Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 89 (2020)

Lower Bounds on the Time/Memory Tradeoff of Function Inversion

Dror Chawin[*] Iftach Haitner[*†] Noam Mazor[*]

October 16, 2020

Abstract

We study time/memory tradeoffs of *function inversion*: an algorithm, i.e., an *inverter*, equipped with an s-bit advice on a randomly chosen function $f\colon [n] \mapsto [n]$ and using q oracle queries to f, tries to invert a randomly chosen output y of f, i.e., to find $x \in f^{-1}(y)$. Much progress was done regarding *adaptive* function inversion—the inverter is allowed to make *adaptive* oracle queries. Hellman [IEEE transactions on Information Theory '80] presented an adaptive inverter that inverts with high probability a random f. Fiat and Naor [SICOMP '00] proved that for any s, q with $s^3q = n^3$ (ignoring low-order terms), an s-advice, q-query variant of Hellman's algorithm inverts a constant fraction of the image points of *any* function. Yao [STOC '90] proved a lower bound of $sq \geq n$ for this problem. Closing the gap between the above lower and upper bounds is a long-standing open question.

Very little is known for the *non-adaptive* variant of the question—the inverter chooses its queries *in advance*. The only known upper bounds, i.e., inverters, are the *trivial* ones (with $s+q = n$), and the only lower bound is the above bound of Yao. In a recent work, Corrigan-Gibbs and Kogan [TCC '19] partially justified the difficulty of finding lower bounds on non-adaptive inverters, showing that a lower bound on the time/memory tradeoff of non-adaptive inverters implies a lower bound on low-depth Boolean circuits. Bounds that, for a strong enough choice of parameters, are notoriously hard to prove.

We make progress on the above intriguing question, both for the adaptive and the non-adaptive case, proving the following lower bounds on restricted families of inverters:

Linear-advice (adaptive inverter). If the advice string is a linear function of f (e.g., $A \times f$, for some matrix A, viewing f as a vector in $[n]^n$), then $s+q \in \Omega(n)$. The bound generalizes to the case where the advice string of $f_1 + f_2$, i.e., the coordinate-wise addition of the truth tables of f_1 and f_2, can be computed from the description of f_1 and f_2 by a *low* communication protocol.

Affine non-adaptive decoders. If the non-adaptive inverter has an *affine decoder*—it outputs a linear function, determined by the advice string and the element to invert, of the query answers—then $s \in \Omega(n)$ (regardless of q).

Affine non-adaptive decision trees. If the non-adaptive inversion algorithm is a d-depth *affine decision tree*—it outputs the evaluation of a decision tree whose nodes compute a linear function of the answers to the queries—and $q < cn$ for some universal $c > 0$, then $s \in \Omega(n/d \log n)$.

Keywords: Function inverters; random functions, time/memory tradeoff.

[*] School of Computer Science, Tel Aviv University. Emails: quefumas@gmail.com,iftachh@cs.tau.ac.il,noammaz@gmail.com. Research supported by ERC starting grant 638121 and Israel Science Foundation grant 666/19.

[†] Member of the Check Point Institute for Information Security.

ISSN 1433-8092

Contents

1 Introduction **1**
1.1 Our Results 1
1.2 Additional Related Work 4

2 Our Technique **6**
2.1 Linear-advice Inverters 6
2.2 Non-adaptive Inverters 7

3 Preliminaries **10**
3.1 Notation 10
3.2 Matrices and Linear Spaces 11
3.3 Random Functions 12
3.4 Additional Inequalities 14

4 Linear-advice Inverters **15**
4.1 Additive-advice Inverters 18

5 Non-adaptive Inverters **18**
5.1 Affine Decoders 19
5.2 Affine Decision Trees 24

1 Introduction

In the *function-inversion* problem, an algorithm, *inverter*, attempts to find a preimage for a randomly chosen $y \in [n]$ of a random function $f\colon [n] \to [n]$. The inverter is equipped with an s-bit advice on f, and may make q oracle queries to f. Since s lowerbounds the inverter space complexity and q lowerbounds the inverter time complexity, it is common to refer to the relation between s and q as the inverter's *time/memory tradeoff*. The function-inversion problem is central to both theoretical and practical cryptography. On the theoretical end, the security of many systems relies on the existence of one-way functions. While the task of inverting one-way functions is very different from that of inverting random functions, understanding the latter task is critical towards developing lower bounds on the possible (black-box) implications of one-way functions, e.g., Impagliazzo and Rudich [17], Gennaro et al. [13]. But advances on this problem (at least on the positive side, i.e., inverters) are likely to find practical applications. Indeed, algorithms for function inversion are used to expose weaknesses in existing cryptosystems.

Much progress was done regarding *adaptive* function inversion—the inverter may choose its queries adaptively (i.e., based on answers for previous queries). Hellman [16] presented an adaptive inverter that inverts with high probability a random f. Fiat and Naor [11] proved that for any s, q with $s^3 q = n^3$ (ignoring low-order terms), an s-advice q-query variant of Hellman's algorithm inverts a constant fraction of the image points of *any* function. Yao [25] proved a lower bound of $s \cdot q \geq n$ for this problem. Closing the gap between the above lower and upper bounds is a long-standing open question. In contrast, very little is known about the non-adaptive variant of this problem—the inverter performs all queries at once. This variant is interesting since such inverter is likely be highly parallelizable, making it significantly more tractable for real world applications. The only known upper bounds for this variant, i.e., inverters, are the *trivial* ones (i.e., $s + q = n$), and the only known lower bound is the above bound of Yao [25]. In a recent work, Corrigan-Gibbs and Kogan [8] have partially justified the difficulty of finding lower bounds on this seemingly easier to tackle problem, showing that lower bounds on non-adaptive inversion imply circuit lower bounds that, for strong enough parameters, are notoriously hard (see further details in Section 1.1.3).

1.1 Our Results

We make progress on this intriguing question, proving lower bounds on restricted families of inverters. To state our results, we use the following formalization to capture inverters with a preprocessing phase: such inverters have two parts, the *preprocessing* algorithm that gets as input the function to invert f and outputs an advice string a, and the *decoding* algorithm that takes as input the element to invert y, the advice string a, and using restricted query access to f tries to find a preimage of y. We start with describing our bound for the time/memory tradeoff of linear-advice (adaptive) inverters, and then present our lower bounds for non-adaptive inverters. In the following, fix $n \in \mathbb{N}$ and let $\mathcal{F}$ be the set of all functions from $[n]$ to $[n]$.

1.1.1 Linear-advice Inverters

We start with a more formal description of adaptive function inverters.

Definition 1.1 (Adaptive inverters, informal)**.** *An* s-advice, q-query adaptive inverter *is a deterministic algorithm pair* $\mathsf{C} := (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{dec}})$*, where* $\mathsf{C}_{\mathsf{pre}} : \mathcal{F} \to \{0,1\}^s$*, and* $\mathsf{C}_{\mathsf{dec}}^{(\cdot)} : [n] \times \{0,1\}^s \to [n]$ *is a q-query algorithm. We say that* C inverts $\mathcal{F}$ with high probability *if*

$$\Pr_{\substack{f \leftarrow \mathcal{F} \\ a := \mathsf{C}_{\mathsf{pre}}(f)}} \left[\Pr_{\substack{x \leftarrow [n] \\ y := f(x)}} \left[\mathsf{C}^{f}_{\mathsf{dec}}(y, a) \in f^{-1}(y) \right] \geq 1/2 \right] \geq 1/2.$$

It is common to refer to a ($:= \mathsf{C}_{\mathsf{pre}}(f)$) as the *advice string.* In *linear-advice* inverters, the preprocessing algorithm $\mathsf{C}_{\mathsf{pre}}$ is restricted to output a linear function of f. That is, $\mathsf{C}_{\mathsf{pre}}(f_1) + \mathsf{C}_{\mathsf{pre}}(f_2) = \mathsf{C}_{\mathsf{pre}}(f_1 + f_2)$, where the addition $f_1 + f_2$ is coordinate-wise with respect to an arbitrary group over $[n]$, and the addition $\mathsf{C}_{\mathsf{pre}}(f_1) + \mathsf{C}_{\mathsf{pre}}(f_2)$ is over an arbitrary group that contains the image of $\mathsf{C}_{\mathsf{pre}}$. An example of such a preprocessing algorithm is $\mathsf{C}_{\mathsf{pre}}(f) := A \times f$, for $A \in \{0,1\}^{s \times n}$, viewing $f \in \mathcal{F}$ as a vector in $[n]^n$. For such inverters, we present the following bound.

Theorem 1.2 (Bound on linear-advice inverters)**.** *Assume there exists an s-advice q-query inverter with linear preprocessing that inverts $\mathcal{F}$ with high probability. Then $s + q \cdot \log n \in \Omega(n)$.*

We prove Theorem 1.2 via a reduction from *set disjointness*, a classical problem in the study of two-party communication complexity. The above result generalizes to the following bound that replaces the restriction on the decoder (e.g., linear and short output) with the ability to compute the advice string of $f_1 + f_2$ by a low-communication protocol over the inputs f_1 and f_2.

Theorem 1.3 (Bound on additive-advice inverters, informal)**.** *Assume there exists a q-query inverter $\mathsf{C} := (\mathsf{C}_{\mathsf{pre}}, \cdot)$ and an s-bit communication two-party protocol $(\mathsf{P}_1, \mathsf{P}_2)$ such that for every $f_1, f_2 \in \mathcal{F}$, the output of P_1 in $(\mathsf{P}_1(f_1), \mathsf{P}_2(f_2))$ equals with constant probability to $\mathsf{C}_{\mathsf{pre}}(f_1 + f_2)$. Then $s + q \cdot \log n \in \Omega(n)$.*

The above bound indeed generalizes Theorem 1.2: a preprocessing algorithm of the type required by Theorem 1.2 immediately implies a two-party protocol of the type required by Theorem 1.3.

1.1.2 Non-adaptive Inverters

In the non-adaptive setting, the decoding algorithm has two phases: the *query selection* algorithm that chooses the queries as a function of the advice and the element to invert y, and the actual decoder that receives the answers to these queries along with the advice string and y.

Definition 1.4 (Non-adaptive inverters, informal)**.** *An s-advice, q-query non-adaptive inverter is a deterministic algorithm triplet of the form $\mathsf{C} := (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{qry}}, \mathsf{C}_{\mathsf{dec}})$, where $\mathsf{C}_{\mathsf{pre}} \colon \mathcal{F} \to \{0,1\}^s$, $\mathsf{C}_{\mathsf{qry}} \colon [n] \times \{0,1\}^s \to [n]^q$ and $\mathsf{C}_{\mathsf{dec}} \colon [n] \times \{0,1\}^s \times [n]^q \to [n]$. We say that C inverts $\mathcal{F}$ with high probability if*

$$\Pr_{\substack{f \leftarrow \mathcal{F} \\ a = \mathsf{C}_{\mathsf{pre}}(f)}} \left[\Pr_{\substack{x \leftarrow [n] \\ y = f(x) \\ v = \mathsf{C}_{\mathsf{qry}}(y, a)}} \left[\mathsf{C}_{\mathsf{dec}}(y, a, f(v)) \in f^{-1}(y) \right] \geq 1/2 \right] \geq 1/2.$$

Note that the query vector v is of length q, so the answer vector $f(v)$ contains q answers. Assuming there exists a field $\mathbb{F}$ of size n (see Remark 1.7), we provide two lower bounds for such inverters.

Affine decoders. The first bound regards inverters with *affine decoders.* A decoder algorithm $\mathsf{C}_{\mathsf{dec}}$ is *affine* if it computes an affine function of f's answers. That is, for every image $y \in [n]$ and advice $a \in \{0,1\}^s$, there exists a q-sparse vector $\alpha_y^a \in \mathbb{F}^n$ and a field element $\beta_y^a \in \mathbb{F}$ such that $\mathsf{C}_{\mathsf{dec}}(y, a, f(\mathsf{C}_{\mathsf{qry}}(y, a))) = \langle \alpha_y^a, f \rangle + \beta_y^a$ for every $f \in \mathcal{F}$. For this type of inverters, we present the following lower bound.

Theorem 1.5 (Bound on non-adaptive inverters with affine decoders, informal)**.** *Assume there exists an s-advice non-adaptive function inverter with an affine decoder, that inverts $\mathcal{F}$ with high probability. Then $s \in \Omega(n)$.*

Note that the above bound on s holds even if the inverter queries f on all inputs. While Theorem 1.5 is not very insightful for its own sake, as we cannot expect a non-adaptive inverter to have such a limiting structure, it is important since it can be generalized to *affine decision trees*, a much richer family of non-adaptive inverters defined below. In addition, the result should be contrasted with the question of *black-box function computation*, see Section 1.2.4, for which linear algorithm are *optimal*. Thus, Theorem 1.5 highlights the differences between these two related problems.

Affine decision trees. The second bound regards inverters whose decoders are *affine decision trees*. An *affine decision tree* is a decision tree whose nodes compute an *affine* function, over $\mathbb{F}$, of the input vector. A decoder algorithm $\mathsf{C}_{\mathsf{dec}}$ is an *affine decision tree*, if for every image $y \in [n]$, advice $a \in \{0,1\}^s$ and queries $v = \mathsf{C}_{\mathsf{qry}}(y, a)$, there exists an affine decision tree $\mathcal{T}^{y,a}$ such that $\mathsf{C}_{\mathsf{dec}}(y, a, f(v)) = \mathcal{T}^{y,a}(f)$ (i.e., the output of $\mathcal{T}^{y,a}$ on input f) for every $f \in \mathcal{F}$. For such inverters, we present the following bound.

Theorem 1.6 (Bounds on non-adaptive inverters with affine decision-tree decoders)**.** *Assume there exists an s-advice q-query non-adaptive function inverter with a d-depth affine decision-tree decoder, that inverts $\mathcal{F}$ with high probability. Then the following hold:*

- *$q < cn$, for some universal constant c, $\implies$ $s \in \Omega(n/d \log n)$.*
- *$q \in n^{1-\Theta(1)}$ $\implies$ $s \in \Omega(n/d)$.*

That is, we pay a factor of $1/d$ comparing to the affine decoder bound, and the bound on s only holds for not too large q. Affine decision trees are much stronger than affine decoders, since the choice of the affine operations it computes can be *adaptively dependent* on the results of previous affine operations. For example, a depth d affine decision tree can compute *any* function on d linear combinations of the inputs. In particular, multiplication of function values, e.g., $f(1) \cdot f(2)$, which cannot be computed by an affine decoder, can be computed by a depth two decision tree. We note that an affine decision tree of depth q can compute *any* function of its q queries. Unfortunately, for $d = q$, our bound only reproduces (up to log factors) the lower bound of Yao [25].

Remark 1.7 (Field size)**.** *In Theorems 1.5 and 1.6, the field size is assumed to be exactly n (the domain of the function to invert). Decoders over fields smaller than n are not particularly useful, since their output cannot cover all possible preimages of f. Our proof breaks down for fields of size larger than n, since we cannot use linear equations to represent the constraint that the decoder's output must be contained in the smaller set $[n]$.*

1.1.3 Applications to Valiant's Common-bit Model

Corrigan-Gibbs and Kogan [8] showed that a lower bound on the time/memory tradeoff of *strongly non-adaptive* function inverters—the queries may not depend on the advice—implies a lower bound on circuit size in *Valiant's common-bit model* [22, 23]. Applying the reduction of [8] with Theorem 1.6 yields the following bound: for every $n \in \mathbb{N}$ for which there exits an n-size field $\mathbb{F}$, there is an explicit function $f \colon \mathbb{F}^n \mapsto \mathbb{F}^n$ that cannot be computed by a three-layer circuit of the following structure:

1. It has $o({}^n\!/_{d \log n})$ middle layer gates.
2. Each output gate is connected to $n^{1-\Theta(1)}$ inputs gates (and to an arbitrary number of middle-layer gates).
3. Each output gate computes a function which is computable by a d-depth affine decision tree in the inputs (and depends arbitrarily on the middle layer).

