i, then $A_1, \dots, A_k$ is a successful output for algorithm* ApproxOrder*; that is:*

1. $\forall i, \mathsf{diam}(A_i) < \epsilon N$.
2. $A_1 < \cdots < A_k$ *holds up to reflection.*
3. *For all* $r \notin \bigcup A_i$, $\mathsf{val}(r) \in [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$.

Claim D.4. *If E occurs and h_3 holds, then the output of* FindNodeT$(\mathcal{Q}, \mathcal{T}, \mathsf{root}(\mathcal{T}))$ *is T.*

Claim D.5. *After*

$$\mathcal{O}\left(\frac{1}{\epsilon}\log\frac{1}{\epsilon} + \frac{1}{\epsilon}\log\frac{1}{\delta}\right).$$

queries, E holds with probability at least $1 - \delta$.

It is clear that claims D.3 to D.5 together imply Theorem D.2. We now define the event E. We will then prove claims D.3 to D.5 in appendices D.3 to D.5 respectively.

Before continuing, a few more definitions are needed.

$$\begin{aligned} g &= \lfloor \epsilon N/2 \rfloor. \\ d &= \lfloor \epsilon N/2 \rfloor = g. \\ k_L &= \lceil a/g \rceil - 1. \\ k_M &= \lceil (b-a)/g \rceil. \\ k_R &= \lceil (N+1-b)/g \rceil - 1. \end{aligned}$$

We now define various sub-intervals of $[1, N]$ which we will call *buckets*.

$$\begin{aligned} B_i^L &= [1 + i \cdot g, 1 + (i+1) \cdot g[\text{ for } 1 \le i < k_L. \\ B_i^{MR} &=]b - (i+1) \cdot g, b - i \cdot g] \text{ for } 1 \le i < k_M. \\ B_i^R &= [N - (i+1) \cdot g, N - i \cdot g[\text{ for } 1 \le i < k_R. \end{aligned}$$

The event E will be defined as the intersection of a number of simple events, defined as follows.

$$\begin{aligned} E_a&: && \exists [x,y] \in \mathcal{Q}_R, x \in]a-g, a], y \ge b. \\ E_b&: && \exists [x,y] \in \mathcal{Q}_R, x \le a, y \in [b, b+g[. \\ E_L&: && \exists [x,y] \in \mathcal{Q}_R, x \le d, y \in [a, b[. \\ E_R&: && \exists [x,y] \in \mathcal{Q}_R, x \in]a, b], y \ge N+1-d. \\ E_i^{B,L}&: && \exists [x,y] \in \mathcal{Q}_R, x \in B_i^L, y \ge a. \\ E_i^{B,R}&: && \exists [x,y] \in \mathcal{Q}_R, x \le b, y \in B_i^R. \\ E^{B,LM}&: && \exists [x,y] \in \mathcal{Q}_R, x \le a, y+1 \in E_0^{B,MR}. \\ E_i^{B,MR}&: && \exists [x,y] \in \mathcal{Q}_R, x \in B_i^{MR}, y \ge b. \\ E^{B,L}&: && \textstyle\bigcap_{1 \le i < k_L} E_i^{B,L}. \\ E^{B,R}&: && \textstyle\bigcap_{1 \le i < k_R} E_i^{B,R}. \\ E^{B,MR}&: && \textstyle\bigcap_{1 \le i < k_M} E_i^{B,MR}. \\ E^{B,M}&: && E^{B,LM} \cap E^{B,MR}. \end{aligned}$$

We can finally define:

$$E \stackrel{\text{def}}{=} E_a \cap E_b \cap E_L \cap E_R \cap E^{B,L} \cap E^{B,M} \cap E^{B,R}.$$

D.3 Proof of Claim D.3

Throughout this proof, we assume event E occurs, and h_2 holds. Our goal is to prove the conclusion of Claim D.3.

We now give names to some values that witness the occurence of events that make up E.

- Because E_a holds, there exists a queried range $[s_{LM}, s'_{LM}]$ with $s_{LM} \in]a-g, a]$ and $s'_{LM} \geq b$. We pick s_{LM} as high as possible among possible choices.
- Because E_b holds, there exists a queried range $[s'_{MR}, s_{MR}]$ with $s_{MR} \in [b, b+g[$ and $s'_{MR} \leq a$. We pick s_{MR} as low as possible among possible choices.
- Because E_L holds, there exists a queried range $[\lambda_0, \lambda'_0]$ with $\lambda_0 \leq d$ and $\lambda'_0 \in [a, b[$.
- Because E_R holds, there exists a queried range $[\rho'_0, \rho_0]$ with $\rho_0 \geq N+1-d$ and $\rho'_0 \in]a, b]$.
- Because $E^{B,L}$ holds, for all $1 \leq i < k_L$ there exists a queried range $[\lambda_i, \lambda'_i]$ with $\lambda_i \in B_i^L$ and $\lambda'_i \geq a$. The λ_i's may be regarded as splitting values to the left of a into subranges of length less than $2g$.
- Because $E^{B,R}$ holds, for all $1 \leq i < k_R$ there exists a queried range $[\rho'_i, \rho_i]$ with $\rho_i \in B_i^R$ and $\rho'_i \leq b$. The ρ_i's may be regarded as splitting values to the right of b into subranges of length less than $2g$.
- Because $E^{B,LM}$ holds, there exists a queried range $[\mu'_0, \mu_0 - 1]$ with $\mu_0 \in B_0^{MR}$ and $\mu'_0 \leq a$.
- Because $E^{B,MR}$ holds, for all $1 \leq i < k_M$ there exists a queried range $[\mu_i, \mu'_i]$ with $\mu_i \in B_i^{MR}$ and $\mu'_i \geq b$. The μ_i's may be regarded as splitting values between a and b into subranges of length less than $2g$.

