Electronic Colloquium on Computational Complexity, Report No. 113 (2016)

Time-Space Hardness of Learning Sparse Parities

Gillat Kol[*] Ran Raz [†] Avishay Tal [‡]

Abstract

We define a concept class $\mathcal{F}$ to be *time-space hard* (or *memory-samples hard*) if any learning algorithm for $\mathcal{F}$ requires either a memory of size super-linear in n or a number of samples super-polynomial in n, where n is the length of one sample. A recent work shows that the class of all parity functions is time-space hard [R16]. Building on [R16], we show that the class of all sparse parities of Hamming weight ℓ is time-space hard, as long as $\ell \geq \omega(\log n / \log\log n)$. Consequently, linear-size DNF Formulas, linear-size Decision Trees and logarithmic-size Juntas are all time-space hard. Our result is more general and provides time-space lower bounds for learning *any* concept class of parity functions.

We give applications of our results in the field of bounded-storage cryptography. For example, for every $\omega(\log n) \leq k \leq n$, we obtain an encryption scheme that requires a private key of length k, and time complexity of n per encryption/decryption of each bit, and is provenly and unconditionally secure as long as the attacker uses at most $o(nk)$ memory bits and the scheme is used at most $2^{o(k)}$ times. Previously, this was known only for $k = n$ [R16].

[*]Princeton University. This work was done when the author was a member at the Institute for Advanced Study, Princeton, NJ. Research supported by the National Science Foundation grant No. CCF-1412958.

[†]Weizmann Institute of Science, Israel, and the Institute for Advanced Study, Princeton, NJ. Research supported by the Israel Science Foundation grant No. 1402/14, by the I-CORE Program of the Planning and Budgeting Committee and the Israel Science Foundation, by the Simons Collaboration on Algorithms and Geometry, by the Fund for Math at IAS, and by the National Science Foundation grant No. CCF-1412958.

[‡]Institute for Advanced Study, Princeton, NJ. Research supported by the Simons Collaboration on Algorithms and Geometry, and by the National Science Foundation grant No. CCF-1412958.

ISSN 1433-8092

1 Introduction

Let $T \subseteq \{0,1\}^n$ be a set. In the problem of parity learning over T, there is an unknown string $x \in T$ that was chosen uniformly at random. A learner (who knows T) tries to learn x from samples (a, b), where $a \in_R \{0,1\}^n$ and $b = a \cdot x$ (where $a \cdot x$ denotes inner product modulo 2). That is, the learning algorithm is given a stream of samples, $(a_1, b_1), (a_2, b_2) \ldots$, where each a_t is uniformly distributed over $\{0,1\}^n$ and for every t, $b_t = a_t \cdot x$.

It was recently conjectured by Steinhardt, Valiant and Wager [SVW15] and proved in [R16] that any algorithm for parity learning over $T = \{0,1\}^n$ requires either a memory of quadratic size or an exponential number of samples.

In this paper, we give time-space tradeoff lower bounds for parity learning over the set T_ℓ containing all vectors of Hamming weight exactly ℓ. In particular, we prove that for any $\ell \geq \omega(\log n / \log\log n)$, any algorithm for parity learning over T_ℓ requires either a memory of super-linear size or a super-polynomial number of samples. Since a sparse parity function with sparsity ℓ is in particular a size-ℓ Junta and can be computed by a DNF formula of size $2^{O(\ell)}$ or a decision tree of size 2^ℓ, we conclude that for any $\ell \geq \omega(\log n / \log\log n)$, any learning algorithm for size-ℓ Juntas, size-2^ℓ DNF formulas or size-2^ℓ decision trees requires either a memory of super-linear size or a super-polynomial number of samples. This shows that some of the most extensively studied learning problems are infeasible under memory constraints.

We define a concept class $\mathcal{F}$ to be *time-space hard* (or *memory-samples hard*) if any learning algorithm for $\mathcal{F}$ requires either a memory of size super-linear in n or a number of samples super-polynomial in n, where n is the length of one sample [1]. Thus, for any $\ell \geq \omega(\log n / \log\log n)$, parity functions with sparsity ℓ, size-ℓ Juntas, size-2^ℓ DNF formulas and size-2^ℓ decision trees are time-space hard concept classes.

Our results are more general and provide time-space tradeoff lower bounds for parity learning over any set T.

1.1 Our Results

As in [R16], we model the learning algorithm by a *branching program*. A branching program is the strongest and most general model to use in this context. Roughly speaking, the model allows a learner with infinite computational power, and bounds only the memory size of the learner and the number of samples used.

Theorem 2 in Section 9 proves a general time-space lower bound for parity learning over a set T, in terms of the Fourier spectrum of the characteristic function of T. While the theorem gives non-trivial lower bounds for any set T of size $\geq \text{poly}(n)$, the bound is more significant when the characteristic function of T has a relatively small number of large Fourier coefficients.