In fact, our bound yields that such circuits cannot even approximate f so that every output gate outputs the right value with probability larger than $1/2$, over the inputs.

1.2 Additional Related Work

1.2.1 Adaptive Inverters

Upper bounds. The main result in this setting is the s-advice, q-query inverter of Hellman [16], Fiat and Naor [11] that inverts a constant fraction of the image of any function, for any s, q such that $s^3 q = n^3$ (ignoring low-order terms). When used for random *permutations*, a variant on the same idea implies an optimum inverter with $s \cdot q = n$. The inverter of Hellman, Fiat and Naor has found application to practical cryptanalysis, e.g., Biryukov and Shamir [5], Biryukov et al. [6], Oechslin [19].

Lower bounds. A long line of research (Gennaro et al. [13], Dodis et al. [10], Abusalah et al. [1], Unruh [21], Coretti et al. [7], De et al. [9]) provides lower bounds for various variations on the classical setting, such as that of randomized inversion algorithms that succeed on a sub-constant fraction of functions. None of these lower bounds, however, manage to improve on Yao's lower bound of $s \cdot q = n$, leaving a large gap between this lower bound and Hellman, Fiat and Naor's inverter.

1.2.2 Non-adaptive Inverters

Upper bounds. In contrast with the adaptive case, it is not clear how to exploit non-adaptive queries in a non trivial way. Indeed, the only known inverters are the trivial ones (roughly, the advice is the function description, or the inverter queries the function on all inputs).

Lower bounds. Somewhat surprisingly, the only known lower bound for non-adaptive inverters is Yao's, mentioned above. This defies the basic intuition that this task should be easier than the adaptive case, due to the extreme limitations under which non-adaptive inverters operate. This difficulty was partially justified by the recent reduction of Corrigan-Gibbs and Kogan [8] (see Section 1.1.3) that implies that a strong enough lower bound on even strongly non-adaptive inverters, would yield a lower bound on low-depth Boolean circuits that is notoriously hard to prove.

1.2.3 Relation to Data Structures

The problem of function inversion with advice may also be phrased as a problem in data structures, where the advice string serves as a succinct data structure for answering questions about f. In particular, it bears strong similarity to the *substring search* problem using the cell-probe model [26]. This is the task of ascertaining the existence of a certain element within a large, unsorted database, using as few queries to the database and as little preprocessing as possible. Upper and lower bounds easily carry over between the two problems, a connection which was made in Corrigan-Gibbs and Kogan [8], where it was used to obtain previously unknown upper bounds on substring search.

1.2.4 Index Coding and Black-box Function Computation

A syntactically related problem to function inversion is the so-called *black-box function computation*: an algorithm tries to compute $f(x)$, for a randomly chosen x, using an advice of length s on f, and by querying f on q inputs other than x. Yao [24] proved that $s \cdot q \geq n$, and presented a linear, non-adaptive algorithm that matches this lower bound.

A much-researched special case of this problem is known as the *index coding* problem [4], originally inspired by information distribution over networks. In this setting, a single party is in possession of a vector f, and broadcasts a short message a such that n different recipients may each recover a particular value of f, using the broadcast message and knowledge of certain other values of f, as determined by a *knowledge graph*. The analogy to non-adaptive black-box function computation is obvious when considering a as the advice string, and the access to various values of f as queries. While Yao's bound on the time/memory tradeoff also holds for the index coding problem, other lower bounds, some of which consider "linear" algorithms [4, 15, 18, 14, 3], do not seem to be relevant for the function inversion problem.

Open Questions

The main challenge remains to gain a better understanding on the power of adaptive and non-adaptive function inverters. A more specific challenge is to generalize our bound on affine decoders and decision trees to affine operations over arbitrary (large) fields.

Paper Organization

A rather detailed description of our proof technique is given in Section 2. Basic notations, definitions and facts are given in Section 3, where we also prove several basic claims regarding random function inversion. The bound on linear-advice inverters is given in Section 4, and the bounds on non-adaptive inverters are given in Section 5.

2 Our Technique

In this section we provide a rather elaborate description of our proof technique. We start with the bound on linear-advice inverters in Section 2.1, and then in Section 2.2 describe the bounds for non-adaptive inverters.

2.1 Linear-advice Inverters

Our lower bound for inverters with linear advice (and its immediate generalization to additive-advice inverters) is proved via a reduction from *set disjointness*, a classical problem in the study of two-party communication complexity. In the set disjointness problem, two parties, Alice and Bob, receive two subsets, $\mathcal{X}$ and $\mathcal{Y} \subseteq [n]$, respectively, and by communicating with each other try to determine whether $\mathcal{X} \cap \mathcal{Y} = \emptyset$. The question is how many bits the parties have to exchange in order to output the right answer with high probability. Given an inverter with linear advice, we use it to construct a protocol that solves the set disjointness problem on *all* inputs in $\mathcal{Q} := \{\mathcal{X}, \mathcal{Y} \subseteq [n] \colon |\mathcal{X} \cap \mathcal{Y}| \leq 1\}$ by exchanging $s + q \cdot \log n$ bits. Razborov [20] proved that to answer with constant success probability on all input pairs in $\mathcal{Q}$, the parties have to exchange $\Omega(n)$ bits. Hence, the above reduction implies the desired lower bound on the time/memory tradeoff of such inverters.

Fix a q-query s-advice inverter $\mathsf{C} := (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{dec}})$ with linear advice, and assume for simplicity that C's success probability is one. The following observation immediately follows by definition: let $a_f := \mathsf{C}_{\mathsf{pre}}(f)$ and $a_g := \mathsf{C}_{\mathsf{pre}}(g)$ be the advice strings for some functions f and $g \in \mathcal{F}$, respectively. The linearity of $\mathsf{C}_{\mathsf{pre}}$ yields that $a := a_f + a_g = \mathsf{C}_{\mathsf{pre}}(f + g)$. That is, a is the advice for the function $f + g$ (all additions are over the appropriate groups). Given this observation, we use C to solve set disjointness as follows: Alice and Bob (locally) convert their input sets $\mathcal{X}$ and $\mathcal{Y}$ to functions f_{A} and f_{B} respectively, such that for any $x \in \mathcal{X} \cap \mathcal{Y}$ it holds that $f(x) := (f_{\mathsf{A}} + f_{\mathsf{B}})(x) = 0$, and $f(x)$ is *uniform* for $x \notin \mathcal{X} \cap \mathcal{Y}$. Alice then sends $a_{\mathsf{A}} := \mathsf{C}_{\mathsf{pre}}(f_{\mathsf{A}})$ to Bob who uses it to compute $a := \mathsf{C}_{\mathsf{pre}}(f) = a_{\mathsf{A}} + \mathsf{C}_{\mathsf{pre}}(f_{\mathsf{B}})$. Equipped with the advice a and the help of Alice, Bob then emulates $\mathsf{C}_{\mathsf{dec}}(0, a)$ and finds $x \in f^{-1}(0)$, if such exists. Since f is unlikely to map many elements outside of $\mathcal{X} \cap \mathcal{Y}$ to 0, finding such x is highly correlated with $\mathcal{X} \cap \mathcal{Y} \neq \emptyset$. In more details, the set disjointness protocol is defined as follows.

Protocol 2.1 (Set disjointness protocol $\Pi = (\mathsf{A}(\mathcal{X}), \mathsf{B}(\mathcal{Y}))$).

1. A *samples* $f_{\mathsf{A}} \in \mathcal{F}$ *by letting* $f_{\mathsf{A}}(i) := \begin{cases} 0 & i \in \mathcal{X} \\ \sim [n] & \textit{otherwise.} \end{cases}$

2. B *samples* $f_{\mathsf{B}} \in \mathcal{F}$ *analogously, with respect to* $\mathcal{Y}$.

– *Let* $f := f_{\mathsf{A}} + f_{\mathsf{B}}$.

3. A *sends* $a_{\mathsf{A}} := \mathsf{C}_{\mathsf{pre}}(f_{\mathsf{A}})$ *to* B*, and* B *sets* $a := a_{\mathsf{A}} + \mathsf{C}_{\mathsf{pre}}(f_{\mathsf{B}})$. [1]

4. B *emulates* $\mathsf{C}^{f}_{\mathsf{dec}}(0, a)$ *while answering each query* r *that* $\mathsf{C}_{\mathsf{dec}}$ *makes to* f *as follows:*

   (a) B *sends* r *to* A.

[1] If the inverter is only assumed to have additive advice, this step is replaced with the parties interacting in the guaranteed protocol for computing the advice for f from the descriptions of f_{A} and f_{B}.

(b) *A sends $w_{\mathsf{A}} := f_{\mathsf{A}}(r)$ back to B.*

(c) *B replies $w := w_{\mathsf{A}} + f_{\mathsf{B}}(r)$ to $\mathsf{C}_{\mathsf{dec}}$ (as the value of $f(r)$).*

– *Let x be $\mathsf{C}_{\mathsf{dec}}$'s answer at the end of the above emulation.*

5. *The parties reject if $x \in \mathcal{X} \cap \mathcal{Y}$ (using an additional $\Theta(\log n)$ bits to find it out), and accept otherwise.*

The communication complexity of Π is essentially $s + q \cdot \log n$. It is also clear that the parties accept if $\mathcal{X} \cap \mathcal{Y} = \emptyset$. For the complementary case, by construction, the intersection point of $\mathcal{X} \cap \mathcal{Y}$ is in $f^{-1}(0)$. Furthermore, since $f(i)$ is a random value for all $i \notin \mathcal{X} \cap \mathcal{Y}$, with constant probability *only* the intersection point is in $f^{-1}(0)$. Therefore, the protocol is likely to answer correctly also in the case that $|\mathcal{X} \cap \mathcal{Y}| = 1$.

2.2 Non-adaptive Inverters

We focus on inverters with an affine decoder, and discuss the extension to affine decision tree decoders in Section 2.2.1. The proof follows by bounding the success probability of *zero-advice* inverters—the preprocessing algorithm outputs an empty string. In particular, we prove that the success probability of such inverters is at most $2^{-\Omega(n)}$. Thus, by a union bound over all advice strings, in order to invert $\mathcal{F}$ with high probability, the advice string of a general (non-zero-advice) inverter has to be of length $\Omega(n)$.[2] Let $\mathsf{C} := (\mathsf{C}_{\mathsf{qry}}, \mathsf{C}_{\mathsf{dec}})$ be a zero-advice q-query non-adaptive inverter with an affine decoder. Let F be a random element of $\mathcal{F}$, and for $i \in [n]$, let Y_i be a randomly and independently selected element of $[n]$. Let $X_i := \mathsf{C}_{\mathsf{dec}}(Y_i, F(\mathsf{C}_{\mathsf{qry}}(Y_i)))$, i.e., C's answer on challenge Y_i, and let Z_i be the indicator for $\{F(X_j) = Y_j\}$ for all $j \in [i]$, i.e., the event that C answers the first i challenges correctly. We prove the bound by showing that for some $m \in \Theta(n)$ it holds that

$$\Pr\left[Z_m\right] \in 2^{-\Omega(m)} \tag{1}$$

Note that Equation (1) bounds the probability that C inverts m random elements drawn from $[n]$ (where some of them might have no preimage at all), whereas we are interested in bounding the probability that C inverts a *random output* of F. Yet, since F is a random function, its image covers with very high probability a constant fraction of $[n]$, and thus Equation (1) can be easily manipulated to derive that

$$\Pr_{f \leftarrow \mathcal{F}}\left[\Pr_{\substack{x \leftarrow [n] \\ y = f(x) \\ v = \mathsf{C}_{\mathsf{qry}}(f,y)}}\left[\mathsf{C}_{\mathsf{dec}}(y, f(v)) \in f^{-1}(y)\right] \geq 1/2\right] < 2^{-\Omega(m)} = 2^{-\Omega(n)} \tag{2}$$

Hence, in order to invert a random function with high probability, a non-zero-advice inverter has to use advice of length $\Omega(n)$.

We prove Equation (1) by showing that for every $i \in [m]$ it holds that

$$\Pr\left[Z_i \mid Z_{i-1}\right] < 3/5 \tag{3}$$

[2] This first part of the proof is rather standard, cf., Akshima et al. [2].

That is, for small enough i, the algorithm C is likely to fail on inverting the i^{th} challenge, even when conditioned on the successful inversion of the first $i-1$ challenges. We note that it is easy to bound $\Pr\left[Z_i \mid Z_{i-1}\right]$ for *zero*-query inverters. The conditioning on Z_{i-1} roughly gives $\Theta(i)$ bits of information about F. Thus, this conditioning gives at most one bit of information about $F^{-1}(Y_i)$, and the inverter does not have enough information to invert Y_i. When moving to non-zero-queries inverters, however, the situation gets much more complicated. By making the right queries, that may depend on Y_i, the inverter can exploit this "small" amount of information to find the preimage of Y_i. This is what happens, for instance, in the adaptive inverter of Hellman [16]. Hence, for bounding $\Pr\left[Z_i \mid Z_{i-1}\right]$, we critically exploit the assumption that C is non-adaptive and has an affine decoder. In particular, we bound $\Pr\left[Z_i \mid Z_{i-1}\right]$ by translating the event Z_i into an affine system of equations and then use a few observations about the structure of the above system to derive the desired bound. These equations will have the form $M \times F = V$, viewing F as a random vector in $[n]^n$, for $\mathbf{M} := \begin{pmatrix} \mathbf{M}^{i-1} \\ \mathbf{M}^i \end{pmatrix}$ and $V := \begin{pmatrix} V^{i-1} \\ V^i \end{pmatrix}$, such that:

1. $\mathbf{M}^{i-1}$ is a deterministic function of $(X_{<i}, Y_{<i})$ and $\mathbf{M}^i$ is a deterministic function of Y_i, letting $X_{<i}$ stand for $(X_1, \ldots, X_{i-1})$ and likewise for $Y_{<i}$.

2. The event $M^{i-1} \times F' = V^{i-1}$ is the event $\bigwedge_{j<i} \{(F'(X_j) = Y_j) \wedge (\mathsf{C}_{\mathsf{dec}}(Y_j, F'(\mathsf{C}_{\mathsf{qry}}(Y_j))) = X_j)\}$, for F' being a uniform, and independent, element of $\mathcal{F}$.

   (In particular, $M^{i-1} \times F = V^{i-1}$ implies that Z_{i-1} holds, and binds the value of $(X_{<i}, Y_{<i})$ to V^{i-1}.)

3. The event $M^i \times F' = V^i$ is the event $\{\mathsf{C}_{\mathsf{dec}}(Y_i, F'(\mathsf{C}_{\mathsf{qry}}(Y_i))) = X_i\}$.

   (In particular, $M^i \times F = V^i$ binds the value of X_i to V^i.)

The above $\mathbf{M}$ and V are defined as follows: assume for ease of notation that C has a *linear*, and not affine, decoder. That is, for every $y \in [n]$ there exists a (q-sparse) vector $\alpha_y \in \mathbb{F}^n$ such that $\langle \alpha_y, F \rangle = X_y$. By definition, for every $j < i$:

1. $\langle \alpha_{Y_j}, F \rangle = X_j$.

Conditioning on Z_{i-1} further implies that for every $j < i$:

2. $F(X_j) = Y_j$.