It is worth noting that the situation between the λ_i's and the ρ_i's is completely symmetric by reflection. The same holds between a and b, and between s_{LM} and s_{MR}. We will sometimes use this fact in the upcoming proofs to avoid repeating the same reasoning for both λ_i's and ρ_i's.

Let us consider the sequence of values:

$$(\lambda_0, \lambda_1, \ldots, \lambda_{k_L-1}, s_{LM}, \mu_{k_M-1}, \ldots, \mu_0, \\ s_{MR}+1, \rho_{k_R-1}+1, \ldots, \rho_1+1, \rho_0+1).$$

From the above sequence, we perform the following actions:

- If $\lambda_{k_L-1} \geq s_{LM}$, remove λ_{k_L-1} from the sequence.
- If $s_{LM} \geq \mu_{k_M-1}$, remove μ_{k_M-1} from the sequence.
- If $s_{MR}+1 \geq \rho_{k_R-1}+1$, remove $\rho_{k_R-1}+1$ from the sequence.

We call the resulting sequence P and denote its elements (in order) by $p_1, \ldots, p_{n_P}$. Let us define $P_i \stackrel{\text{def}}{=} [p_i, p_{i+1}[$ for $1 \leq i < n_P$. The elements we just removed from the sequence ensure that it is strictly increasing. In fact P is built so that it essentially satisfies the conclusion of Claim D.3, as expressed by Lemma D.7 below. Before proving that lemma, we begin with a few minor facts about the p_i's.

Lemma D.6. *The following holds.*

1. $\lambda_{k_L-2} < s_{LM}$.

2. $\mu_{k_M-2} > a$.

3. $\rho_{k_R-2} > s_{MR}$.

4. *Let i such that P_i contains a. Then if $\mu_{k_M-1} > a$, $P_i = [s_{LM}, \mu_{k_M-1}[$; otherwise $P_i = [s_{LM}, \mu_{k_M-2}[$.*

5. *Let i such that P_i contains b. Then $P_i = [\mu_0, s_{MR} + 1[$.*

6. *The sequence P is strictly increasing.*

Proof. Proof of item 1.

$$\begin{aligned}\lambda_{k_L-2} &< 1 + (k_L - 1)g \\ &= 1 + (\lceil a/g \rceil - 2)g \\ &< 1 + a - g \\ &\leq s_{LM}.\end{aligned}$$

Proof of item 2.

$$\begin{aligned}\mu_{k_M-2} &> b - (k_M - 1)g \\ &= b - (\lceil (b-a)/g \rceil - 1)g \\ &> a.\end{aligned}$$

Proof of item 3. This is the symmetric of item 1 via the reflection symmetry.

Proof of item 4. By item 1 above, $\lambda_{k_L-2} < s_{LM}$, and by construction of P, if $\lambda_{k_L-1} \geq s_{LM}$, λ_{k_L-1} was removed from the sequence. It follows that all λ_i's in the sequence are strictly less than s_{LM}. Moreover observe that if $\mu_{k_M-1} \leq a$, then $[\mu_{k_M-1}, \mu'_{k_M-1}]$ is a queried range that contains a and b, and hence by construction of M we have $s_{LM} \geq \mu_{k_M-1}$. Moreover in that case, by construction of P, μ_{k_M-1} was removed from the sequence. Combining this observation with item 2, we deduce that all μ_i's in the sequence are strictly more than a. Since all λ_i's in the sequence are strictly less than s_{LM}, and all μ_i's in the sequnce are strictly more than a, item 4 follows.

Proof of item 5. It is clear that all μ_i's are less than b by construction, and $s_{MR} \geq b$ by construction. Combining this with item 3, we deduce item 5.

Proof of item 6. We have shown so far that all λ_i's in the sequence are strictly less than s_{LM}; all μ_i's in the sequence are strictly more than $a \geq s_{LM}$; all μ_i's are less than $b < s_{MR} + 1$; and using item 3 all $\rho_i + 1$'s in the sequence are strictly more than s_{MR}. Combining these results with the observation that by construction the λ_i's are strictly increasing; the μ_i's are strictly decreasing; and the ρ_i's are also strictly decreasing, we deduce that the sequence P is strictly increasing. □

Lemma D.7. *P satisfies the following three properties:*

1. *P is strictly increasing.*

2. *$\forall i, p_{i+1} - p_i \leq \epsilon N$.*

3. *$\bigcup P_i$ forms an interval such that $[1, N] \setminus \bigcup P_i \subseteq [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$.*

Proof. We have already shown in Lemma D.6 that P is strictly increasing.

We continue by proving the third point. It is clear that $\bigcup P_i$ forms an interval, by construction of the P_i's; and since $\lambda_0 \leq d \leq \epsilon N/2$, and $\rho_0 \geq N + 1 - d \geq N + 1 - \epsilon N/2$, we have that $[1, N] \setminus \bigcup P_i \subseteq [1, \epsilon N/2[\cup]N + 1 - \epsilon N/2, N] \subseteq [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$ as desired[4].

It remains to show that $\forall i, p_{i+1} - p_i \leq \epsilon N$. This is true whenever the p_i values both correspond to λ_i's (resp. μ_i's, ρ_i's) in the sequence P: indeed consecutive λ_i's (resp. μ_i's, ρ_i's) belong to consecurive buckets of size $g \leq \epsilon N/2$, and hence are at most ϵN apart. Hence we only need to prove something for edge cases, i.e. whenever p_i or p_{i+1} is equal to either s_{LM} or s_{MR}. This leads to four cases.