More precisely, we say that a set T is an (ϵ, δ)-biased set if at most a δ fraction of the 2^n linear functions over $\{0,1\}^n$ are of bias larger than ϵ, on the uniform distribution over T. Theorem 2 shows that for any (ϵ, δ)-biased set T, any algorithm for parity learning over T requires either a memory of size larger than $\Omega\left(\log\left(\frac{1}{\epsilon}\right) \cdot \log\left(\frac{1}{\delta}\right)\right)$ or at least $\left(\frac{1}{\epsilon}\right)^{\Omega(1)}$ samples.

[1]This definition is tailored mainly for concept classes of size at most $2^{O(n)}$.

In Section 10, we prove several consequences of Theorem 2. First, as mentioned above, we prove time-space lower bounds for parity learning over the set T_ℓ of all vectors of Hamming weight exactly ℓ: Theorem 3 proves the following time-space lower bounds:

1. For any $\ell \leq n/2$, any algorithm for parity learning over T_ℓ requires either memory of size larger than $\Omega(n\ell)$ or at least $2^{\Omega(\ell)}$ samples.

2. For any $\ell \leq n^{0.9}$, any algorithm for parity learning over T_ℓ requires either memory of size larger than $\Omega(n \cdot \ell^{0.99})$ or at least $\ell^{\Omega(\ell)}$ samples.

In particular, the second result (and a padding argument) show that for any $\ell \geq \omega(\log n / \log \log n)$, parity learning over T_ℓ is time-space hard. In Section 3, we give an upper bound that shows that this result is tight, when the learner is modelled by a branching program.

Theorem 4 in Section 10 proves the following time-space lower bound for parity learning over ϵ-biased sets (as a simple consequence of Theorem 2):

> For any ϵ-biased set T, any algorithm for parity learning over T requires either memory of size larger than $\Omega(n \cdot \log(1/\epsilon))$ or at least $\left(\frac{1}{\epsilon}\right)^{\Omega(1)}$ samples.

Our results are stated for a learner that learns x *exactly*. Nevertheless, it is not hard to see that all of our results (including the consequences for Juntas, small DNF formulas and small decision trees) hold also for (both proper and improper) PAC learning. This is true because if a learner is able to output a function that *approximates* a parity function, the learner may as well output the parity function itself, as this function is unique (since parity functions are orthogonal to each other). The learner can compute the parity function because, as mentioned above, our lower bounds hold for a learner with an infinite computational power.

1.2 Related Works

Several recent works studied the resources needed for learning, under memory constraints (see in particular [S14, SVW15, R16] and the references there). Steinhardt, Valiant and Wager asked whether there exist concept classes that can be efficiently learnt from a polynomial number of samples, but cannot be learnt from a polynomial number of samples, under memory constraints, and suggested parity learning as a candidate [SVW15]. They conjectured that any algorithm for parity learning over $T = \{0,1\}^n$ requires either a memory of quadratic size or an exponential number of samples, a conjecture that was proven in [R16]. Our proofs build on the proof given in [R16].

Independently of our results, a recent work of Valiant and Valiant studies the problem of learning sparse parities, under information constraints [VV16]. However, their work focuses on the case where the learner can extract only r bits of information from each sample, where $r < n$. In the context of our paper, this gives a lower bound on the number of samples needed, when the memory size of the learner is at most n. In contrast, our work gives lower bounds on the number of samples needed, when the memory size of the learner is super-linear. The main motivation of Valiant and Valiant was constructing information theoretically secure databases [VV16].

1.3 Applications for Bounded Storage Cryptography

Let $T \subseteq \{0,1\}^n$ be a set and assume that any algorithm for parity learning (with non-negligible probability) over T requires either a memory of size at least s or more than m samples.

In [R16], the following application was suggested: Assume that a group of (two or more) users share a (random) secret key $x \in T$. Assume that user Alice wants to send an encrypted bit $M \in \{0,1\}$ to user Bob. Let a be a string of n bits, uniformly distributed over $\{0,1\}^n$, and assume that both Alice and Bob know a (we can think of a as taken from a source of randomness that streams random bits to all parties and if such a source is not available Alice can just choose a randomly and send it to Bob). Let b be the inner product of a and x, modulo 2. Thus, b is known to both Alice and Bob and can be used as a one time pad to encrypt/decrypt M, that is, Alice encrypts by computing $M \oplus b$ and Bob decrypts by computing $M = (M \oplus b) \oplus b$.