Let $\ell := 2i - 2$, and let $\mathbf{M}^{i-1} \in \mathbb{F}^{\ell \times n}$ be the (random) matrix defined by $\mathbf{M}^{i-1}_{2k-1} := \alpha_{Y_k}$ and $\mathbf{M}^{i-1}_{2k} := e_{X_k}$, letting e_j being the *unit vector* $(0^{j-1}, 1, 0^{n-j})$. Let $V^{i-1} \in \mathbb{F}^{\ell}$ be the (random) vector defined by $V^{i-1}_{2k-1} := X_k$ and $V^{i-1}_{2k} = Y_k$. By definition, the event Z_{i-1} is equivalent to the event $\mathbf{M}^{i-1} \times F = V^{i-1}$. The computation C makes on input Y_i can also be described by the linear equation $\langle \alpha_{Y_i}, F \rangle = X_i$. Let $\mathbf{M} := \begin{pmatrix} \mathbf{M}^{i-1} \\ \alpha_{Y_i} \end{pmatrix}$ and $V := \begin{pmatrix} V^{i-1} \\ X_i \end{pmatrix}$. We make use of the following claims (see proofs in Section 3.2).

Definition 2.2 (Spanned unit vectors)**.** *For a matrix $\boldsymbol{A} \in \mathbb{F}^{a \times n}$, let $\mathcal{E}(\boldsymbol{A}) := \{j \in [n] \colon e_j \in \operatorname{Span}(\boldsymbol{A})\}$, for $\operatorname{Span}(\boldsymbol{A})$ being the (linear) span of $\boldsymbol{A}$'s rows.*

That is, $\mathcal{E}(\mathbf{A})$ is the set of indices of all unit vectors spanned by $\mathbf{A}$. It is clear that $|\mathcal{E}(\mathbf{A})| \leq \operatorname{rank}(\mathbf{A}) \leq \min\{a, n\}$. The following claim states that for $j \notin \mathcal{E}(\mathbf{A})$, knowing the value of $\mathbf{A} \times F$ gives no information about F_j.

Claim 2.3. *Let $\boldsymbol{A} \in \mathbb{F}^{a \times n}$ and $v \in \operatorname{Im}(\boldsymbol{A})$. Then for every $j \in [n] \setminus \mathcal{E}(\boldsymbol{A})$ and $y \in [n]$, it holds that $\Pr_{f \leftarrow [n]^n}[f_j = y \mid \boldsymbol{A} \times f = v] = 1/n$.*

The second claim roughly states that by concatenating a c-row matrix to a given matrix $\mathbf{A}$, one does not increase the spanned unit set of $\mathbf{A}$ by more than c elements.

Claim 2.4. *For every $\boldsymbol{A} \in \mathbb{F}^{\ell \times n}$ there exists an ℓ-size set $\mathcal{S}_A \subseteq [n]$ such that the following holds: for every $\boldsymbol{B} \in \mathbb{F}^{c \times n}$ there exists a c-size set $\mathcal{S}_B \subseteq [n]$ such that $\mathcal{E}\begin{pmatrix}\boldsymbol{A} \\ \boldsymbol{B}\end{pmatrix} \subseteq \mathcal{S}_A \cup \mathcal{S}_B$.*

For bounding $\Pr[Z_i \mid Z_{i-1}]$ using the above observations, we write

$$\Pr[Z_i \mid Z_{i-1}] = \Pr[Z_i \wedge X_i \in \mathcal{E}(\mathbf{M}) \mid Z_{i-1}] + \Pr[Z_i \wedge X_i \notin \mathcal{E}(\mathbf{M}) \mid Z_{i-1}] \tag{4}$$

and finish the proof by separately bounding the two terms of the above equation. Let $H := (X_i, Y_{\leq i}, \mathbf{M}, V)$. We first note that

$$\begin{aligned}
&\Pr[Z_i \wedge X_i \notin \mathcal{E}(\mathbf{M}) \mid Z_{i-1}] \leq \Pr[Z_i \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}] \\
&= \mathop{\mathrm{E}}_{(x_i, y_{\leq i}, m, v) \leftarrow H \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}} [\Pr[F(x_i) = y_i \mid m \times F = v, Y_{\leq i} = y_{\leq i}]] \\
&= \mathop{\mathrm{E}}_{(x_i, y_{\leq i}, m, v) \leftarrow H \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}} [\Pr[F(x_i) = y_i \mid m \times F = v]] \\
&= 1/n.
\end{aligned} \tag{5}$$

The first equality holds by definition of Z_{i-1}, the second equality since F is independent of Y, and the last one follows by Claim 2.3. For bounding the left-hand term of Equation (4), let $\mathcal{S}$ and T be the ℓ-size set and the index guaranteed by Claim 2.4 for the matrices $\mathbf{M}^{i-1}$ and vector α_{Y_i}, respectively. Compute,

$$\begin{aligned}
\Pr[Z_i \wedge X_i \in \mathcal{E}(\mathbf{M}) \mid Z_{i-1}] &\leq \Pr[Y_i \in F(\mathcal{E}(\mathbf{M})) \mid Z_{i-1}] \\
&\leq \Pr[Y_i \in F(\mathcal{S} \cup \{T\}) \mid Z_{i-1}] \\
&\leq \Pr[Y_i \in F(\mathcal{S}) \mid Z_{i-1}] + \Pr[Y_i = F(T) \mid Z_{i-1}].
\end{aligned} \tag{6}$$

The second inequality is by Claim 2.4. Since $F(\mathcal{S})$ is independent of Y_i, it holds that

$$\Pr[Y_i \in F(\mathcal{S}) \mid Z_{i-1}] \leq |\mathcal{S}|/n = \ell/n \tag{7}$$

Bounding $\Pr[Y_i = F(T) \mid Z_{i-1}]$ is more involved since T might depend on Y_i.[3] Yet since f is a random function, a simple counting argument yields that for any (fixed and independent of f) function g:

$$\Pr_{f \leftarrow \mathcal{F}}\left[\Pr_{y \leftarrow [n]}[y = f(g(y))] \geq 1/2\right] \leq n^{-n/3} \tag{8}$$

[3] Indeed, this dependency between the queries to f and the value to invert is exactly what makes (efficient) inversion by adaptive inverters possible.

Let $H := (X_{<i}, Y_{<i})$, and for $h = (x_{<i}, y_{<i}) \in \operatorname{Supp}(H)$ compute

$$\begin{aligned}
&\Pr_{f \leftarrow F | Z_{i-1}, H=h}\left[\Pr\left[Y_i = f(T) \mid H = h\right] \geq 1/2\right] \qquad (9)\\
&\leq \frac{1}{\Pr\left[H = h, Z_{i-1} \mid Y_{<i} = y_{<i}\right]} \cdot \Pr_{f \leftarrow F | Y_{<i} = y_{<i}}\left[\Pr\left[Y_i = F(T) \mid H = h\right] \geq 1/2\right]\\
&= \frac{1}{\Pr\left[H = h, Z_{i-1} \mid Y_{<i} = y_{<i}\right]} \cdot \Pr_{f \leftarrow F}\left[\Pr\left[Y_i = F(T) \mid H = h\right] \geq 1/2\right]\\
&\leq \frac{1}{\Pr\left[H = h, Z_{i-1} \mid Y_{<i} = y_{<i}\right]} \cdot n^{-n/3}\\
&\leq n^{n/4} \cdot n^{-n/3} \in o(1).
\end{aligned}$$

The first equality holds since F is independent of Y. The second inequality holds by Equation (8), noting that under the conditioning on $H = h$, the value of T is a deterministic function of Y_i. The third inequality holds since for not too big i, $\Pr\left[H = h, Z_{i-1} \mid Y_{<i} = y_{<i}\right] \geq n^{-n/4}$, since this probabilistic event is essentially a system of linear equations over a randomly selected vector. Since the above holds for any h, we conclude that $\Pr\left[Y_i = F(T) \mid Z_{i-1}\right] \leq 1/2 + o(1)$. Putting it all together, yields that $\Pr\left[Z_i \mid Z_{i-1}\right] < 1/n + \ell/n + 1/2 + o(1) < 3/5$, for not too large i.

2.2.1 Affine Decision Trees

Similarly to the affine decoder case, we prove the theorem by bounding $\Pr\left[Z_i \mid Z_{i-1}\right]$ for all "not too large i". Also in this case, we bound this probability by translating the conditioning on Z_{i-1} into a system of affine equations. In particular, we would like to find proper definitions for the matrix $\mathbf{M} = \begin{pmatrix} \mathbf{M}^{i-1} \\ \mathbf{M}^{i} \end{pmatrix}$ and vector $V = \begin{pmatrix} V^{i-1} \\ V^{i} \end{pmatrix}$, functions of $(X_{\leq i}, Y_{\leq i})$, such that conditions 1–3 mentioned in the affine decoder case hold.

We achieve these conditions by adding for each $j < i$ an equation for each of the linear computations done in the decision tree that computes X_j from Y_j. The price is that rather than having $\Theta(i)$ equations, we now have $\Theta(d \cdot i)$, for d being the depth of the decision tree. In order to have $\mathbf{M}^i$ a deterministic function of Y_i alone, we cannot simply make $\mathbf{M}^i$ reflect the d linear computations performed by the decoder, since each of these may depend on the results of previous computations, and thus depend on F. So rather, we have to add a row (i.e., an equation) for each of the q queries the decoder might use (queries that span all possible computations), which by definition also imply the dependency on q. Taking these additional rows into account yields the desired bound.

3 Preliminaries

3.1 Notation

All logarithms considered here are in base two. We use calligraphic letters to denote sets, uppercase for random variables and probabilistic events, lowercase for functions and fixed values, and bold uppercase for matrices. Let $[n] := \{1, \ldots, n\}$. Given a vector $v \in \Sigma^n$, let v_i denote its i^{th} entry, let $v_{<i} := v_{1,\ldots,i-1}$ and let $v_{\leq i} := v_{1,\ldots,i}$. Let $\binom{[n]}{k}$ denote the set of all subsets of $[n]$ of size k. The vector v is *q-sparse* if it has no more than q non-zero entries.

Functions. We naturally view functions from $[n]$ to $[m]$ as vectors in $[m]^n$, by letting $f_i = f(i)$. For a finite ordered set $\mathcal{S} := \{s_1, \ldots, s_k\}$, let $f(\mathcal{S}) := \{f(s_1), f(s_2), \ldots, f(s_k)\}$. Let $f^{-1}(y) := \{x \in [n] \colon f(x) = y\}$ and let $\mathrm{Im}(f) = \{f(x) \colon x \in [n]\}$. A function $f \colon \mathbb{F}^n \to \mathbb{F}$, for a field $\mathbb{F}$ and $n \in \mathbb{N}$, is *affine* if there exist a vector $v \in \mathbb{F}^n$ and a constant $\beta \in \mathbb{F}$ such that $f(x) = \langle v, x \rangle + \beta$ for every $x \in \mathbb{F}^n$, letting $\langle v, x \rangle := \sum v_i \cdot x_i$ (all operations are over $\mathbb{F}$).

Distributions and random variables. The support of a distribution P over a finite set $\mathcal{S}$ is defined by $\mathrm{Supp}(P) := \{x \in \mathcal{S} : P(x) > 0\}$. For a set $\mathcal{S}$, let $s \leftarrow \mathcal{S}$ denote that s is uniformly drawn from $\mathcal{S}$. For $\delta \in [0, 1]$, let $h(\delta) := -\delta \log \delta - (1 - \delta) \log(1 - \delta)$, i.e., the binary entropy function.

3.2 Matrices and Linear Spaces

We identify the elements of a finite field of size n with the elements of the set $[n]$, using some arbitrary, fixed, mapping. Let e_i denote the i$^{\text{th}}$ unit vector $e_j = (0^{j-1}, 1, 0^{n-j})$.

For a matrix $\mathbf{A} \in \mathbb{F}^{a \times b}$, let $\mathbf{A}_i$ denote the i$^{\text{th}}$ row of $\mathbf{A}$. The span of $\mathbf{A}$'s rows is defined by $\mathrm{Span}(\mathbf{A}) := \left\{v \in \mathbb{F}^b \colon \exists \delta_1, \ldots, \delta_a \in \mathbb{F} \colon v = \sum_{i=1}^{a} \delta_i \mathbf{A}_i\right\}$. Let $\mathrm{Im}(\mathbf{A}) = \left\{v \in \mathbb{F}^a \colon \exists w \in \mathbb{F}^b \ : \ \mathbf{A} \times w = v\right\}$, or equivalently, the image set of the function $f_{\mathbf{A}}(w) := \mathbf{A} \times w$. We use the following well-known fact:

Fact 3.1. *Let $\mathbb{F}$ be a finite field of size n, let $\boldsymbol{A} \in \mathbb{F}^{a \times b}$, let $v \in \mathrm{Im}(\boldsymbol{A})$, and let $\mathcal{F} \subseteq \mathbb{F}^b$ be the solution set of the system of equations $\boldsymbol{A} \times F = v$. Then $|\mathcal{F}| = n^{b - \mathrm{rank}(\boldsymbol{A})}$.*

We also make use of the following less standard notion.

Definition 3.2 (Spanned unit vectors)**.** *For a matrix $\boldsymbol{A} \in \mathbb{F}^{a \times b}$, let $\mathcal{E}(\boldsymbol{A}) := \{j \in [b] \colon e_j \in \mathrm{Span}(\boldsymbol{A})\}$.*

That is, $\mathcal{E}(\mathbf{A})$ is the indices of all unit vectors spanned by $\mathbf{A}$. It is clear that $|\mathcal{E}(\mathbf{A})| \leq \mathrm{rank}(\mathbf{A}) \leq \min\{a, b\}$. It is also easy to see that for any $v \in \mathrm{Im}(\mathbf{A})$, $\mathcal{E}(\mathbf{A})$ holds those entries that are *common to all solutions w of the system $\boldsymbol{A} \times w = v$.* [4] The following claim states that for $i \notin \mathcal{E}(\mathbf{A})$, the number of solutions w of the system $\mathbf{A} \times w = v$ with $w_i = y$, is the same for every y.

Claim 3.3. *Let $\mathbb{F}$ be a finite field of size n, let $\boldsymbol{A} \in \mathbb{F}^{a \times b}$ and $v \in \mathrm{Im}(\boldsymbol{A})$. Then for every $i \in [n] \setminus \mathcal{E}(\boldsymbol{A})$ and $y \in [n]$, it holds that $\Pr_{f \leftarrow [n]^b}[f_i = y \mid \boldsymbol{A} \times f = v] = 1/n$.*

Proof. Let $\mathcal{F}_{\mathbf{A},v} := \left\{f \in [n]^b \colon \mathbf{A} \times f = v\right\}$ be the set of solutions for the equation $\mathbf{A} \times F = v$. Since, by assumption, $\mathbf{A} \times F = v$ has a solution, by Fact 3.1 it holds that $|\mathcal{F}_{\mathbf{A},v}| = n^{b - \mathrm{rank}(\mathbf{A})}$. Next, let $\mathbf{A}' := \begin{pmatrix} \mathbf{A} \\ e_i \end{pmatrix}, v' := \begin{pmatrix} v \\ y \end{pmatrix}$, and $\mathcal{F}_{\mathbf{A},v,i,y} := \left\{f \in [n]^b \colon \mathbf{A}' \times f = v'\right\}$ (i.e., $\mathcal{F}_{\mathbf{A},v,i,y}$ is the set of solutions for $\mathbf{A}' \times F = v'$). Since, by assumption, $e_i \notin \mathrm{Span}(\mathbf{A})$, it holds that $\mathbf{A}' \times F = v'$ has a solution and $\left|\mathcal{F}_{\mathbf{A},v,i,y}\right| = n^{b - \mathrm{rank}(\mathbf{A}')} = n^{b - \mathrm{rank}(\mathbf{A}) - 1}$. We conclude that $\Pr_{f \leftarrow [n]^b}[f_i = y \mid \mathbf{A} \times f = v] = \frac{|\mathcal{F}_{\mathbf{A},v,i,y}|}{|\mathcal{F}_{\mathbf{A},v}|} = 1/n$. □

The following claim states that adding a small number of rows to a given matrix $\mathbf{A}$ does not increase the set $\mathcal{E}(\mathbf{A})$ by much.