Case 1: $p_i = s_{LM}$. In the proof of item 4 in Lemma D.6, we have already observed that $s_{LM} \geq \mu_{k_M - 1}$, so $p_{i+1} - p_i = p_{i+1} - s_{LM} \leq p_{i+1} - \mu_{k_M - 1}$. Using Lemma D.6 item 4, p_{i+1} is either $\mu_{k_M - 1}$ or $\mu_{k_M - 2}$, so the conclusion follows.

Case 2: $p_{i+1} = s_{LM}$. In that case, either we have $p_i = \lambda_{k_L - 1}$, and so:

$$\begin{aligned} p_{i+1} - p_i &= s_{LM} - \lambda_{k_L - 1} \\ &\leq a - (1 + (\lceil a/g \rceil - 2)g) \\ &< 2g \\ &\leq \epsilon N. \end{aligned}$$

The other case is that $\lambda_{k_L - 1}$ was removed from the sequence, whence $p_i = \lambda_{k_L - 2}$. However by construction of P, this only happens if $\lambda_{k_L - 1} \geq s_{LM}$, and in that case $p_{i+1} - p_i = s_{LM} - \lambda_{k_L - 2} \leq \lambda_{k_L - 1} - \lambda_{k_L - 2} \leq 2g$.

Case 3: $p_i = s_{MR}$. This case is the symmetric of the case $p_{i+1} = s_{LM}$ just above via the reflection symmetry.

Case 4: $p_{i+1} = s_{MR}$. In that case $p_i = \mu_0$, and we have:

$$\begin{aligned} p_{i+1} - p_i &= s_{MR} - \mu_0 \\ &< b + g - (b - g) \\ &= 2g. \end{aligned}$$

This concludes the proof. □

Informally, the partition of records induced by P is "good enough" to satisfy the conclusion of Claim D.3. In the remainder of the proof, we show that, roughly speaking, the partition induced by the children of node T must be a refinement of P. Hence we will be able to deduce that they satisfy the same three properties, which will conclude the proof.

Recall that r_a, r_b denote records with respective value a and b (which exist by the assumption h_2). Recall that the *real* order on records is the one induced by their values (i.e. the order we are trying to approximately reconstruct).

Lemma D.8. *Let r denote a record such that $\mathsf{val}(r)$, a, b belong to distinct P_i's. Then for every order $\prec$ on records compatible with $\mathcal{Q}$, $\prec$ must match the real order on r, r_a, r_b up to reflection.*

Proof. Let $x = \mathsf{val}(r)$. Since $a < b$ there are only three possibilities for the position of x relative to a and b: $x < a$, $a < x < b$ or $b < x$.

Case 1: $x < a$. We have $x \geq \lambda_0$ by construction of P. It follows that $x \in [\lambda_0, \lambda_0'[$ with $\lambda_0' \in [a, b[$. By Lemma D.6 item 4, we have that $x < s_{LM}$, so $x \notin [s_{LM}, s'_{LM}[$. Thus, we have exhibited two queried

[4]It may be observed that we are proving something a little stronger that what is required. Informally, we are only sacrificing values within $\epsilon N/2$ of 1 and N, rather than within ϵN. Proving the stronger statement makes the proof slightly easier, and does not affect the final result (this is because the upper bound we eventually prove is up to a constant).

ranges (namely $[\lambda_0, \lambda'_0]$ and $[s_{LM}, s'_{LM}]$) such that the first range matches r and r_a but not r_b; and the second matches r_a and r_b but not r. Any order on records compatible with these two ranges must satisfy that r_a is (strictly) between r and r_b. Hence any order compatible with $\mathcal{Q}$ matches the real order up to reflection.

Case 2: $a < x < b$. Using Lemma D.6 items 2 and 4, we have $x > \mu_i > a$ (where $i \in \{k_M - 1, k_M - 2\}$); and on the other hand $x < \mu_0$. Hence $[\mu_i, \mu'_i]$ is a queried range that contains a and x but not b; and $[\mu'_0, \mu_0]$ is a queried range that contains x and b but not a. Hence any order compatible with $\mathcal{Q}$ must satisfy that r is (strictly) between r_a and r_b.

Case 3: $b < x$. This is the symmetric of the first case by the reflection symmetry. □

Lemma D.9. *Let r_x, r_y denote two records such that $\mathsf{val}(r_x)$, $\mathsf{val}(r_y)$, a, b all belong to distinct P_i's. Then for every order $\prec$ on records compatible with $\mathcal{Q}$, $\prec$ must match the real order on r_a, r_b, r_x, r_y up to reflection.*

Proof. Let $x = \mathsf{val}(r_x)$ and $y = \mathsf{val}(r_y)$. Without loss of generality $x < y$. If x and y are separated by either a or b (or both), then Lemma D.8 suffices to conclude. Hence we can restrict to attention to the following three cases: $x < y < a$, $a < x < y < b$, and $b < x < y$. Let $\prec$ denote an arbitrary order compatible with $\mathcal{Q}$.