Assume that this protocol is used $m+1$ times, with the same secret key x. Denote by a_t, b_t the string a and bit b used at time t. Suppose that during all that time, an attacker could see $(a_1, b_1), \ldots, (a_m, b_m)$, but the attacker has less than s bits of memory. By the assumption, the attacker cannot guess the secret key x, with better than negligible probability. Therefore, using the fact that inner product is a strong extractor, even if the attacker sees a_{m+1}, the attacker cannot predict b_{m+1}, with better than negligible advantage over a random guess.

Thus, if the attacker has less than s bits of memory, the encryption remains secure as long as it is used at most $m+1$ times.

Using our result for ϵ-biased sets and the fact that such (explicitly constructed) sets of size $\text{poly}(n/\epsilon)$ exist [NN93, AGHP92], we obtain the following encryption/decryption schemes: for every $\omega(\log n) \leq k \leq n$, we obtain an encryption scheme that requires a private key of length k, and time complexity of n per encryption/decryption of each bit, and is provenly and unconditionally secure as long as the attacker uses at most $o(nk)$ memory bits and the scheme is used at most $2^{o(k)}$ times. (In [R16] such a scheme was obtained for $k = n$).

Using our first result for sparse parities, we obtain the following encryption/decryption schemes: for every $\omega(\log n) \leq \ell \leq n$, we obtain an encryption scheme that requires a private key of length $\log \binom{n}{\ell} \leq \ell \log n$, and time complexity of n per encryption/decryption of each bit, and is provenly and unconditionally secure as long as the attacker uses at most $o(n\ell)$ memory bits and the scheme is used at most $2^{o(\ell)}$ times.

This last scheme that is based on sparse parities is slightly worse than the one based on ϵ-biased sets, in terms of length of keys. Nevertheless, it has the advantage that encryption/decryption is done by taking the sum (modulo 2) of only ℓ bits of a, and thus, depending on the exact setting and model of computation considered, the time complexity may be considered to be ℓ, rather than n.

In all previous works on bounded storage cryptography, except for [R16] (see for example [M92, CM97, AR99, ADR02, V03, DM04], and many other works), the number of random bits transmitted during the encryption was assumed to be larger than the memory-size of the attacker. Thus, the time needed for encryption/decryption was at least linear in the memory-size of the attacker. In contrast, the encryption schemes here are secure against attackers with up to quadratic memory size.

2 Proof Outline

In this section, we give an overview of the proof of Theorem 2, our main theorem. Recall that we define a set T to be (ϵ, δ)-biased if at most a δ fraction of the 2^n linear functions over $\{0,1\}^n$ are of bias larger than ϵ, on the uniform distribution over T. Let $T \subseteq \{0,1\}^n$ be an (ϵ, δ)-biased set, and suppose that we want to prove a time-space lower bound for parity learning over T.

Computational Model

As in [R16], we model the learning algorithm by a *branching program*. A branching program of length m and width d, for parity learning, is a directed (multi) graph with vertices arranged in $m+1$ layers containing at most d vertices each. Intuitively, each layer represents a time step and each vertex represents a memory state of the learner. In the first layer, that we think of as layer 0, there is only one vertex, called the start vertex. A vertex of outdegree 0 is called a leaf. Every non-leaf vertex in the program has 2^{n+1} outgoing edges, labeled by elements $(a, b) \in \{0,1\}^n \times \{0,1\}$, with exactly one edge labeled by each such (a, b), and all these edges going into vertices in the next layer. Intuitively, these edges represent the action when reading (a_t, b_t). The samples $(a_1, b_1), \ldots, (a_m, b_m) \in \{0,1\}^n \times \{0,1\}$ that are given as input, define a computation-path in the branching program, by starting from the start vertex and following at Step t the edge labeled by (a_t, b_t), until reaching a leaf.

Each leaf v in the program is labeled by a vector $\tilde{x}(v) \in \{0,1\}^n$, that we think of as the output of the program on that leaf. We interpret the output of the program as a guess of x.[2]

We also consider *affine branching programs*, where every vertex v is labeled by an affine subspace $w(v) \subseteq \{0,1\}^n$, such that, the start vertex is labeled by the space $\{0,1\}^n$, and for any edge (u, v), labeled by (a, b), we have $w(u) \cap \{x' \in \{0,1\}^n : a \cdot x' = b\} \subseteq w(v)$. These properties guarantee that if the computation-path reaches a vertex v then $x \in w(v)$. Thus, we can interpret $w(v)$ as an affine subspace that is known to contain x.

Intuitively, each vertex v in an affine branching program "remembers" a set of linear equations that the input x satisfies, the equations that correspond to the vectors orthogonal[3] to $w(v)$. The *soundness* property, $w(u) \cap \{x' \in \{0,1\}^n : a \cdot x' = b\} \subseteq w(v)$ (for any edge (u, v), labeled by (a, b)) captures the fact that the only new equation that we can learn is the one that we just saw, in addition to linear combinations of that equation with equations we already remember. We may also forget equations in each step

An affine branching program is called *accurate* if for (almost) all vertices v, the distribution of x, conditioned on the event that the computation-path reached v, is close to the uniform distribution over $w(v) \cap T$.