[4] That is, for every $i \in \mathcal{E}(\mathbf{A})$, w_i can be described as a linear combination of the entries of v, and thus w_i is fixed by v.

Claim 3.4. *For every $\boldsymbol{A} \in \mathbb{F}^{\ell \times n}$ there exists an ℓ-size set $\mathcal{S}_{\boldsymbol{A}} \subseteq [n]$ such that the following holds: for any $\boldsymbol{B} \in \mathbb{F}^{c \times n}$ there exists a c-size set $\mathcal{S}_{\boldsymbol{B}} \subseteq [n]$ for which $\mathcal{E}\begin{pmatrix}\boldsymbol{A}\\ \boldsymbol{B}\end{pmatrix} \subseteq \mathcal{S}_{\boldsymbol{A}} \cup \mathcal{S}_{\boldsymbol{B}}$.*

Proof. Standard row operations performed on a matrix $\mathbf{M}$ do not affect $\mathrm{Span}(\mathbf{M})$, and thus do not affect $\mathcal{E}(\mathbf{M})$. Therefore, we may assume that both $\mathbf{A}$ and $\mathbf{B}$ are in row canonical form.[5] For a matrix $\mathbf{M}$ in row canonical form, let $\mathcal{L}(\mathbf{M}) := \{i \in [n] \colon \text{the i}^{\text{th}} \text{ column of } \mathbf{M} \text{ contains a leading } 1\}$. Let $\mathcal{S}_{\mathbf{A}} := \mathcal{L}(\mathbf{A})$ and note that $|\mathcal{S}_{\mathbf{A}}| = \mathrm{rank}(\mathbf{A}) \leq \ell$. Perform Gaussian elimination on $\begin{pmatrix}\mathbf{A}\\ \mathbf{B}\end{pmatrix}$ to yield a matrix $\mathbf{E}$ in row canonical form, and let $\mathcal{S}_{\mathbf{E}} := \mathcal{L}(\mathbf{E})$. Note that $\mathcal{S}_{\mathbf{A}} \subseteq \mathcal{S}_{\mathbf{E}}$, since adding rows to a matrix may only expand the set of leading ones. Furthermore, $|\mathcal{S}_{\mathbf{E}}| = \mathrm{rank}(\mathbf{E}) \leq \mathrm{rank}(\mathbf{A}) + c$. Clearly, $\mathcal{E}(\mathbf{E}) \subseteq \mathcal{S}_{\mathbf{E}}$, and we can write $\mathcal{S}_{\mathbf{E}} = \mathcal{S}_{\mathbf{A}} \cup \mathcal{S}_{\mathbf{B}}$, for $\mathcal{S}_{\mathbf{B}} := (\mathcal{S}_{\mathbf{E}} \setminus \mathcal{S}_{\mathbf{A}})$. Finally, $|\mathcal{S}_{\mathbf{B}}| = |\mathcal{S}_{\mathbf{E}}| - |\mathcal{S}_{\mathbf{A}}| \leq \mathrm{rank}(\mathbf{A}) + c - \mathrm{rank}(\mathbf{A}) = c$, and the proof follows. □

3.3 Random Functions

Let $\mathcal{F}_n$ be the set of all functions from $[n]$ to $[n]$. We make the following observations.

Claim 3.5. *Let $\mathcal{S}_1, \ldots, \mathcal{S}_n \subseteq [n]$ be c-size sets, and for $f \in \mathcal{F}_n$ let $\mathcal{K}_f := \{y \in [n] \colon y \in f(\mathcal{S}_y)\}$. Then for any $\mu \in [0, \frac{1}{2}]$:*

$$\Pr_{f \leftarrow \mathcal{F}_n}[|\mathcal{K}_f| \geq \mu n] \leq 2^{2\lceil \mu n \rceil \log(1/\mu) + \lceil \mu n \rceil \log(c/n)}.$$

Proof. For $\mathcal{T} := \left\{t_1, \ldots, t_{\lceil \mu n \rceil}\right\} \subseteq [n]$, let $\mathcal{F}_{\mathcal{T}} := \{f \in \mathcal{F}_n \colon \mathcal{T} \subseteq \mathcal{K}_f\}$. We make a rough over-counting for the size of $\mathcal{F}_{\mathcal{T}}$: one can describe $f \in \mathcal{F}_{\mathcal{T}}$ by choosing $x_i \in [n]$ for each set $\mathcal{S}_{t_i}$, and require that $f(x_i) = t_i$ (to ensure $t_i \in f(\mathcal{S}_{t_i})$). There are at most $c^{\lceil \mu n \rceil}$ ways to perform these choices. There are no constraints on the remaining $n - \lceil \mu n \rceil$ values of f. Therefore $|\mathcal{F}_{\mathcal{T}}| \leq c^{\lceil \mu n \rceil} \cdot n^{n - \lceil \mu n \rceil}$. This immediately implies that $\Pr_{f \leftarrow \mathcal{F}_n, \mathcal{T} \leftarrow \binom{[n]}{\lceil \mu n \rceil}}[\mathcal{T} \subseteq \mathcal{K}_f] \leq \left(\frac{c}{n}\right)^{\lceil \mu n \rceil}$. We conclude that

$$\begin{aligned}\Pr_{f \leftarrow \mathcal{F}_n}[|\mathcal{K}_f| \geq \mu n] &= \Pr\left[\exists \mathcal{T} \subseteq \mathcal{K}_f \colon |\mathcal{T}| = \lceil \mu n \rceil\right] \\ &\leq \sum_{\mathcal{T} \in \binom{[n]}{\lceil \mu n \rceil}} \Pr_{f \leftarrow \mathcal{F}_n}[\mathcal{T} \subseteq \mathcal{K}_f] \leq \binom{n}{\lceil \mu n \rceil} \cdot \left(\frac{c}{n}\right)^{\lceil \mu n \rceil} \leq 2^{2\lceil \mu n \rceil \log(1/\mu) + \lceil \mu n \rceil \log(c/n)}.\end{aligned}$$

The last inequality follows Facts 3.10 and 3.11, and the fact that $\log(1/\mu) \geq \log(n/\lceil \mu n \rceil)$. □

Claim 3.6. *Let $n \in \mathbb{N}$, let $F \leftarrow \mathcal{F}_n$ and let W be an event (jointly distributed with F) of probability at least p. Let $Y \leftarrow [n]$ be independent of F and W. Then for every c-size sets $\mathcal{S}_1, \ldots, \mathcal{S}_n \subseteq [n]$ and $\gamma \in [0, \frac{1}{2}]$, it holds that*

$$\Pr[Y \in F(\mathcal{S}_Y) \mid W] \leq \gamma + 2^{2\lceil \gamma n \rceil \log(1/\gamma) + \lceil \gamma n \rceil \log(c/n) + \log(1/p)}.$$

[5] (1) all zero rows are at the bottom (2) the first non-zero entry in each row is equal to 1 (known as the "leading 1") (3) the leading 1 in each row appears strictly to the right of the leading 1 in all the rows above it (4) a column that contains a leading 1 is zero in all other entries. It is a well-known that a matrix can be reduced to row canonical form using Gaussian elimination, and the set of columns containing a leading 1 is unique.

Proof. Let $\mathcal{K}_f := \{y \in [n] \colon y \in f(\mathcal{S}_y)\}$. For $\gamma \in [0,1]$, compute:

$$\begin{aligned}
&\Pr\left[Y \in F(\mathcal{S}_Y) \mid W\right] = \Pr\left[Y \in \mathcal{K}_F \mid W\right] \qquad (10)\\
&\le \Pr\left[|\mathcal{K}_F| \ge \gamma n \mid W\right] \cdot \Pr\left[Y \in \mathcal{K}_F \mid W, |\mathcal{K}_F| \ge \gamma n\right] + \Pr\left[|\mathcal{K}_F| < \gamma n \mid W\right] \cdot \Pr\left[Y \in \mathcal{K}_F \mid W, |\mathcal{K}_F| < \gamma n\right]\\
&\le \Pr\left[|\mathcal{K}_F| \ge \gamma n \mid W\right] + \gamma.
\end{aligned}$$

The last inequality holds since Y is independent of W and F. Since $\Pr[W] \ge p$, it holds that:

$$\Pr\left[|\mathcal{K}_F| \ge \gamma n \mid W\right] \le \frac{\Pr\left[|\mathcal{K}_F| \ge \gamma n\right]}{\Pr[W]} \le 2^{2\lceil \gamma n \rceil \log(1/\gamma) + \lceil \gamma n \rceil \log(c/n) + \log(1/p)} \tag{11}$$

The second inequality is by Claim 3.5. We conclude that:

$$\Pr\left[Y \in F(\mathcal{S}_Y) \mid W\right] \le \gamma + 2^{2\lceil \gamma n \rceil \log(1/\gamma) + \lceil \gamma n \rceil \log(c/n) + \log(1/p)}.$$

□

The next claim bounds the probability that a random function compresses an image set.

Claim 3.7. *For any $n \in \mathbb{N}$ and $\tau, \delta \in [0, \frac{1}{2}]$, it holds that* $\alpha_{\tau,\delta} := \Pr_{f \leftarrow \mathcal{F}_n}\left[\exists \mathcal{X} \subseteq [n] \colon\ |\mathcal{X}| \ge \tau n \wedge |f(\mathcal{X})| \le \delta n\right] \le 2^{n(h(\tau)+h(\delta)) + \lfloor \tau n \rfloor \log \delta}$.

Proof. Compute:

$$\begin{aligned}
\alpha_{\tau,\delta} &= \Pr_{f \leftarrow \mathcal{F}_n}\left[\exists \mathcal{X}, \mathcal{Y} \subseteq [n] \colon\ |\mathcal{X}| \ge \tau n \wedge |\mathcal{Y}| \le \delta n \wedge f(\mathcal{X}) \subseteq \mathcal{Y}\right]\\
&\le \Pr_{f \leftarrow \mathcal{F}_n}\left[\exists \mathcal{X}, \mathcal{Y} \subseteq [n] \colon\ |\mathcal{X}| = \lfloor \tau n \rfloor \wedge |\mathcal{Y}| = \lfloor \delta n \rfloor \wedge f(\mathcal{X}) \subseteq \mathcal{Y}\right]\\
&\le \sum_{\mathcal{Y} \in \binom{[n]}{\lfloor \delta n \rfloor}} \sum_{\mathcal{X} \in \binom{[n]}{\lfloor \tau n \rfloor}} \Pr\left[f(\mathcal{X}) \subseteq \mathcal{Y}\right] \le \binom{n}{\lfloor \delta n \rfloor}\binom{n}{\lfloor \tau n \rfloor} \cdot \delta^{\lfloor \tau n \rfloor} \le 2^{n(h(\tau)+h(\delta)) + \lfloor \tau n \rfloor \log \delta}.
\end{aligned}$$

The last inequality follows from Fact 3.11, and since h is monotone in $[0, \frac{1}{2}]$. □

The last claim states that an algorithm that inverts $f(x)$ with good probability, is likely to return x itself.

Claim 3.8. *Let C be a function from $\mathcal{F}_n \times [n]$ to $[n]$ such that $\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}}\left[\mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \ge \alpha$. Then, $\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}}\left[\mathsf{C}(f, f(x)) = x\right] \ge \frac{\alpha^2}{8}$.*

Proof. For $f \in \mathcal{F}_n$ let $\mathcal{P}_f(x) := f^{-1}(f(x)) \setminus \{x\}$. We would like to provide a bound on the size of this set to ensure that x is output with high probability. Compute

$$\begin{aligned}
\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}}\left[\mathsf{C}(f, f(x)) = x\right] &= \Pr\left[\mathsf{C}(f, f(x)) = x \wedge \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \qquad (12)\\
&\ge \Pr\left[\mathsf{C}(f, f(x)) = x \mid \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \cdot \alpha.
\end{aligned}$$

We now provide a lower bound for the left-hand term. For $d \geq 1$ compute

$$
\begin{aligned}
&\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}} \left[\mathsf{C}(f, f(x)) = x \mid \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \qquad (13) \\
&\geq \Pr\left[\mathsf{C}(f, f(x)) = x \wedge |\mathcal{P}_f(x)| \leq d \mid \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \\
&= \Pr\left[\mathsf{C}(f, f(x)) = x \mid |\mathcal{P}_f(x)| \leq d, \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \cdot \Pr\left[|\mathcal{P}_f(x)| \leq d \mid \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \\
&\geq \frac{1}{d+1} \cdot \Pr\left[|\mathcal{P}_f(x)| \leq d \mid \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \\
&= \frac{1}{d+1}\left(1 - \Pr\left[|\mathcal{P}_f(x)| > d \mid \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right]\right).
\end{aligned}
$$

By linearity of expectation, $\mathrm{E}_{f \leftarrow \mathcal{F}_n}[|\mathcal{P}_f(x)|] = \frac{n-1}{n} < 1$. Hence by Markov's inequality, $\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}}[|\mathcal{P}_f(x)| > d] < 1/d$. It follows that

$$
\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}} \left[|\mathcal{P}_f(x)| > d \mid \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \leq \frac{\Pr\left[|\mathcal{P}_f(x)| > d\right]}{\Pr\left[\mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right]} \leq \frac{1}{d\alpha} \qquad (14)
$$

Combining Equations (13) and (14) yields that

$$
\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}} \left[\mathsf{C}(f, f(x)) = x \mid \mathsf{C}(f, f(x)) \in f^{-1}(f(x))\right] \geq \frac{1}{d+1}\left(1 - \frac{1}{d\alpha}\right) \qquad (15)
$$

Finally, by Equations (12) and (15) we conclude that

$$
\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}} \left[\mathsf{C}(f, f(x)) = x\right] \geq \frac{\alpha}{d+1}\left(1 - \frac{1}{d\alpha}\right) \geq \frac{\alpha}{2d}\left(1 - \frac{1}{d\alpha}\right) = \frac{\alpha}{2d} - \frac{1}{2d^2}.
$$

Setting $d = \frac{2}{\alpha}$ yields that $\Pr_{\substack{f \leftarrow \mathcal{F}_n \\ x \leftarrow [n]}}[\mathsf{C}(f, f(x)) = x] \geq \frac{a^2}{4} - \frac{\alpha^2}{8} = \frac{\alpha^2}{8}$.

□

3.4 Additional Inequalities

We use the following easily-verifiable facts:

Fact 3.9. *For $x \geq 1$: $\log x \geq 1 - 1/x$.*

Fact 3.10. *For $\delta \leq 1/2$: $h(\delta) \leq -2\delta \log \delta$.*

We also use the following bound:

Fact 3.11 ([12])**.** $\binom{n}{k} \leq 2^{nh(\frac{k}{n})}$.

4 Linear-advice Inverters

In this section we present lower bounds on the time/memory tradeoff of adaptive function inverters with linear advice. The extension to additive-advice inverters is given in Section 4.1.

To simplify notation, the following definitions and results are stated with respect to a fixed $n \in \mathbb{N}$. Let $\mathcal{F}$ be the set of all functions from $[n]$ to $[n]$. All asymptotic notations (e.g., Θ) hide constant terms that are independent of n. We start by formally defining adaptive function inverters.