Case 1: $x < y < a$. Because $[\lambda_0, \lambda'_0]$ is a queried range containing a but not b, we must have either $r_a \prec r_b$ or $r_b \prec r_a$ (otherwise we have both $r_a \succeq r_b$ and $r_b \succeq r_a$, which implies any interval for $\preceq$ contains a iff it contains b, which implies $\prec$ is not compatible with the queried range $[\lambda_0, \lambda'_0]$, a contradiction). Since we are only trying to prove something on $\prec$ up to reflection, we are free to assume $r_a \prec r_b$ (reversing $\prec$ if necessary). Since x and y are assumed to be in distinct P_i's, and using Lemma D.6 item 4, x and y must be in distinct buckets B_i^L's, and there exists λ_i in the sequence P such that $x < \lambda_i \leq y$. Because $[\lambda_0, \lambda'_0]$ contains x and a, but not b; and $[s_{LM}, s'_{LM}]$ contains a and b but not x; we have that records with values in $[\lambda_0, \lambda'_0]$, and records with values in $[s_{LM}, s'_{LM}]$, are two queries in $\mathcal{Q}$, both containing r_a, and both containing a point the other does not contain (r_x and r_b respectively). It follows that records with values in $[\lambda_0, s_{LM}[$ must form a range for $\prec$ (indeed the set difference of two ranges in the previous configuration is always a range). Thus records in $[\lambda_0, s_{LM}[$ form a range for $\prec$ that contains r_x and r_y but not a; while records with values in $[\lambda_i, \lambda'_i]$ form a range that contains r_y and a but not r_x. It follows that $r_x \prec r_y \prec a$ and we are done (the reverse order is impossible due to Lemma D.8 and the assumption $r_a \prec r_b$).

Case 2: $a < x < y < b$. Using Lemma D.6 items 4 and 5 and the fact that x and y are in distinct P_i's, there must exist $i > 0$ such that $[\mu_i, \mu'_i]$ contains y and b but not a (note that by Lemma D.6 item 4, both x and y must be less than μ_0, whence $i > 0$). On the other hand, $[\mu'_0, \mu_0]$ contains x and y but not b. Hence we conclude that $x \prec y \prec b$ holds up to reflection. Combining this with Lemma D.8 yields the conclusion.

Case 3: $b < x < y$. This is the symmetric of the first case by the reflection symmetry. □

Lemma D.10. *Let r_1, r_2, r_3 denote three records such that $\mathsf{val}(r_1)$, $\mathsf{val}(r_2)$, $\mathsf{val}(r_3)$ belong to distinct P_i's. Then for every order $\prec$ on records compatible with $\mathcal{Q}$, $\prec$ must match the real order on r_1, r_2, r_2 up to reflection.*

Proof. This is a direct corollary of Lemma D.9. □

Lemma D.11. *Let Q_i denote the set of records whose value lies in P_i. For every order $\prec$ on records compatible with $\mathcal{Q}$, $Q_1 \prec \cdots \prec Q_{n_P}$ holds up to reflection.*

Proof. It is enough to show this property for three distinct Q_i's. It is then a direct corollary of Lemma D.10. □

The following two lemmas express general properties of PQ-trees.

Lemma D.12. *Let $r_1, \ldots, r_k$ denote $k \geq 3$ records, and assume for every order $\prec$ compatible with $\mathcal{Q}$, $r_1 \prec \cdots \prec r_k$ holds up to reflection. Let $S = \mathsf{meet}(r_1, \ldots, r_k)$. Then the following holds:*

1. *S is a Q-node.*
2. *$\forall i \neq j, \mathsf{meet}(r_i, r_j) = S$.*

Proof. We first prove the second item. Pick distinct r_i, r_j and assume towards contradiction that $\mathsf{meet}(r_i, r_j) = S'$ with $S' < S$ (for the tree order). Since $\mathsf{meet}(r_1, \ldots, r_k) = S > S'$, there must exist r_h such that $\mathsf{meet}(r_i, r_h) > S'$. This means the PQ-tree is compatible with both the order $r_i \prec r_j \prec r_h$; and with the order $r_j \prec r_i \prec r_h$. These two orders are not reflections of each other, which yields a contradiction. It follows that $\mathsf{meet}(r_i, r_j) < S$ is impossible and we are done.

We now turn to proving the first item. Observe that the second item implies that every r_i belongs to a distinct child of S (in fact it is equivalent). If S were a P-node, all orders between the k records would be compatible with the tree. Since we assume $r_1 \prec \cdots \prec r_k$, and $k \geq 3$, this is impossible. It follows that S is a Q-node. □

Lemma D.13. *Let $A_1, \ldots, A_k$ denote non-empty subsets of records with $k \geq 3$, and assume for every order $\prec$ compatible with $\mathcal{Q}$, $A_1 \prec \cdots \prec A_k$ holds up to reflection. Let $A = \bigcup A_i$, and let $S = \mathsf{meet}(A)$. Then for every child C of S, either $\mathsf{leaf}(C) \cap A = \varnothing$, or $\exists i, \mathsf{leaf}(C) \subseteq A_i$.*

The conclusion of the lemma may be expressed as: the partition induced by the children of S refines $A_1, \ldots, A_k$.

Proof. Let C be a child of S, and $L = \mathsf{leaf}(C)$. Assume $\mathsf{leaf}(C) \cap A \neq \varnothing$. Assume towards contradiction $\exists i \neq j, L \cap A_i \neq \varnothing \wedge L \cap A_j \neq \varnothing$. Let $r_1 \in L \cap A_i$ and $r_2 \in L \cap A_j$. Since $S = \mathsf{meet}(A)$ and $S > C$, there must exist a record $r \in A \setminus L$. In that case the PQ-tree is compatible with both the order $r_1 \prec r_2 \prec r$; and with the order $r_2 \prec r_1 \prec r$. These two orders are not reflections of each other, which yields a contradiction. It follows that L cannot have a non-empty intersection with two distinct A_i's, and we are done. □

We now have all the tools needed to finish the proof of Claim D.3. Let Q_i denote the set of records whose value lies in P_i. By assumption h_2, there exist at least three records with values in $]\epsilon N, (1-\epsilon)N]$ that are at least ϵN apart. It follows that there exist at least three distinct non-empty Q_i's.

Combining Lemmas D.11 and D.12, we can deduce the crucial fact that $\mathsf{meet}(\bigcup Q_i) = \mathsf{meet}(r_a, r_b) = T$.

Let $C_1, \ldots, C_{n_C}$ denote the children of T and $A_i = \mathsf{leaf}(C_i)$ for all i. In order to prove Claim D.3, we need to prove three facts about the A_i's. We prove each in turn.

Fact 1: $\forall i, \mathsf{diam}(A_i) < \epsilon N$. By Lemma D.13, either A_i is contained in one of the Q_j's, or it is disjoint from $\bigcup Q_j$. In the first case, By Lemma D.7, we have $\mathsf{diam}(A_i) < \epsilon N$ as desired. In the second case, by Lemma D.7 and the fact that $r_a \notin A_i$, we have that $\mathsf{val}(A_i)$ is contained either in $[1, \epsilon N[$, or in $]N + 1 - \epsilon N, N]$. In either case we also get $\mathsf{diam}(A_i) < \epsilon N$.

Fact 2: $A_1 < \cdots < A_k$ holds up to reflection. By Lemma D.12, T is a Q-node, so this holds.

Fact 3: $\mathsf{val}(\bigcup A_i)$ forms an interval such that $[1, N] \setminus \mathsf{val}(\bigcup A_i) \subseteq [1, \epsilon N[\cup]N + 1 - \epsilon N, N]$. By Lemma D.7, $\bigcup P_i = \mathsf{val}(\bigcup Q_i)$ satisfies this property, and since $\mathsf{meet}(\bigcup Q_i) = T$, $\bigcup Q_i \subseteq \mathsf{leaf}(T) = \bigcup A_i$ so we are done.

This concludes the proof of Claim D.3.

D.4 Proof of Claim D.4

Let S be the node output by FindNodeT$(\mathcal{Q}, \mathcal{T}, \mathsf{root}(\mathcal{T}))$ in Algorithm 7. We want to prove $S = T$.

Since Claim D.4 assumes E occurs, we can apply Claim D.3 to deduce that the node T contains all records whose value lies in $]\epsilon N, (1-\epsilon)N]$; and that all of its children contain values at most ϵN from each other. By assumption h_3, a strict majority of records have their value in $]\epsilon N, (1-\epsilon)N]$, so T contains a strict majority of records. By the same assumption, no interval of length ϵN contains a majority of records, so no child of T can contain a majority of records. It follows that the nodes of the tree containing a (strict) majority of records are exactly T and its ancestors. By construction of S, we deduce that $S = T$.

D.5 Proof of Claim D.5

The claim is a direct application of the ϵ-net theorem. Define the ground set as the set of all ranges in $[1, N]$, with the uniform distribution. The concept set is defined as follows: each choice of $a < b < c < d$ in $[1, N]$ yields one concept $\{[x, y] : x \in [a, b], y \in [c, d]\}$. We claim that the concept space has finite VC dimension. It suffices to show that its growth function is polynomially bounded (in fact it is equivalent). Given n ranges within $[1, N]$, the $2n$ endpoints of the ranges induce a partition of $[1, N]$ into at most $2n + 1$ sub-intervals. This yields at most $(2n + 1)^4$ choices for the values a, b, c, d (indeed any two choices of those four values that do not modify the sub-interval that each value belongs to does not change the resulting concept). Hence the growth function is polynomially bounded, and we are done.

It is straightforward to check that events E_a, E_b, E_L and E_R all belong to that concept space, and have probability $\Theta(1)$. Likewise all events $E_i^{B,L}$, $E_i^{B,R}$, $E^{B,LM}$, $E_i^{B,MR}$ belong to the same concept space, and have probability $\Theta(\epsilon)$; moreover in each case the probability does not depend on i. It follows that in order for all these events to hold simultaneously, it is enough to have a $C\epsilon$-net on the previous concept space, for some constant $C > 0$ (namely pick C so that $C\epsilon$ lower-bounds the probability of all previous events). By the ϵ-net theorem, we get the desired result.

E Generic Query Complexity Lower Bounds

E.1 Lower Bound for Sacrificial ϵ-ADR

Notation.

- For a probabilty $a \in [0, 1]$, let $B(a)$ denote a Bernoulli trial with probability of success a (i.e. a coin flip with probability of heads a).
- Given two probability distributions P and Q, $\delta(P, Q)$ denotes the statistical distance between P and Q.
- Likewise, $D_{\mathrm{KL}}(P, Q)$ denotes the Kullback-Leibler divergence between P and Q. By a small abuse of notation, given $a, b \in]0, 1[$, we write $D_{\mathrm{KL}}(a, b)$ for $D_{\mathrm{KL}}(B(a), B(b))$.

The "counterexamples" we use to prove the lower bounds are very similar to those of the $\Omega(N^4)$ lower bound found in [KKNO16], although the technique we use to prove the bounds themselves is vastly different. We note that Proposition E.2 implies the previously cited bound (as full reconstruction is equivalent to sacrificial $1/N$-approximate reconstruction).