Definition 4.1 (Adaptive inverters)**.** *An s-advice, q-query adaptive inverter is a deterministic algorithm pair $\mathsf{C} := (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{dec}})$, where $\mathsf{C}_{\mathsf{pre}} : \mathcal{F} \to \{0,1\}^s$, and $\mathsf{C}_{\mathsf{dec}}^{(\cdot)} \colon [n] \times \{0,1\}^s \to [n]$ makes up to q oracle queries. For $f \in \mathcal{F}$ and $y \in [n]$, let*

$$\mathsf{C}(y; f) := \mathsf{C}_{\mathsf{dec}}^{f}(y, \mathsf{C}_{\mathsf{pre}}(f)).$$

That is, $\mathsf{C}_{\mathsf{pre}}$ is the *preprocessing* algorithm that takes as input the function description and outputs a string of length s that we refer to as the *advice* string. The oracle-aided $\mathsf{C}_{\mathsf{dec}}$ is the *decoder* algorithm that performs the actual inversion action. It receives the element to invert y and the advice string, and using q (possibly adaptive) queries to f, attempts to output a preimage of y. Finally, $\mathsf{C}(y; f)$ is the candidate preimage the algorithms of C produce for the element to invert y given the (restricted) access to f. We define adaptive inverters with linear advice as follows, recalling that we may view $f \in \mathcal{F}$ as a vector $\in [n]^n$.

Definition 4.2 (Linear preprocessing)**.** *A deterministic algorithm $\mathsf{C}_{\mathsf{pre}} \colon \mathcal{F} \to \{0,1\}^s$ is linear if there exist an additive group $\mathcal{G} \subseteq \{0,1\}^s$ that contains $\mathsf{C}_{\mathsf{pre}}(\mathcal{F})$, and an additive group $\mathcal{K}$ of size n such that for every $f_1, f_2 \in \mathcal{F}$ it holds that $\mathsf{C}_{\mathsf{pre}}(f_1 +_{\mathcal{K}} f_2) = \mathsf{C}_{\mathsf{pre}}(f_1) +_{\mathcal{G}} \mathsf{C}_{\mathsf{pre}}(f_2)$, letting $f_1 +_{\mathcal{K}} f_2 := ((f_1)_1 +_{\mathcal{K}} (f_2)_1, \ldots, (f_1)_n +_{\mathcal{K}} (f_2)_n)$.*

Below we omit the subscripts from $+_{\mathcal{G}}$ and $+_{\mathcal{K}}$ when clear from the context.

We prove the bound for inverters with linear preprocessing by presenting a reduction from the well-known *set disjointness* problem.

Definition 4.3 (Set disjointness)**.** *A protocol $\Pi = (\mathsf{A}, \mathsf{B})$ solves set disjointness with error ε over all inputs in $\mathcal{Q} \subseteq \{(\mathcal{X}, \mathcal{Y}) \colon \mathcal{X}, \mathcal{Y} \subseteq [\mathbb{N}]\}$, if for every $(\mathcal{X}, \mathcal{Y}) \in \mathcal{Q}$*

$$\Pr_{\substack{r_{\mathsf{A}} \leftarrow \{0,1\}^*, r_{\mathsf{B}} \leftarrow \{0,1\}^* \\ r_p \leftarrow \{0,1\}^*}} [(\mathsf{A}(\mathcal{X}; r_{\mathsf{A}}), \mathsf{B}(\mathcal{Y}; r_{\mathsf{B}}))(r_p) = (\delta_{\mathcal{X},\mathrm{Y}}, \delta_{\mathcal{X},\mathrm{Y}})] \geq 1 - \varepsilon$$

for $\delta_{\mathcal{X},\mathrm{Y}}$ being the indicator for $\mathcal{X} \cap \mathcal{Y} = \emptyset$.

Namely, except with probability ε over their private and public randomness, the two parties find out whether their input sets intersect. Set disjointness is known to require large communication over the following set of inputs.

Definition 4.4 (Communication complexity)**.** *The communication complexity of a protocol $\Pi = (\mathsf{A}, \mathsf{B})$, denoted $CC(\Pi)$, is the maximal number of bits the parties exchange in an execution (over all possible inputs and randomness).*

Theorem 4.5 (Hardness set disjointness, Razborov [20])**.** *Exists $\varepsilon > 0$ such that for every protocol Π that solves set disjointness over* all *inputs in $\mathcal{Q} := \{\mathcal{X}, \mathcal{Y} \subseteq [n] \colon |\mathcal{X} \cap \mathcal{Y}| \leq 1\}$ with error ε, it holds that $CC(\Pi) \geq \Omega(n)$.* [6]

Our main result is the following reduction from set disjointness to function inversion.

Theorem 4.6 (From set disjointness to function inversion)**.** *Assume exists an s-advice, q-query linear-advice inversion algorithm with $\Pr_{\substack{f \leftarrow \mathcal{F} \\ x \leftarrow [n]}} \left[\mathsf{C}(f(x); f) \in f^{-1}(f(x))\right] \geq \alpha$, and let $\mathcal{Q} := \{\mathcal{X}, \mathcal{Y} \subseteq [n] \colon |\mathcal{X} \cap \mathcal{Y}| \leq 1\}$. Then for every $\varepsilon > 0$ there exists a protocol that solves set disjointness with (one-sided) error ε and communication $O\left(\frac{\log(\varepsilon)}{\log(1-\alpha^2/8)} \cdot (s + q \log n)\right)$, on all inputs in $\mathcal{Q}$.*

Combining Theorems 4.5 and 4.6 yields the following bound on linear-advice inverters.

Corollary 4.7 (Theorem 1.2, restated)**.** *Let $\mathsf{C} = (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{dec}})$ be an s-advice q-query inversion algorithm with linear preprocessing such that $\Pr_{\substack{f \leftarrow \mathcal{F} \\ x \leftarrow [n]}} \left[\mathsf{C}(f(x); f) \in f^{-1}(f(x))\right] \geq \alpha$. Then $s + q \log n \in \Omega(\alpha^2 \cdot n)$.*

Proof of Corollary 4.7. By Theorem 4.6, the existence of an s-advice, q-query linear-advice inverter C with success probability $\geq \alpha$ implies that set disjointness can be solved over $\mathcal{Q}$, with error $\varepsilon > 0$ and communication complexity $O\left(\frac{\log(\varepsilon)}{\log(1-\alpha^2/8)} \cdot (s + q \log n)\right)$. Thus, Theorem 4.5 yields that $\frac{\log(\varepsilon)}{\log(1-\alpha^2/8)} \cdot (s + q \log n) \in \Omega(n)$. Since $\frac{\log(\varepsilon)}{\log(1-\alpha^2/8)} = \log(1/\varepsilon) \cdot \frac{1}{\log(1/(1-\alpha^2/8))}$, and since, by Fact 3.9, it holds that $\log(1/(1-\alpha^2/8)) \geq \alpha^2/8$, it follows that $s + q \log n \in \Omega(\alpha^2 \cdot n)$. □

The rest of this section is devoted to proving Theorem 4.6. Fix an s-advice, q-query inverter $\mathsf{C} = (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{dec}})$ with linear preprocessing. We use C in Protocol 4.8 to solve set disjointness. In the protocol below we identify a vector $v \in \{0,1\}^n$ with the set $\{i : v_i = 1\}$.

Protocol 4.8 ($\Pi = (\mathsf{A}, \mathsf{B})$)**.**

A*'s input:* $a \in \{0,1\}^n$.

B*'s input:* $b \in \{0,1\}^n$.

Public randomness: $d \in [n]$.

Operation:

1. B *chooses* $y \leftarrow [n]$.
2. A *constructs a function* $f_{\mathsf{A}} : [n] \to [n]$ *as follows:*
   - *for* i *such that* $a_i = 0$*, it samples* $f_{\mathsf{A}}(i + d \mod n)$ *uniformly at random.*
   - *for* i *such that* $a_i = 1$*, it sets* $f_{\mathsf{A}}(i + d \mod n) = 0$.
3. B *constructs a function* $f_{\mathsf{B}} : [n] \to [n]$ *as follows:*
   - *for* i *such that* $b_i = 0$*, it samples* $f_{\mathsf{B}}(i + d \mod n)$ *uniformly at random.*

[6] [20] proved a stronger result: there exists a distribution that fails all low communication protocols. For the sake of our argument, however, it is easier to work with the weaker statement of Theorem 4.5.

- *for i such that $b_i = 1$, it sets $f_{\mathsf{B}}(i + d \mod n) = y$.*

– *Let $f := f_{\mathsf{A}} + f_{\mathsf{B}}$.*

4. A *sends* $\mathsf{C}_{\mathsf{pre}}(f_{\mathsf{A}})$ *to* B.

5. B *sets* $c := \mathsf{C}_{\mathsf{pre}}(f_{\mathsf{A}}) +_{\mathcal{G}} \mathsf{C}_{\mathsf{pre}}(f_{\mathsf{B}}) = \mathsf{C}_{\mathsf{pre}}(f)$.

6. B *emulates* $\mathsf{C}_{\mathsf{dec}}^{f}(y, c)$*: whenever* $\mathsf{C}_{\mathsf{dec}}$ *sends a query r to f, algorithm* B *forwards it to* A*, and feeds $f_{\mathsf{A}}(r) + f_{\mathsf{B}}(r)$ back into* $\mathsf{C}_{\mathsf{dec}}$.

– *Let x be* $\mathsf{C}_{\mathsf{dec}}$*'s output in the above emulation, and let $i = x - d \mod n$.*

7. B *sends (i, b_i) to* A*. If $a_i = b_i = 1$, algorithm* A *outputs False and informs* B.

8. *Otherwise, both parties output True.*

In the following we analyze the communication complexity and success probability of Π.

Claim 4.9 (Π's communication complexity)**.** *It holds that $CC(\Pi) \leq s + 2q(\log n + 1) + \log n + 3$.*

Proof.

1. In Step 4, party A sends $\mathsf{C}_{\mathsf{pre}}(f_{\mathsf{A}})$ to B.

2. In Step 6, the parties exchange at most $2 \log n + 2$ bits for every query $\mathsf{C}_{\mathsf{dec}}$ makes.

3. In Step 7, the parties exchange at most $\log n + 3$ bits.

Thus, the total communication is bounded by $s + 2q(\log n + 1) + \log n + 3$. □

Claim 4.10 (Π's success probability)**.**

1. $\Pr[(\mathsf{A}(a), \mathsf{B}(b)) = (\mathit{True}, \mathit{True})] = 1$ *for every $(a, b) \in \mathcal{Q}^0 := \{\mathcal{X}, \mathcal{Y} \subseteq [n] \colon |\mathcal{X} \cap \mathcal{Y}| = 0\}$.*

2. $\Pr[(\mathsf{A}(a), \mathsf{B}(b)) = (\mathit{False}, \mathit{False})] \geq \frac{\alpha^2}{8}$ *for every $(a, b) \in \mathcal{Q}^1 := \{\mathcal{X}, \mathcal{Y} \subseteq [n] \colon |\mathcal{X} \cap \mathcal{Y}| = 1\}$.*

Proof. By construction, it is clear that Π always accepts (the parties output True) on inputs $(a, b) \in \mathcal{Q}^0$. Fix $(a, b) \in \mathcal{Q}^1$, and let $Y, D, F, F_{\mathsf{A}}, F_{\mathsf{B}}$ and I be the values of $y, d, f, f_{\mathsf{A}}, f_{\mathsf{B}}$ and i respectively, in a random execution of $(\mathsf{A}(a), \mathsf{B}(b))$. By construction, $F(j) = F_{\mathsf{A}}(j) + F_{\mathsf{B}}(j)$ for all $j \in [n]$. For j not in the intersection, either $F_{\mathsf{A}}(j)$ or $F_{\mathsf{B}}(j)$ is chosen uniformly at random by one of the parties, and therefore $F(j)$ is uniformly distributed and independent of all other outputs. For the intersection element w, it holds that $F(w) = y$, which is uniform, and since there is exactly one intersection, is independent from all other outputs.

Let $W := w + D \mod n$. Note that W is uniformly distributed over $[n]$ and is independent of F. Also note that, by construction, $Y = F(W)$. Therefore, (F, W, Y) is distributed exactly as $(F, X, F(X))$ for $X \leftarrow [n]$. Hence, the assumption on C yields that

$$\Pr\left[\mathsf{C}(Y; F) \in F^{-1}(Y)\right] \geq \alpha$$

and by Claim 3.8,

$$\Pr\left[\mathsf{C}(Y; F) = W\right] \geq \alpha^2/8.$$

Therefore, both parties output False with probability at least $\alpha^2/8$. □

Proving Theorem 4.6 We now use Claims 4.9 and 4.10 to prove Theorem 4.6.

Proof of Theorem 4.6. Let $t = \left\lceil \frac{\log(\varepsilon)}{\log(1-\alpha^2/8)} \right\rceil$, and consider the protocol Π^t, in which on input (a, b) the parties interact in protocol Π for t times, and accept only if they do so in *all* iterations. By Claims 4.9 and 4.10, the communication complexity and success probability of Π^t in solving set disjointness over $\mathcal{Q}$ match the theorem statement. □

4.1 Additive-advice Inverters

The following result generalizes Corollary 4.7 by replacing the restriction on the decoder (e.g., linear and short output) with the ability to compute the advice string of $f_1 + f_2$ by a low-communication protocol over the inputs f_1 and f_2.

Theorem 4.11 (Bound on additive-advice inverters)**.** *Let $\mathsf{C} = (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{dec}})$ be an q-query inversion algorithm such that $\Pr_{\substack{f \leftarrow \mathcal{F} \\ x \leftarrow [n]}} \left[\mathsf{C}(f(x); f) \in f^{-1}(f(x))\right] \geq \alpha$. Assume exists a two-party protocol $(\mathsf{P}_1, \mathsf{P}_2)$ with communication complexity k such that for every $f_1, f_2 \in \mathcal{F}$, the output of P_2 in $(\mathsf{P}_1(f_1), \mathsf{P}_2(f_2))$ equals to $\mathsf{C}_{\mathsf{pre}}(f_1 + f_2)$ with probability at least $1 - \gamma$ for some $\gamma \geq 0$, letting $f_1 + f_2$ be according to Definition 4.2. Then $k + q \log n \in \Omega(\alpha^2(1-\gamma) \cdot n)$.*

Proof. The proof follows almost the exact same lines as that of Theorem 4.6, with the following changes: first, steps 4. and 5. in Protocol 4.8 are replaced by the parties A and B interacting in $(\mathsf{P}_1(f_{\mathsf{A}}), \mathsf{P}_2(f_{\mathsf{B}}))$, resulting in B outputting $\mathsf{C}_{\mathsf{pre}}(f_{\mathsf{A}} + f_{\mathsf{B}})$ (thus, transmitting a total of $k + 2q(\log n + 1) + \log n + 3 \in O(k + q \log n)$ bits over the entire execution of the protocol). Second, note that due to the constant failure probability of $(\mathsf{P}_1, \mathsf{P}_2)$ in computing $\mathsf{C}_{\mathsf{pre}}(f_{\mathsf{A}} + f_{\mathsf{B}})$, the success probability of each execution of the protocol is now lowered by a constant factor $(1 - \gamma)$. This means that the rate of success when $\mathcal{X} \cap \mathcal{Y} \neq \emptyset$ is now bounded from below by only $\alpha^2(1-\gamma)/8$ (rather than $\alpha^2/8$). The rest of the analysis is identical to that of Theorem 4.6.

□

5 Non-adaptive Inverters

In this section we present lower bounds on the time/memory tradeoff of non-adaptive function inverters. In Section 5.1, we present a bound for non-adaptive affine decoders, and in Section 5.2 we extend this bound to non-adaptive affine decision trees. To simplify notation, the following definitions and results are stated with respect to some fixed $n \in \mathbb{N}$, for which there exists a finite field of size n which we denote by $\mathbb{F}$. Let $\mathcal{F}$ be the set of all functions from $[n]$ to $[n]$. All asymptotic notations (e.g., Θ) hide constant terms that are independent of n. We start by formally defining non-adaptive function inverters.