Lemma E.1. *Let $0 < a < 1$ be a constant, let $\epsilon < 1 - a$, and let Q be an integer representing a number of samples. Then the advantage of a computationally unbounded adversary trying to distinguish $B(a)^Q$ from $B(a + \epsilon)^Q$ is $\mathcal{O}(\sqrt{Q}\epsilon)$. In particular, if the adversary has a constant distinguishing advantage, then it must hold that $Q = \Omega(\epsilon^{-2})$.*

Proof. We have:

$$\begin{aligned}
&D_{\mathrm{KL}}(a, a+\epsilon) \\
&= -a \log\left(\frac{a+\epsilon}{a}\right) - (1-a)\log\left(\frac{1-(a+\epsilon)}{1-a}\right) \\
&= -a \log\left(1+\frac{\epsilon}{a}\right) - (1-a)\log\left(1-\frac{\epsilon}{1-a}\right) \\
&= -a\left(\frac{\epsilon}{a} - \frac{\epsilon^2}{2a^2}\right) - (1-a)\left(-\frac{\epsilon}{1-a} - \frac{\epsilon^2}{2(1-a)^2}\right) + o(\epsilon^2) \\
&= \mathcal{O}(\epsilon^2).
\end{aligned}$$

Using Pinsker's inequality we deduce:

$$\begin{aligned}
&\delta(B(a)^Q, B(a+\epsilon)^Q) \\
&\leq \sqrt{D_{\mathrm{KL}}(B(a)^Q, B(a+\epsilon)^Q)/2} \\
&= \sqrt{Q \cdot D_{\mathrm{KL}}(a, a+\epsilon)/2} \\
&= \mathcal{O}(\sqrt{Q}\epsilon).
\end{aligned}$$

To conclude, recall that the statistical distance is an (exact) upper bound on the advantage of any distinguisher. □

We first prove a lower bound on sacrificial ϵ-approximate reconstruction.

Proposition E.2. *Let $\epsilon < 1/5$ such that $\epsilon N > 0$ is an integer. Assume a uniform distribution on range queries. Consider a (computationally unbounded) adversary achieving sacrificial ϵ-Approximate Reconstruction for all databases, and failing with probability at most $\delta < 1/2$ (over the choice of queries). Then the adversary must require $\Omega(\epsilon^{-4})$ queries.*

Proof. For simplicity we assume N is odd. Consider two databases DB_1 and DB_2 as follows. All records in DB_1 have the same value $v_1 = (N+1)/2$. All records in DB_2 have value $v_2 = (N+1)/2 + 2\epsilon N + 1$.

Recall that the probility that a record with value k is hit by a uniform query is:

$$p(k) = \frac{2}{N(N+1)} k(N+1-k).$$

Querying database DB_1 with a range query either yields all records, or none. It constitutes a Bernoulli trial $B(p(v_1))$. Likewise, querying DB_2 with uniform queries yields a Bernoulli trial $B(p(v_2))$.

Now the key fact is that a successful sacrificial ϵ-ADR adversary cannot output the same database for DB_1 and DB_2. This is because the (single) value in each database is $2\epsilon N + 1$ away from the value of the other database, and neither value is within ϵN of 1 or N; hence the value of any given record in the reconstructed database output by the adversary can only be compatible with one of DB_1 or DB_2. Hence a successful sacrificial ϵ-ADR adversary (with probability of failure at most $\delta < 1/2$) yields a distinguisher between $B(p(v_1))^Q$ and $B(p(v_2))^Q$.

We have:

$$\begin{aligned}
p(v_1) - p(v_2) &= \frac{2(v_1 - v_2)}{N(N+1)}(N+1-2v_1 - (v_2 - v_1)) \\
&= \frac{2}{N(N+1)}(v_1 - v_2)^2 \\
&= \mathcal{O}(\epsilon^2).
\end{aligned}$$

A direct application of Lemma E.1 yields that $\Omega(\epsilon^{-4})$ queries are necessary. □

We now turn to sacrificial ϵ-approximate reconstruction in presence of the hypothesis h_1. Recall that h_1 states the existence of a record with (symmetric) value within $[0.2N, 0.3N]$.

Proposition E.3. *Let $\epsilon < 1/20$ such that $\epsilon N > 0$ is an integer. Assume a uniform distribution on range queries. Consider a (computationally unbounded) adversary achieving sacrificial ϵ-Approximate Reconstruction for all databases satisfying h_1, and failing with probability at most $\delta < 1/2$ (over the choice of queries). Then the adversary must require $\Omega(\epsilon^{-2})$ queries.*

Proof. The reasoning is the same as in the proof of Proposition E.2, choosing slightly different databases DB_1 and DB_2. For simplicity we assume N is a multiple of 4. Consider two databases DB_1 and DB_2 as follows. All records in DB_1 have the same value $v_1 = N/4$. All records in DB_2 have value $v_2 = N/4 + 2\epsilon N + 1$.

Querying database DB_1 (resp. DB_2) yields a Bernoulli trial $B(p(v_1))$ (resp. $B(p(v_2))$). Like in the proof of Proposition E.2, a successful ϵ-ADR adversary yields a distinguisher between $B(p(v_1))^Q$ and $B(p(v_2))^Q$. Moreover DB_1 and DB_2 both satisfy h_1, so the reasoning still holds if the adversary is restricted to databases satisfying h_1.

It is straightforward to check that $p(v_1)$ and $p(v_2)$ are $\Theta(1)$. Moreover:

$$\begin{aligned} p(v_2) - p(v_1) &= \frac{2(v_2 - v_1)}{N(N+1)}(N + 1 - 2v_2 - (v_1 - v_2)) \\ &\leq \frac{2(v_2 - v_1)}{N} \\ &= \mathcal{O}(\epsilon). \end{aligned}$$

A direct application of Lemma E.1 yields that $\Omega(\epsilon^{-2})$ queries are necessary. □

E.2 Lower Bound for Sacrificial ϵ-AOR

Recall that sacrificial ϵ-AOR succeeds iff the attacker is able to split the set of records into disjoint buckets $B_1, \ldots, B_k$, with $B \stackrel{\text{def}}{=} \bigcup B_i$, such that:

1. Sacrificed records have value within ϵN of 1 or N. That is, the values of records *not* in B are within $[1, \epsilon N] \cup [(1-\epsilon)N + 1, N]$.
2. The *diameter* (i.e. the largest pairwise difference of values) within each bucket is less than ϵN. Formally, $\forall i, \mathsf{diam}(B_i) < \epsilon N$.
3. The attackers knows the order of records belonging to distinct buckets (up to reflection). Formally, it holds that either $B_1 < B_2 < \cdots < B_k$, or $B_1 > B_2 > \cdots > B_k$ (where "$B_i < B_j$" denotes $\forall x \in B_i, \forall y \in B_j, \mathsf{val}(x) < \mathsf{val}(y)$).

We now state our generic lower bound result for sacrificial ϵ-AOR.

Proposition E.4. *Let $\epsilon < 1/4$ such that $\epsilon N > 0$ is an integer. Assume a uniform distribution on range queries. Consider a (computationally unbounded) adversary achieving sacrificial ϵ-AOR for all databases, and failing with probability at most $\delta < 1/2$ (over the choice of queries). Then the adversary must require $\Omega(\epsilon^{-1} \log \epsilon^{-1})$ queries.*

Before proving Proposition E.4, we introduce two lemmas. The first lemma is a variant of the coupon collector's problem.

Lemma E.5. *Let C denote a set of* coupons*, among which are n* distinguished *coupons for some $n \leq |C|$. Let π denote a probability distribution on C. At each round of the experiment, one coupon is drawn from C according to π. Let $\eta > 0$ be a constant. Assume that after Q rounds, with probability at least η, all n distinguished coupons have been collected. Then it must hold that $Q = \Omega(n \log n)$.*

Remark. The result holds regardless of the underlying distribution π.

Proof. We prove the result by progressing from a more restricted case towards the full statement in three steps. At each step except the first one, we will reduce the problem to an instance of the previous step. Throughout, T is the random variable denoting the number of rounds after which all distinguished coupons have been collected (for the first time). What we want to prove is that $\Pr[T \leq Q] \geq \eta$ implies $Q = \Omega(n \log n)$. For simplicity we assume n is even (the case of odd n is similar).

Step 1. In this first step, we assume that all coupons are distinguished, and have the same probability $1/n$. Then we are faced with an instance of the standard coupon collector's problem. By standard arguments, $\mathbb{E}[T] = nH_n$, where $H_n \stackrel{\text{def}}{=} \sum_{i=1}^{n} 1/i = \Theta(\log n)$ denotes the harmonic series, and $\text{Var}[T] < \pi^2 n^2/6$. Using Chebyshev's inequality it follows that $\Pr[T \leq nH_n/2] = o(1)$, which implies $Q = \Omega(nH_n) = \Omega(n \log n)$ and we are done.

Step 2. In this second step, we lift previous restrictions, and instead assume that all distinguished coupons have probability at most $1/n$. Then it is clear that the number of rounds necessary to collect all distinguished coupons with probability at least η can only be higher than in the previous case, since every distinguished coupon has lower probability. Hence the result still holds.

Step 3. In this final step, we lift all restrictions and consider the general case. Let n' denote the number of distinguished coupons that have probability at most $2/n$. By the pigeonhole principle, $n' \geq n/2$ (otherwise we would have $> n/2$ coupons with probability $> 2/n$, so the total probability mass would be > 1, which is a contradiction). Hence we can choose $n/2$ coupons among distinguished coupons, such that they each have probability at most $2/n$. We can now apply the result of Step 2, where the $n/2$ chosen coupons play the role of distinguished coupons, to deduce that $\Omega(n/2 \log n/2) = \Omega(n \log n)$ rounds are necessary to collect all chosen (a fortiori, distinguished) coupons with probability at least η. □

We now state the second lemma, which contains the bulk of the argument.

Lemma E.6. *Let $I_1, \ldots, I_n$ denote n* disjoint *sub-intervals of $[1, N]$. Assume Q range queries are drawn uniformly at random in $[1, N]$, forming a set of queried ranges $\mathcal{Q}_R$. Let $\eta > 0$ denote a constant, and assume that the following statement holds with probability at least η: each interval I_i is* split *by a queried range, in the following sense:*

$$\forall i \leq n, \exists q \in \mathcal{Q}_R, \quad I_i \cap q \notin \{\varnothing, I_i\}.$$

Then it must hold that $Q = \Omega(n \log n)$.

Remark. The result holds regardless of the choice of intervals, as long as they are disjoint.

Proof. Define a_i, b_i such that for all i, $I_i = [a_i, b_i + 1]$. Without loss of generality, we can assume that each interval I_i has length at least two, i.e. $b_i \geq a_i$. Indeed, it is otherwise impossible to split the interval in two, as defined in the lemma statement, so the lemma becomes vacuous. For each i, define $C_i \stackrel{\text{def}}{=} [a_i, b_i]$. Notice that the C_i's are still disjoint.

Observe that a queried range $[x, y]$ splits an interval I_i in the sense of the the lemma iff either $x - 1 \in C_i$ or $y \in C_i$ (or both). For a queried range $[x, y]$, we are going to think of the events $x - 1 \in C_i$ and $y \in C_i$ as *collecting* the i-th coupon (once either event has occured, the coupon is collected).

From that standpoint, the lemma states that collecting all coupons with probability at least η implies $\Omega(n \log n)$ queries.