Definition 5.1 (Non-adaptive inverters)**.** *An s-advice q-query non-adaptive inverter is a deterministic algorithm triplet of the form $\mathsf{C} := (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{qry}}, \mathsf{C}_{\mathsf{dec}})$, where $\mathsf{C}_{\mathsf{pre}} \colon \mathcal{F} \to \{0,1\}^s$, $\mathsf{C}_{\mathsf{qry}} \colon [n] \times \{0,1\}^s \to [n]^q$, and $\mathsf{C}_{\mathsf{dec}} \colon [n] \times \{0,1\}^s \times [n]^q \to [n]$. For $f \in \mathcal{F}$ and $y \in [n]$, let*

$$\mathsf{C}(y; f) := \mathsf{C}_{\mathsf{dec}}\left(y, \mathsf{C}_{\mathsf{pre}}(f), f\left(\mathsf{C}_{\mathsf{qry}}(y, \mathsf{C}_{\mathsf{pre}}(f))\right)\right).$$

That is, $\mathsf{C}_{\mathsf{pre}}$ is the *preprocessing* algorithm. It takes the function description as input and outputs a string of length s, to which we refer as the *advice* string. In the case that $s = 0$, we say that C has *zero-advice*, and omit $\mathsf{C}_{\mathsf{pre}}$ from the notation. Algorithm $\mathsf{C}_{\mathsf{qry}}$ is the *query selection* algorithm. It chooses the queries according to the element to invert y and the advice string, and outputs q indices, to which we refer as the *queries*. Algorithm $\mathsf{C}_{\mathsf{dec}}$ is the *decoder* algorithm that performs the actual inversion. It receives the element y, the advice string and the function's answers to the (non-adaptive) queries selected by $\mathsf{C}_{\mathsf{qry}}$ (the query indices themselves may be deduced from y and the advice), and attempts to output a preimage of y. Finally, $\mathsf{C}(y; f)$ is the candidate preimage of y produced by the algorithms of C given the (restricted) access to f.

5.1 Affine Decoders

In this section we present our bound for non-adaptive affine decoders, defined as follows:

Definition 5.2 (Affine decoder)**.** *A non-adaptive inverter* $\mathsf{C} := (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{qry}}, \mathsf{C}_{\mathsf{dec}})$ *has an* affine decoder*, if for every* $y \in [n]$ *and* $a \in \{0,1\}^s$ *there exists a q-sparse vector* $\alpha_y^a \in \mathbb{F}^n$ *and a field element* $\beta_y^a \in \mathbb{F}$*, such that for every* $f \in \mathcal{F}$*:* $\mathsf{C}_{\mathsf{dec}}(y, a, f(\mathsf{C}_{\mathsf{qry}}(y,a))) = \langle \alpha_y^a, f \rangle + \beta_y^a$.

The following theorem bounds the probability, over a random function f, that a non-adaptive inverter with an affine decoder inverts a random output of f with probability τ.

Theorem 5.3. *Let* $\mathsf{C} = (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{qry}}, \mathsf{C}_{\mathsf{dec}})$ *be an s-advice non-adaptive inverter with an affine decoder and let* $\tau \in [0,1]$*. Then for every* $\delta \in [0,1]$ *and* $m \leq n/16$*, it holds that*

$$\Pr_{f \leftarrow \mathcal{F}}\left[\Pr_{\substack{x \leftarrow [n] \\ y = f(x)}}\left[\mathsf{C}(y;f) \in f^{-1}(y)\right] \geq \tau\right] \leq \alpha_{\tau,\delta} + 2^s \cdot \delta^{-m} \cdot \prod_{j=1}^{m}\left(\frac{2j}{n} + \max\left\{\sqrt[4]{1/n}, \frac{4j}{n}\right\}\right)$$

for $\alpha_{\tau,\delta} := \Pr_{f \leftarrow \mathcal{F}}[\exists \tau n\text{-size set } \mathcal{X} \subset [n] \colon |f(\mathcal{X})| \leq \delta n]$.

While it is not easy to see what is the best choice, per τ, of the parameters δ and m above, the following corollary (proven in Section 5.1.2) exemplifies the usability of Theorem 5.3 by considering the consequences of such a choice.

Corollary 5.4 (Theorem 1.5, restated)**.** *Let* C *be as in Theorem 5.3, let* $\tau \geq 2 \cdot n^{-1/8}$ *and assume* $\Pr_{f \leftarrow \mathcal{F}}\left[\Pr_{\substack{x \leftarrow [n] \\ y = f(x)}}\left[\mathsf{C}(y;f) \in f^{-1}(y)\right] \geq \tau\right] \geq 1/2$*, then* $s \in \Omega(\tau^2 \cdot n)$*.* [7]

Our key step towards proving Theorem 5.3 is showing that even when conditioned on the (unlikely) event that a zero-advice inverter successfully inverts $i-1$ random elements, the probability the inverter successfully inverts the next element is still low. To formulate the above statement, we define the following jointly distributed random variables: let F be uniformly distributed over $\mathcal{F}$ and let $Y = (Y_1, ..., Y_n)$ be a uniform vector over $[n]^n$. For a zero-advice inverter, we define the following random variables (jointly distributed with F and Y).

Notation 5.5. *For a zero-advice inverter* D*, let* $X_i^{\mathsf{D}} := \mathsf{D}(Y_i; F)$*, let* Z_i^{D} *be the event* $\bigwedge_{j \in [i]}\left\{F(X_j^{\mathsf{D}}) = Y_j\right\}$*, and let* $X^{\mathsf{D}} = (X_1^{\mathsf{D}}, \ldots, X_n^{\mathsf{D}})$*.*

[7] The constant $1/2$ lower bounding the probability is arbitrary.

That is, X_i^{D} is D's answers to the challenges Y_i, and Z_i^{D} indicates whether D successfully answered each of the first i challenges. Given the above notation, our main lemma is stated as follows:

Lemma 5.6. *Let D be a zero-advice, non-adaptive inverter with affine decoder and let Z^{D} be as in Notation 5.5. Then for every $i \in [n]$ and $\mu \in [0, \frac{1}{2}]$:*

$$\Pr\left[Z_i^{\mathsf{D}} \mid Z_{i-1}^{\mathsf{D}}\right] \leq \frac{2i-1}{n} + \mu + 2^{2\lceil \mu n\rceil \log(1/\mu) - \lceil \mu n\rceil \log(n) + (2i-2)\log n}.$$

We prove Lemma 5.6 below, but first use it to prove Theorem 5.3.

Proving Theorem 5.3. Lemma 5.6 immediately yields a bound on the probability that D, *a zero-advice inverter, successfully inverts the first i elements of Y*. For proving Theorem 5.3, however, we need to bound the probability that D, and later on, an inverter with non-zero advice, finds a preimage of a *random output* of f. Yet, the conversion between these two measurements is rather straightforward. Hereafter we assume $n \geq 16$, as otherwise Theorem 5.3 is trivial, as $m = 0$.

Proof of Theorem 5.3. Let $\mathsf{C} = (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{qry}}, \mathsf{C}_{\mathsf{dec}})$, $\tau \in [0,1]$, $\delta \in [0,1]$ and m be as in the theorem statement. Fix an advice string $a \in \{0,1\}^s$, and let $\mathsf{C}^a = (\mathsf{C}^a_{\mathsf{qry}}, \mathsf{C}^a_{\mathsf{dec}})$ denote the *zero-advice* inverter obtained by hardcoding a as the advice of C (i.e., $\mathsf{C}^a_{\mathsf{pre}}(f) = a$ for every f). For $j \in [n]$, let $Z_j = Z_j^{\mathsf{C}^a}$ and let $\mu_j := \max\left\{\sqrt[4]{1/n}, \frac{4j}{n}\right\}$. We start by showing that for every $j \leq n/16$ it holds that

$$\Pr\left[Z_j \mid Z_{j-1}\right] \leq \frac{2j}{n} + \mu_j \tag{16}$$

Indeed, by Lemma 5.6

$$\Pr\left[Z_j \mid Z_{j-1}\right] \leq \frac{2j-1}{n} + \mu_j + 2^{\overbrace{2\lceil \mu_j n\rceil \log(1/\mu_j) - \lceil \mu_j n\rceil \log n + (2j-2)\log n}^{\beta}} \tag{17}$$

We write,

$$\beta = \underbrace{2\lceil \mu_j n\rceil \log(1/\mu_j) - \frac{\lceil \mu_j n\rceil}{2}\log n}_{\beta_1} + \underbrace{\left(-\frac{\lceil \mu_j n\rceil}{2}\right)\log n + (2j-2)\log n}_{\beta_2} \tag{18}$$

Since

$$\beta_1 \leq \lceil \mu_j n\rceil \left(\log\frac{1}{\mu_j^2} - \log\sqrt{n}\right) = \lceil \mu_j n\rceil \left(\log\frac{1}{\mu_j^2\sqrt{n}}\right) \leq 0$$

and

$$\beta_2 = \frac{-\lceil \mu_j n\rceil}{2}\log n + 2j\log n - 2\log n \leq \frac{-2j}{n} n\log n + 2j\log n - 2\log n \leq -2\log n,$$

we conclude that $\Pr\left[Z_j \mid Z_{j-1}\right] \leq \frac{2j-1}{n} + \mu_j + 2^{-2\log n} \leq \frac{2j}{n} + \mu_j$, proving Equation (16).

Equation (16) immediately yields that

$$\Pr\left[Z_m\right] = \prod_{j=1}^{m} \Pr\left[Z_j \mid Z_{j-1}\right] \leq \prod_{j=1}^{m} \left(\frac{2j}{n} + \mu_j\right) \tag{19}$$

We use the above to produce a bound on the number of elements that C^a successfully inverts. Let $\mathcal{G}^a_{\mathcal{Y}}(f) := \left\{y \in [n] \colon \mathsf{C}^a(y; f) \in f^{-1}(y)\right\}$, and compute:

$$\begin{aligned}
\Pr\left[Z_m\right] &= \Pr_{f \leftarrow \mathcal{F}}\left[\forall j \in [m] \colon Y_j \in \mathcal{G}^a_{\mathcal{Y}}(f)\right] && (20)\\
&\geq \Pr_{f \leftarrow \mathcal{F}}\left[\forall j \in [m] \colon Y_j \in \mathcal{G}^a_{\mathcal{Y}}(f) \bigwedge |\mathcal{G}^a_{\mathcal{Y}}(f)| \geq \delta n\right]\\
&= \Pr_{f \leftarrow \mathcal{F}}\left[\forall j \in [m] \colon Y_j \in \mathcal{G}^a_{\mathcal{Y}}(f) \mid |\mathcal{G}^a_{\mathcal{Y}}(f)| \geq \delta n\right] \cdot \Pr_{f \leftarrow \mathcal{F}}\left[|\mathcal{G}^a_{\mathcal{Y}}(f)| \geq \delta n\right]\\
&\geq \delta^m \cdot \Pr_{f \leftarrow \mathcal{F}}\left[|\mathcal{G}^a_{\mathcal{Y}}(f)| \geq \delta n\right].
\end{aligned}$$

Combining Equations (19) and (20) yields the following bound on the number of images C^a successfully inverts:

$$\Pr\left[|\mathcal{G}^a_{\mathcal{Y}}(f)| \geq \delta n\right] \leq \delta^{-m} \cdot \prod_{j=1}^{m} \left(\frac{2j}{n} + \mu_j\right) \tag{21}$$

We now adapt the above bound to (the non zero-advice) C. Let $\mathcal{G}_{\mathcal{Y}}(f) := \left\{y \in [n] \colon \mathsf{C}(y; f) \in f^{-1}(y)\right\}$ and let $\mathcal{G}_{\mathcal{X}}(f) = f^{-1}(\mathcal{G}_{\mathcal{Y}}(f))$. By Equation (21) and a union bound,

$$\Pr_{f \leftarrow \mathcal{F}}\left[|\mathcal{G}_{\mathcal{Y}}(f)| \geq \delta n\right] \leq 2^s \cdot \delta^{-m} \cdot \prod_{j=1}^{m} \left(\frac{2j}{n} + \mu_j\right) \tag{22}$$

We conclude that

$$\begin{aligned}
&\Pr_{f \leftarrow \mathcal{F}}\left[\Pr_{\substack{x \leftarrow [n]\\ y = f(x)}}\left[\mathsf{C}(y; f) \in f^{-1}(y)\right] \geq \tau\right] = \Pr_{f \leftarrow \mathcal{F}}\left[|\mathcal{G}_{\mathcal{X}}(f)| \geq \tau n\right]\\
&= \Pr_{f \leftarrow \mathcal{F}}\left[|\mathcal{G}_{\mathcal{X}}(f)| \geq \tau n \bigwedge |\mathcal{G}_{\mathcal{Y}}(f)| < \delta n\right] + \Pr_{f \leftarrow \mathcal{F}}\left[|\mathcal{G}_{\mathcal{X}}(f)| \geq \tau n \bigwedge |\mathcal{G}_{\mathcal{Y}}(f)| \geq \delta n\right]\\
&\leq \Pr_{f \leftarrow \mathcal{F}}\left[|\mathcal{G}_{\mathcal{X}}(f)| \geq \tau n \bigwedge |\mathcal{G}_{\mathcal{Y}}(f)| < \delta n\right] + \Pr_{f \leftarrow \mathcal{F}}\left[|\mathcal{G}_{\mathcal{Y}}(f)| \geq \delta n\right]\\
&\leq \alpha_{\tau,\delta} + 2^s \cdot \delta^{-m} \cdot \prod_{j=1}^{m} \left(\frac{2j}{n} + \mu_j\right).
\end{aligned}$$

The second inequality follows by the definition of $\alpha_{\tau,\delta}$ and Equation (22). □

5.1.1 Proving Lemma 5.6

In the rest of this section we prove Lemma 5.6. Fix a zero-advice non-adaptive inverter with an affine decoder $\mathsf{D} = (\mathsf{D}_{\mathsf{qry}}, \mathsf{D}_{\mathsf{dec}})$, $i \in [n]$ and $\mu \in [0, \frac{1}{2}]$. Let $X := X^{\mathsf{D}}$ and, for $j \in [n]$ let $Z_j := Z^{\mathsf{D}}_j$. We start by proving the following claim that bounds the probability in hand, assuming X_i, the inverter's answer, is coming from a small linear space. (Recall, from Definition 3.2, that $\mathcal{E}(\mathbf{M}) = \{j \in [m] \colon e_j \in \operatorname{Span}(\mathbf{M})\}$, where e_j is the j^{th} unit vector in $\mathbb{F}^n$.)