One important difference with a standard coupon collector's problem is that at each step, a query can split up to two intervals I_i. So at each step we can collect up to two coupons. Moreover the two coupons we collect at a given step are not independent, since they are derived from the left and right endpoint of the same query. The main idea, which we borrow from the lower bound proof in [LMP18], is to replace the two dependent coupon draws from a given step by two independent draws, using the following argument.

Let P denote the subsets of size at most two of $[1, N-1]$. The uniform distribution $\mathcal{U}$ on (non-empty) ranges in $[1, N]$ induces a distribution $f(\mathcal{U})$ on P defined by $f : [x, y] \mapsto \{x-1, y\} \setminus \{0, N\}$. A queried range $[x, y]$ collects the i-th coupon iff it splits I_i, equivalently, iff $f([x, y])$ intersects C_i. Our problem can now be reformulated as follows: assume we have drawn Q elements of P according to $\mathcal{D} \stackrel{\text{def}}{=} f(\mathcal{U})$. Assume that with probability at least η, we have collected all coupons (i.e. for all i we have drawn an element $[x, y]$ such that $f([x, y])$ intersects C_i). Then we must have queried $\Omega(n \log n)$ elements.

Now consider the distribution $\mathcal{D}'$ on P induced by drawing two elements a and b independently and uniformly at random from $[1, N-1]$, then taking their union $\{a, b\} \in P$. Define the probabilistic mapping $g : P \to P$ by $g(S) = S'$ where:

$$\begin{cases} \text{if } S = \varnothing, \text{ draw } S' \leftarrow \mathcal{D}'; \\ \text{if } S = \{s\} \begin{cases} \text{with probability } \frac{N+2}{4(N-1)} \text{ set } S' = S; \\ \text{else } S' = \{s, t\} \text{ with } t \stackrel{\$}{\leftarrow} [1, N-1] \setminus \{s\}; \end{cases} \\ \text{if } S = \{s, t\}, \text{ set } S' = S. \end{cases}$$

Claim E.7.

$$g(\mathcal{D}) = \mathcal{D}'.$$

Proof. The proof is given in [LMP18]. We reproduce it here for completeness. Observe that both $\mathcal{D}$ and $\mathcal{D}'$ are uniform on subsets of a given size. Hence it suffices to check that they match on subsets of size 0 and 1. For the first case, both distributions assign probability 0 to the empty set so we are done. For the second case, given singleton $\{s\}$, $\mathcal{D}'$ assigns probability $1/(N-1)^2$ and $g(\mathcal{D})$ assigns probability:

$$\begin{aligned} &\frac{2}{N(N+1)} \cdot \frac{1}{(N-1)^2} + \frac{4}{N(N+1)} \cdot \frac{N+2}{4(N-1)} \\ &= \frac{1}{(N-1)^2}. \end{aligned}$$

This concludes the proof of the claim. □

The point of the mapping g is that it only *adds* elements, i.e. for all $S \in P$, $S \subseteq g(S)$ (regardless of the coins used in g). As a consequence, the number of draws necessary to gather all coupons with probability at least η can only be lower with distribution $\mathcal{D}' = g(\mathcal{D})$ than with $\mathcal{D}$. Hence in order to prove our lower bound result for $\mathcal{D}$ (as is our goal), it is enough to prove it for $\mathcal{D}'$.

That is, it is enough to prove the following statement: assume that with probability at least η, after drawing Q elements of P according to $\mathcal{D}'$, all coupons have been collected; then $Q = \Omega(n \log n)$. Recall that distribution $\mathcal{D}'$ was induced by drawing two elements independently and uniformly at random, and taking their union. Hence, drawing Q elements according to $\mathcal{D}'$ and collecting all coupons is equivalent to drawing $2Q$ elements in $[1, N-1]$ independently and uniformly at random, and having at least one element within each interval C_i. We can now directly apply Lemma E.5 to get the result, where the events $x \in C_i$ (for $x \leftarrow\!\!\$\, [1, N-1]$) play the roles of the distinguished coupons. □

We can now prove the main result of the section.

Proof of Proposition E.4. The result must hold for all databases, so we are free to assume that the database is dense (each value in $[1, N]$ is taken by at least one record). Let I_i, $1 \leq i \leq n$, denote n disjoint sub-intervals within $[1 + \epsilon N, (1 - \epsilon)N]$, each of length $\ell \stackrel{\text{def}}{=} \epsilon N + 2$, with $n = \lfloor N(1 - 2\epsilon)/\ell \rfloor$. It is straightforward to show that $n = \Omega(1/\epsilon)$ (using $\epsilon N \geq 1$ and $\epsilon < 1/4$).

Now assume that the conditions of Lemma E.6 are not satisfied, i.e. there exists at least one interval I_i that is not split. The crucial observation is that because I_i is not split by any queried range, all records with value within I_i have the same access pattern leakage, i.e. they are hit by exactly the same set of queries. In particular, by density there must exist two records r and r' with respective values $v \stackrel{\text{def}}{=} \min(I_i)$ and $v' \stackrel{\text{def}}{=} \max(I_i)$, so that their distance is $v' - v > \epsilon N$. Since these records are indistinguishable given the query leakage (they are matched by exactly the same queries), in this situation the adversary has no information about the order between records r and r', so that it can only correctly predict that order with probability $1/2$. Since by assumption the adversary succeeds with probability at least $\eta > 1/2$, it follows that the previous sitation, i.e. there existing an interval I_i not split by any query, can only occur with probability at most $2(1 - \eta)$. Flipping this statement around, it must be the case that with probability at least $2\eta - 1 > 0$, all intervals are split. Hence we can apply Lemma E.6 to deduce that $\Omega(n \log n) = \Omega(\epsilon^{-1} \log \epsilon^{-1})$ queries are necessary. □