Claim 5.7. *Let $\boldsymbol{A} \in \mathbb{F}^{\ell \times n}$, let $v \in \operatorname{Im}(\boldsymbol{A})$, let $\boldsymbol{B}^1, \ldots, \boldsymbol{B}^n \in \mathbb{F}^{t \times n}$, and, for $y \in [n]$, let $\boldsymbol{A}^y := \begin{pmatrix} \boldsymbol{A} \\ \boldsymbol{B}^y \end{pmatrix}$. Then*

$$\Pr\left[Y_i \in F(\mathcal{E}(\boldsymbol{A}^{Y_i})) \mid \boldsymbol{A} \times F = v\right] \leq \left(\frac{\ell}{n} + \mu\right) + 2^{2\lceil \mu n \rceil \log(1/\mu) + \lceil \mu n \rceil \log(t/n) + \ell \log n}.$$

Proof. By Claim 3.4 there exist an ℓ-size set $\mathcal{S} := \mathcal{S}_{\mathbf{A}}$ and t-size sets $\left\{\mathcal{S}_k := \mathcal{S}_{\mathbf{B}^k}\right\}_{k \in [n]}$ such that

$$\mathcal{E}(\mathbf{A}^y) \subseteq \mathcal{S} \cup \mathcal{S}_y \tag{23}$$

for every $y \in [n]$. By Fact 3.1,

$$\Pr\left[\mathbf{A} \times F = v\right] = \frac{n^{n - \operatorname{rank}(\mathbf{A})}}{n^n} \geq n^{-\ell} \tag{24}$$

Compute,

$$\begin{aligned}
\Pr\left[Y_i \in F(\mathcal{E}(\mathbf{A}^{Y_i})) \mid \mathbf{A} \times F = v\right] &\leq \Pr\left[Y_i \in F(\mathcal{S} \cup \mathcal{S}_{Y_i}) \mid \mathbf{A} \times F = v\right] &(25)\\
&\leq \Pr\left[Y_i \in F(\mathcal{S}) \mid \mathbf{A} \times F = v\right] + \Pr\left[Y_i \in F(\mathcal{S}_{Y_i}) \mid \mathbf{A} \times F = v\right]\\
&\leq \frac{\ell}{n} + \Pr\left[Y_i \in F(\mathcal{S}_{Y_i}) \mid \mathbf{A} \times F = v\right].
\end{aligned}$$

The first inequality holds since $\mathcal{E}(\mathbf{A}^{Y_i}) \subseteq \mathcal{S} \cup \mathcal{S}_{Y_i}$, and the last one since $|\mathcal{S}| \leq \ell$ and Y_i is independent of F. Applying Claim 3.6 with respect to $p := n^{-\ell}$, $\gamma := \mu$, $W := \{\mathbf{A} \times F = v\}$, $Y := Y_i$ and the sets $\mathcal{S}_1, \ldots \mathcal{S}_n$, yields that

$$\Pr\left[Y_i \in F(\mathcal{S}_{Y_i}) \mid \mathbf{A} \times F = v\right] \leq \mu + 2^{2\lceil \mu n \rceil \log(1/\mu) + \lceil \mu n \rceil \log(t/n) + \ell \log n} \tag{26}$$

We conclude that $\Pr\left[Y_i \in F(\mathcal{E}(\mathbf{A}(Y_i))) \mid \mathbf{A} \times F = v\right] \leq \frac{\ell}{n} + \mu + 2^{2\lceil \mu n \rceil \log(1/\mu) + \lceil \mu n \rceil \log(t/n) + \ell \log n}$. □

Given the above claim, we prove Lemma 5.6 as follows.

Proof of Lemma 5.6. Since D has an affine decoder, for every $y \in [n]$ and $X := \mathsf{D}(y; F)$ there exist a q-sparse vector $\alpha^y \in \mathbb{F}^n$ and a field element $\beta^y \in \mathbb{F}$ such that $\langle \alpha^y, F \rangle + \beta^y = X$. Therefore, for every $j < i$:

1. $\langle \alpha^{Y_j}, F \rangle = -\beta^{Y_j} + X_j$.

Conditioning on Z_{i-1} further implies that for every $j < i$:

2. $F(X_j) = Y_j$.

Let $\ell := 2i - 2$, and let $\mathbf{M}^{i-1} \in \mathbb{F}^{\ell \times n}$ be the (random) matrix defined, for every $j \in [i-1]$, by $\mathbf{M}^{i-1}_{2j-1} := \alpha^{Y_j}$ and $\mathbf{M}^{i-1}_{2j} := e_{X_j}$. Let $V^{i-1} \in \mathbb{F}^{\ell}$ be the (random) vector defined by $V^{i-1}_{2j-1} := -\beta^{Y_j} + X_j$ and $V^{i-1}_{2j} = Y_j$. By definition, conditioned on Z_{i-1} it holds that $\mathbf{M}^{i-1} \times F = V^{i-1}$. This incorporates in a single equation all that is known about F given Z_{i-1}. To take into account the knowledge gained from the queries made while attempting to invert Y_i, we combine the above with α^{Y_i} and $\langle \alpha^{Y_i}, F\rangle$, into the matrix $\mathbf{M} := \begin{pmatrix} \mathbf{M}^{i-1} \\ \alpha^{Y_i} \end{pmatrix}$ and vector $V := \begin{pmatrix} V^{i-1} \\ \langle \alpha^{Y_i}, F\rangle \end{pmatrix}$. By definition, $\mathbf{M} \times F = V$. We write

$$\Pr\left[Z_i \mid Z_{i-1}\right] = \Pr\left[Z_i \wedge X_i \in \mathcal{E}(\mathbf{M}) \mid Z_{i-1}\right] + \Pr\left[Z_i \wedge X_i \notin \mathcal{E}(\mathbf{M}) \mid Z_{i-1}\right] \tag{27}$$

and prove the lemma by separately bounding the two terms of the above equation. Let $H := (Y_{<i}, \mathbf{M}^{i-1}, V^{i-1})$, and note that

$$\begin{aligned}
&\Pr\left[Z_i \wedge X_i \in \mathcal{E}(\mathbf{M}) \mid Z_{i-1}\right] \le \Pr\left[Y_i \in F(\mathcal{E}(\mathbf{M})) \mid Z_{i-1}\right] \qquad (28)\\
&= \mathop{\mathrm{E}}_{h \leftarrow H|Z_{i-1}}\left[\Pr\left[Y_i \in F(\mathcal{E}(\mathbf{M})) \mid H = h, Z_{i-1}\right]\right]\\
&= \mathop{\mathrm{E}}_{h=(y_{<i}, m^{i-1}, v^{i-1}) \leftarrow H|Z_{i-1}}\left[\Pr\left[Y_i \in F\left(\mathcal{E}\begin{pmatrix} m^{i-1} \\ \alpha^{Y_i}\end{pmatrix}\right) \mid H = h, m^{i-1} \times F = v^{i-1}\right]\right]\\
&= \mathop{\mathrm{E}}_{(y_{<i}, m^{i-1}, v^{i-1}) \leftarrow H|Z_{i-1}}\left[\Pr\left[Y_i \in F\left(\mathcal{E}\begin{pmatrix} m^{i-1} \\ \alpha^{Y_i}\end{pmatrix}\right) \mid Y_{<i} = y_{<i}, m^{i-1} \times F = v^{i-1}\right]\right]\\
&= \mathop{\mathrm{E}}_{(y_{<i}, m^{i-1}, v^{i-1}) \leftarrow H|Z_{i-1}}\left[\Pr\left[Y_i \in F\left(\mathcal{E}\begin{pmatrix} m^{i-1} \\ \alpha^{Y_i}\end{pmatrix}\right) \mid m^{i-1} \times F = v^{i-1}\right]\right]\\
&\le \left(\frac{2i-2}{n} + \mu\right) + 2^{2\lceil \mu n\rceil \log(1/\mu) + \lceil \mu n\rceil \log(1/n) + (2i-2)\log n}.
\end{aligned}$$

The first inequality holds by the definition of Z_i. The second equality holds by the definition of Z_{i-1}. The third equality holds since the event $\left\{Y_{<i} = y_{<i}, m^{i-1} \times F = v^{i-1}\right\}$ implies that $\left\{\mathbf{M}^{i-1} = m^{i-1}, V^{i-1} = v^{i-1}\right\}$. The last equality holds since F is independent of Y, and the last inequality follows by Claim 5.7 with respect to $\mathbf{A} := m^{i-1}, v := v^{i-1}$, and $(\mathbf{B}^1, \ldots, \mathbf{B}^n) := (\alpha^1, \ldots, \alpha^n)$ (viewing α^i as a matrix in $\mathbb{F}^{1\times n}$).

For bounding the right-hand term of Equation (27), let $H := (X_i, Y_{\le i}, \mathbf{M}, V)$, and compute

$$\begin{aligned}
&\Pr\left[Z_i \wedge X_i \notin \mathcal{E}(\mathbf{M}) \mid Z_{i-1}\right] \le \Pr\left[Z_i \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}\right] \qquad (29)\\
&= \mathop{\mathrm{E}}_{h \leftarrow H|X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}}\left[\Pr\left[Z_i \mid H = h, Z_{i-1}\right]\right]\\
&= \mathop{\mathrm{E}}_{h=(x_i, y_{\le i}, m, v) \leftarrow H|X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}}\left[\Pr\left[F(x_i) = y_i \mid H = h, m \times F = v\right]\right]\\
&= \mathop{\mathrm{E}}_{(x_i, y_{\le i}, m, v) \leftarrow H|X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}}\left[\Pr\left[F(x_i) = y_i \mid Y_{\le i} = y_{\le i}, m \times F = v\right]\right]\\
&= \mathop{\mathrm{E}}_{(x_i, y_{\le i}, m, v) \leftarrow H|X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}}\left[\Pr\left[F(x_i) = y_i \mid m \times F = v\right]\right]\\
&= 1/n.
\end{aligned}$$

The second equality holds by the definition of Z_{i-1}. The third equality holds since the event $\{Y_{\le i} = y_{\le i}, m \times F = v\}$ implies that $\{\mathbf{M} = m, V = v\}$, and X_i is a function of V. The fourth equality holds since F is independent from Y. The last inequality follows by Claim 3.3. Combining Equations (27) to (29), we conclude that

$$\begin{aligned}
\Pr\left[Z_i \mid Z_{i-1}\right] &\le \left(\frac{2i-2}{n} + \mu\right) + 2^{2\lceil \mu n\rceil \log(1/\mu) + \lceil \mu n\rceil \log(1/n) + (2i-2)\log n} + 1/n\\
&= \frac{2i-1}{n} + \mu + 2^{2\lceil \mu n\rceil \log(1/\mu) - \lceil \mu n\rceil \log(n) + (2i-2)\log n}.
\end{aligned}$$

□

5.1.2 Proving Corollary 5.4

Proof of Corollary 5.4. We prove for $\tau \leq 0.16$, which clearly yields the same bound for larger values of τ. Let $\delta := \tau^2$, and let $\alpha_{\tau,\delta}$ be as in Theorem 5.3. By Claim 3.7,

$$\alpha_{\tau,\tau^2} \leq 2^{n(h(\tau)+h(\tau^2))+\lfloor \tau n \rfloor \log \tau^2} \leq 2^{n(h(\tau)+h(\tau^2)+\tau \log \tau^2)-\log \tau^2} \tag{30}$$

$$\leq 2^{n\underbrace{(h(\tau)+h(\tau^2)+\tau\log\tau^2)}_{\beta}} \cdot \tau^{-2} \tag{31}$$

Since $\tau \leq 0.16$, it holds that $h(\tau) \leq \frac{3}{2} \cdot \tau \cdot \log(1/\tau)$. We also note that

$$\beta \leq \frac{3}{2} \cdot \tau \cdot \log \frac{1}{\tau} + \frac{6}{2} \cdot \tau^2 \cdot \log \frac{1}{\tau} + 2\tau \log \tau = (3\tau - 1/2) \cdot \tau \cdot \log \frac{1}{\tau} \leq \frac{-\log n}{200\sqrt[8]{n}} \tag{32}$$

The last inequality holds since, by assumption, $0.16 \geq \tau \geq \frac{2}{\sqrt[8]{n}}$, noting that $\tau \cdot \log 1/\tau$ is monotonically increasing over $[0, 0.16]$. Given the above bound on α_{τ,τ^2} and the assumption on C's success probability, Theorem 5.3 yields that for every $m \leq n/16$:

$$1/2 \leq 2^{-(n^{7/8}\log n)/200} \cdot \tau^{-2} + 2^s \delta^{-m} \prod_{j=1}^{m} \left(\frac{2j}{n} + \max\left\{ \sqrt[4]{1/n}, \frac{4j}{n} \right\} \right) \tag{33}$$

Let $m := \delta n/16$. Since $\tau \geq \frac{2}{\sqrt[8]{n}}$, for every $j \in [m]$ it holds that $\frac{2j}{n} + \max\left\{ \sqrt[4]{1/n}, \frac{4j}{n} \right\} \leq \delta/2$. Thus, by Equation (33),

$$1/2 \leq 2^{-(n^{7/8}/200-1)\log n} + 2^s \cdot \delta^{-m} \cdot (\delta/2)^m \leq 2^{-\log n} + 2^{s-m} \tag{34}$$

We conclude that $s \in \Omega(m)$ and thus $s \in \Omega(\tau^2 \cdot n)$. □

5.2 Affine Decision Trees

In this section we present lower bounds for non-adaptive affine decision trees. These are formally defined as follows:

Definition 5.8 (Affine decision trees)**.** *An n-input* affine decision tree *over $\mathbb{F}$ is a labeled, directed, degree $|\mathbb{F}|$ tree $\mathcal{T}$. Each internal node v of $\mathcal{T}$ has label $\alpha_v \in \mathbb{F}^n$, each leaf ℓ of $\mathcal{T}$ has label $o_\ell \in \mathbb{F}$, and the $|\mathbb{F}|$ outgoing edges of every internal node are labeled by the elements of $\mathbb{F}$. Let $\Gamma_{\mathcal{T}}(v,\gamma)$ denote the (direct) child of v connected via the edge labeled by γ. The* node path *$p = (p_1, \ldots, p_{d+1})$ of $\mathcal{T}$ on input $w \in \mathbb{F}^n$ is defined by:*

- *p_1 is the root of $\mathcal{T}$.*
- *$p_{i+1} = \Gamma_{\mathcal{T}}(p_i, \langle \alpha_{p_i}, w \rangle)$.*

The edge path of $\mathcal{T}$ on w *is defined by $(\langle \alpha_{p_1}, w \rangle, \cdots, \langle \alpha_{p_d}, w \rangle)$. Lastly, the* output of $\mathcal{T}$ on w*, denoted $\mathcal{T}(w)$, is the value of $o_{p_{d+1}}$.*

Note that the edge path determines the computation path and output. Given the above, affine decision tree decoders are defined as follows.

Definition 5.9 (Affine decision tree decoder)**.** *An inversion algorithm* $\mathsf{C} := (\mathsf{C}_{\mathsf{pre}}, \mathsf{C}_{\mathsf{qry}}, \mathsf{C}_{\mathsf{dec}})$ *has a* d-depth affine decision tree decoder*, if for every* $y \in [n]$*,* $a \in \{0,1\}^s$ *and* $v = \mathsf{C}_{\mathsf{qry}}(y,a)$*, there exists an* n*-input,* d*-depth affine decision tree* $\mathcal{T}^{y,a}$ *such that* $\mathsf{C}_{\mathsf{dec}}(y,a,f(v)) = \mathcal{T}^{y,a}(f)$*.*

Note that such a decision tree may be of size $O(n^d)$. The following theorem bounds the probability, over a random function f, that a non-adaptive inverter with an affine decision tree decoder inverts a random output of f with probability τ.

Theorem 5.10. *Let* C *be an* s*-advice,* $(q \leq n/16)$*-query, non-adaptive inverter with a* d*-depth affine decision tree decoder, and let* $\tau \in [0,1]$*. Then for every* $\delta \in [0,1]$ *and* $m \leq \frac{n\log(n/q)}{4(d+1)\log n}$ *it holds that*

$$\Pr_{f\leftarrow\mathcal{F}}\left[\Pr_{\substack{x\leftarrow[n]\\ y=f(x)}}\left[\mathsf{C}(y;f)\in f^{-1}(y)\right]\geq\tau\right]\leq\alpha_{\tau,\delta}+2^s\cdot\delta^{-m}\prod_{j=1}^{m}\left(\frac{(d+1)j}{n}+\max\left\{\sqrt[4]{q/n},\frac{2(d+1)j\log n}{n\log(n/q)}\right\}\right)$$

for $\alpha_{\tau,\delta} := \Pr_{f\leftarrow\mathcal{F}_n}\left[\exists\tau n\text{-size set } \mathcal{X}\subset[n]\colon |f(\mathcal{X})|\leq\delta n\right]$*.*

Comparing to the bound we derive on affine decoders (Theorem 5.3), we are paying above for the tree depth d, but also for the number of queries q. In particular, we essentially multiply each term of the above product by the tree depth d, and by $\frac{\log n}{\log(n/q)}$. In addition, the theorem only holds for smaller values of m. The following corollary exemplifies the usability of Theorem 5.10 by considering the consequences of two choices of parameters.

Corollary 5.11 (Theorem 1.6, restated)**.** *Let* C *be as in Theorem 5.10 and assume* $\Pr_{f\leftarrow\mathcal{F}}\left[\Pr_{\substack{x\leftarrow[n]\\ y=f(x)}}\left[\mathsf{C}(y;f)\in f^{-1}(y)\right]\geq\tau\right]\geq {}^1\!/_2$*, then the following holds:*

- *If* $q \leq n\cdot({}^{\tau}\!/_2)^8$*, then* $s\in\Omega({}^{n}\!/_{d}\cdot{}^{\tau^2}\!/_{\log n})$*.*
- *If* $q\leq n^{1-\epsilon}$*, then* $s\in\Omega({}^{n}\!/_{d}\cdot\tau^2\epsilon)$*.*

Proof. Omitted, follows by Theorem 5.10 using very similar lines to those used to derive Corollary 5.4 from Theorem 5.3. □

In the rest of this section we explain how to modify the proof of Theorem 5.3, in order to derive the proof of Theorem 5.10. Hereafter we assume $n \geq 16$, as otherwise the bound trivially holds.

Let $F \leftarrow \mathcal{F}$ and let $Y = (Y_1, ..., Y_n)$ be a uniform vector over $[n]^n$. For a zero-advice affine decision tree inverter $\mathsf{D} = (\mathsf{D}_{\mathsf{qry}}, \mathsf{D}_{\mathsf{dec}})$, let X^{D} and Z^{D}, jointly distributed with F and Y, be according to Notation 5.5. The crux of the proof of Theorem 5.10 lies in the following lemma.

Lemma 5.12. *Let* $\mathsf{D} = (\mathsf{D}_{\mathsf{qry}}, \mathsf{D}_{\mathsf{dec}})$ *be a zero-advice,* $(q\leq n/16)$*-query, non-adaptive inverter with a* d*-depth affine decision trees decoder, and let* Z^{D} *be as in Notation 5.5. Then for every* $i\in[n]$ *and* $\mu\in[0,\frac{1}{2}]$*:*

$$\Pr\left[Z_i^{\mathsf{D}}\mid Z_{i-1}^{\mathsf{D}}\right]\leq\frac{(d+1)i-1}{n}+\mu+2^{2\lceil\mu n\rceil\log(1/\mu)+\lceil\mu n\rceil\log(q/n)+(i-1)(d+1)\log n}.$$

Proving Theorem 5.10.

Proof of Theorem 5.10 . Omitted, follows Lemma 5.12 using essentially the same lines we used to derive Theorem 5.3 from Lemma 5.6. □

5.2.1 Proving Lemma 5.12

Proof of Lemma 5.12. Fix $\mathsf{D} = (\mathsf{D}_{\mathsf{qry}}, \mathsf{D}_{\mathsf{dec}})$, $i \in [n]$ and $\mu \in [0, \frac{1}{2}]$. Let $\mathcal{T}^y$ be the affine decision tree associated with the computation of $\mathsf{D}_{\mathsf{dec}}$ on input y, let $p^y(f)$ and $g^y(f)$ be the node and edge paths, receptively, of $\mathcal{T}^y$ on f, and let $\alpha^y(f) := (\alpha^y_1(f), \ldots, \alpha^y_d(f)), o^y(f)$ be the labels of $p^y(f)$ according to $\mathcal{T}^y$. For $j \in [n]$, let $p^j = p^{Y_j}(F)$, $g^j = g^{Y_j}(F)$, $\alpha^j := \alpha^{Y_j}(F)$ and $o^j := o^{Y_j}(F)$. Finally, let $X := X^{\mathsf{D}}$ and $Z_j := Z^{\mathsf{D}}_j$. By definition, for every $j < i$:

1. $\forall k \in [d]: \quad \langle \alpha^j_k, F\rangle = g^k_j$,

2. $o^j = X_j$.

Conditioning on Z_{i-1} further implies that for every $j < i$:

2. $F(X_j) = Y_j$.

Let $\ell := (d+1)(i-1)$, and let $\mathbf{M}^{i-1} \in \mathbb{F}^{\ell \times n}$ be the (random) matrix defined by $\mathbf{M}^{i-1}_{(d+1)(j-1)+k} := \alpha^j_k$ for $k \in [d]$, and $\mathbf{M}^{i-1}_{(d+1)j} := e_{X_j}$. Let $V^{i-1} \in \mathbb{F}^{\ell}$ be the (random) vector defined by $V^{i-1}_{(d+1)(j-1)+k} := g^k_j$ and $V^{i-1}_{(d+1)j} := Y_j$. By definition, conditioned on Z_{i-1} it holds that $\mathbf{M}^{i-1} \times F = V^{i-1}$. That is, the matrix $\mathbf{M}^{i-1}$ contains also the internal computations done by the $i-1$ trees (and not only the final outcome of the each computation as in the affine decoder case). For $y \in [n]$, let $Q^y := \mathsf{D}_{\mathsf{qry}}(y) \in [n]^q$ be the queries that D makes on input y, and let $A^y \in [n]^q$ be F's answers to these queries. That is, for every $k \in [q]$:

3. $F(Q^y_k) = A^y_k$.

Let $\mathbf{E}^y \in \mathbb{F}^{q \times n}$ be the matrix defined by $\mathbf{E}^y_k := e_{Q^y_k}$. Let $\mathbf{M} := \begin{pmatrix}\mathbf{M}^{i-1}\\ \mathbf{E}^{Y_i}\end{pmatrix}$ and let $V := \begin{pmatrix}V^{i-1}\\ A^{Y_i}\end{pmatrix}$. That is, we add to $\mathbf{M}$ all queries made by D on input Y_i (and not only the output of the computations made by $\mathsf{D}_{\mathsf{dec}}$). We write

$$\Pr[Z_i \mid Z_{i-1}] = \Pr[Z_i \wedge X_i \in \mathcal{E}(\mathbf{M}) \mid Z_{i-1}] + \Pr[Z_i \wedge X_i \notin \mathcal{E}(\mathbf{M}) \mid Z_{i-1}] \tag{35}$$

and, using the above notions, prove the claim by separately bounding the two terms of the above equation. Let $H := (Y_{<i}, \mathbf{M}^{i-1}, V^{i-1})$, and note that

$$\begin{aligned}
&\Pr[Z_i \wedge X_i \in \mathcal{E}(\mathbf{M}) \mid Z_{i-1}] \leq \Pr[Y_i \in F(\mathcal{E}(\mathbf{M})) \mid Z_{i-1}] && (36)\\
&= \mathop{\mathrm{E}}_{h \leftarrow H|Z_{i-1}}\left[\Pr[Y_i \in F(\mathcal{E}(\mathbf{M})) \mid H = h, Z_{i-1}]\right]\\
&= \mathop{\mathrm{E}}_{h=(y_{<i}, m^{i-1}, v^{i-1}) \leftarrow H|Z_{i-1}}\left[\Pr\left[Y_i \in F\left(\mathcal{E}\begin{pmatrix}m^{i-1}\\ \mathbf{E}^{Y_i}\end{pmatrix}\right) \mid H = h, m^{i-1} \times F = v^{i-1}\right]\right]\\
&= \mathop{\mathrm{E}}_{(y_{<i}, m^{i-1}, v^{i-1}) \leftarrow H|Z_{i-1}}\left[\Pr\left[Y_i \in F\left(\mathcal{E}\begin{pmatrix}m^{i-1}\\ \mathbf{E}^{Y_i}\end{pmatrix}\right) \mid Y_{<i} = y_{<i}, m^{i-1} \times F = v^{i-1}\right]\right]\\
&= \mathop{\mathrm{E}}_{(y_{<i}, m^{i-1}, v^{i-1}) \leftarrow H|Z_{i-1}}\left[\Pr\left[Y_i \in F\left(\mathcal{E}\begin{pmatrix}m^{i-1}\\ \mathbf{E}^{Y_i}\end{pmatrix}\right) \mid m^{i-1} \times F = v^{i-1}\right]\right]\\
&\leq \left(\frac{(d+1)(i-1)}{n} + \mu\right) + 2^{2\lceil \mu n\rceil \log(1/\mu) + \lceil \mu n\rceil \log(q/n) + (d+1)(i-1)\log n}.
\end{aligned}$$

The first inequality holds by the definition of Z_i. The second equality holds by the definition of Z_{i-1}. The third equality holds since the event $\left\{Y_{<i} = y_{<i}, m^{i-1} \times F = v^{i-1}\right\}$ implies that $\left\{\mathbf{M}^{i-1} = m^{i-1}, V^{i-1} = v^{i-1}\right\}$. The last equality holds since F is independent of Y, and the last inequality follows by Claim 5.7 with respect to $\mathbf{A} := m^{i-1}, v := v^{i-1}$, and $(\mathbf{B}^1, \ldots, \mathbf{B}^n) := (\mathbf{E}^1, \ldots, \mathbf{E}^n)$.

For bounding the right-hand term of Equation (27), let $H := (X_i, Y_{\leq i}, \mathbf{M}, V)$, and compute

$$\begin{aligned}
&\Pr\left[Z_i \wedge X_i \notin \mathcal{E}(\mathbf{M}) \mid Z_{i-1}\right] \leq \Pr\left[Z_i \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}\right] \qquad (37)\\
&= \mathop{\mathrm{E}}_{h \leftarrow H \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}}\left[\Pr\left[Z_i \mid H = h, Z_{i-1}\right]\right]\\
&= \mathop{\mathrm{E}}_{h = (x_i, y_{\leq i}, m, v) \leftarrow H \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}}\left[\Pr\left[F(x_i) = y_i \mid H = h, m \times F = v\right]\right]\\
&= \mathop{\mathrm{E}}_{(x_i, y_{\leq i}, m, v) \leftarrow H \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}}\left[\Pr\left[F(x_i) = y_i \mid Y_{\leq i} = y_{\leq i}, m \times F = v\right]\right]\\
&= \mathop{\mathrm{E}}_{(x_i, y_{\leq i}, m, v) \leftarrow H \mid X_i \notin \mathcal{E}(\mathbf{M}), Z_{i-1}}\left[\Pr\left[F(x_i) = y_i \mid m \times F = v\right]\right]\\
&= 1/n.
\end{aligned}$$

The second equality holds by the definition of Z_{i-1}. The third equality holds since the event $\{Y_{\leq i} = y_{\leq i}, m \times F = v\}$ implies that $\{\mathbf{M} = m, V = v\}$, and X_i is a function of V (which contains all the answers to the queries of the decoder). The fourth equality holds since F is independent from Y. The last inequality follows by Claim 3.3. Combining Equations (35) to (37), we conclude that

$$\begin{aligned}
\Pr\left[Z_i \mid Z_{i-1}\right] &\leq \left(\frac{(d+1)(i-1)}{n} + \mu\right) + 2^{2\lceil \mu n \rceil \log(1/\mu) + \lceil \mu n \rceil \log(q/n) + (d+1)(i-1)\log n} + 1/n\\
&\leq \left(\frac{(d+1)i - 1}{n} + \mu\right) + 2^{2\lceil \mu n \rceil \log(1/\mu) + \lceil \mu n \rceil \log(q/n) + (d+1)(i-1)\log n}.
\end{aligned}$$

□

Acknowledgment

We are thankful to Dmitry Kogan, Uri Meir and Alex Samorodnitsky for very useful discussions. We also thank the anonymous reviewers for their comments.

References

[1] H. Abusalah, J. Alwen, B. Cohen, D. Khilko, K. Pietrzak, and L. Reyzin. Beyond hellman's time-memory trade-offs with applications to proofs of space. In *Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*, pages 357–379, 2017. 4

[2] Akshima, D. Cash, A. Drucker, H. Wee, et al. Time-space tradeoffs and short collisions in merkle-damgård hash functions. In *Annual International Cryptology Conference (CRYPTO)*, pages 157–186, 2020. 7

[3] N. Alon, I. Balla, L. Gishboliner, A. Mond, and F. Mousset. The minrank of random graphs over arbitrary fields. *Israel Journal of Mathematics*, 235(1):63–77, 2020. 5

[4] Z. Bar-Yossef, Y. Birk, T. Jayram, and T. Kol. Index coding with side information. *IEEE Transactions on Information Theory*, 57(3):1479–1494, 2011. 5

[5] A. Biryukov and A. Shamir. Cryptanalytic time/memory/data tradeoffs for stream ciphers. In *Annual International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT)*, pages 1–13, 2000. 4

[6] A. Biryukov, A. Shamir, and D. Wagner. Real time cryptanalysis of a5/1 on a pc. In *International Workshop on Fast Software Encryption (FSE)*, pages 1–18, 2000. 4

[7] S. Coretti, Y. Dodis, S. Guo, and J. Steinberger. Random oracles and non-uniformity. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 227–258, 2018. 4

[8] H. Corrigan-Gibbs and D. Kogan. The function-inversion problem: Barriers and opportunities. In *Theory of Cryptography (TCC)*, pages 393–421, 2019. , 1, 4, 5

[9] A. De, L. Trevisan, and M. Tulsiani. Time space tradeoffs for attacks against one-way functions and prgs. In *Annual International Cryptology Conference (CRYPTO)*, pages 649–665, 2010. 4

[10] Y. Dodis, S. Guo, and J. Katz. Fixing cracks in the concrete: Random oracles with auxiliary input, revisited. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 473–495, 2017. 4

[11] A. Fiat and M. Naor. Rigorous time-space trade-offs for inverting functions. *SIAM Journal on Computing*, 29(3):790–803, 2000. , 1, 4

[12] D. Galvin. Three tutorial lectures on entropy and counting. Technical Report 1406.7872, arXiv, 2014. 14

[13] R. Gennaro, Y. Gertner, J. Katz, and L. Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM Journal on Computing*, 35(1):217–246, 2005. 1, 4

[14] A. Golovnev, O. Regev, and O. Weinstein. The minrank of random graphs. *IEEE Transactions on Information Theory*, 64(11):6990–6995, 2018. 5

[15] I. Haviv and M. Langberg. On linear index coding for random graphs. In *2012 IEEE International Symposium on Information Theory Proceedings*, pages 2231–2235. IEEE, 2012. 5

[16] M. Hellman. A cryptanalytic time-memory trade-off. *IEEE transactions on Information Theory*, 26(4):401–406, 1980. , 1, 4, 8

[17] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Annual ACM Symposium on Theory of Computing (STOC)*, pages 44–61, 1989. 1

[18] E. Lubetzky and U. Stav. Nonlinear index coding outperforming the linear optimum. *IEEE Transactions on Information Theory*, 55(8):3544–3551, 2009. 5

[19] P. Oechslin. Making a faster cryptanalytic time-memory trade-off. In *Annual International Cryptology Conference (CRYPTO)*, pages 617–630, 2003. 4

[20] A. A. Razborov. On the distributional complexity of disjointness. *Theoretical Computer Science*, 106(2):385–390, 1992. 6, 16

[21] D. Unruh. Random oracles and auxiliary input. In *Annual International Cryptology Conference (CRYPTO)*, pages 205–223, 2007. 4

[22] L. G. Valiant. Graph-theoretic arguments in low-level complexity. In *International Symposium on Mathematical Foundations of Computer Science*, pages 162–176, 1977. 4

[23] L. G. Valiant. Why is boolean complexity theory difficult. *Boolean Function Complexity*, 169: 84–94, 1992. 4

[24] A. C. Yao. Protocols for secure computations. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 160–164, 1982. 5

[25] A.-C. Yao. Coherent functions and program checkers. In *Annual ACM Symposium on Theory of Computing (STOC)*, pages 84–94, 1990. , 1, 3, 4, 5

[26] A. C.-C. Yao. Should tables be sorted? *Journal of the ACM*, 28(3):615–628, 1981. 5