For exact definitions, see Section 6.

[2]We note that our definition of the program's output differs from the original definition in [R16]. In [R16], the output was defined to be an affine subspace $w(v)$, that was interpreted as a guess that $x \in w(v)$.

[3]Here and below in this section, a vector is orthogonal to an affine subspace w if it corresponds to a linear function that is constant on w.

The High-Level Approach

We follow the proof structure of [R16], and divide our proof into two parts: We first show how to reduce general branching programs to accurate affine branching programs, and then prove lower bounds for accurate affine branching programs. Both parts of the proof differ from the proof of [R16], as we need to take into account the more general structure of the underlying set T. Below, we highlight the new emerged problems and their solutions.

The proof of [R16] considers the case where $T = \{0,1\}^n$. One property crucially used by the proof is that the set T "samples well" every large affine subspace, say a subspace of dimension larger than $n/2$. In particular, the proof can be generalized to handle the case where T is a large random set. For a general set T and an affine subspace w, such that $w \cap T \neq \emptyset$, we define

$$C_w = C_{w,T} = \frac{\Pr_{x'}[x' \in w] \cdot \Pr_{x'}[x' \in T]}{\Pr_{x'}[x' \in w \cap T]},$$

where the probabilities are over a uniformly random x' in $\{0,1\}^n$. We note that $C_{w,T}$ measures the correlation between T and w, and captures how well T samples w (if $C_{w,T}$ is close to 1 then T samples w well). We will focus on affine subspaces of high dimension (or, low co-dimension), as if T is relatively small, it cannot sample well most of the subspaces of low dimension.

We want to consider "good" subspaces w, of (low co-dimension and) $C_{w,T}$ close to 1. However, it will be helpful for us to require a stronger property, that the vector space orthogonal to w does not contain large Fourier coefficients of $\mathcal{U}_T$, the uniform distribution over T. (It is relatively easy to prove that this is a stronger property – see Claim 5.3). Hence, we define $\mathcal{G}_{T,k}$ as the set of all affine subspaces w of co-dimension at most k, such that, $w \cap T \neq \emptyset$ and the space orthogonal to w does not contain any large Fourier coefficient of $\mathcal{U}_T$. Let v be a vertex in an affine branching program. We say that v is *good* if $w(v) \in \mathcal{G}_{T,k}$.

Intuitively, the definition of a good vertex requires that the equations that the vertex "remembers" are not biased with respect to the set T.

The notion of *good* vertices plays a major role in our proof. First, in the reduction from general branching programs to accurate affine branching programs, we will get no upper bound at all on the number of *bad* vertices in the simulating affine program; we will only get an upper bound on the number of good vertices in that program. Nevertheless, this will be sufficient for us. In the proof of the lower bound for accurate affine branching programs, we show that since the number of good vertices is bounded, the probability that the computation-path reaches *any* bad vertex is small. In particular, the last vertex reached by the computation-path is good with high probability. Since the last vertex reached by the computation-path is good (with high probability), and since the affine branching program is accurate, it is relatively easy to show that conditioned on the event that the computation-path reached that vertex, the input x is close to being uniformly distributed over a large set, which implies that the program cannot output the correct x with non-negligible probability.

From Branching Programs to Accurate Affine Branching Programs

In Section 7, we show how to simulate a branching program by an accurate affine branching program. We do that layer after layer. Assume that we are already done with layer $j-1$, so every vertex in layer $j-1$ is already labeled by an affine subspace, and the distribution of x, conditioned on the event that the computation-path reached that vertex, is close to the uniform distribution over the intersection of T and the affine subspace that labels the vertex.

Now, take a vertex v in layer j, and consider the distribution of x, conditioned on the event that the computation-path reached the vertex v. By the property that we already know on layer $j-1$, this distribution is close to a convex combination of uniform distributions over the intersection of T and affine subspaces of $\{0,1\}^n$.

One could split v into a large number of vertices, one vertex for each affine subspace in the combination. However, this practically means that we would have a vertex for any affine subspace. We would like to keep the number of vertices somewhat smaller. This is done by grouping many affine subspaces into one group. The group will be labeled by an affine subspace that contains all the affine subspaces in the group. Moreover, we will have the property that for each such group, the uniform distribution over the intersection of T and the affine subspace that labels the group is close to the relevant weighted average of the uniform distributions over the intersection of T and the affine subspaces in the group. Thus, practically, we can replace all the affine subspaces in the group by one affine subspace that represents all of them.

Lemma 5.6 shows that it is possible to group all the good affine subspaces $w \in \mathcal{G}_{T,k}$ into a relatively small number of groups, each labeled by a good subspace $s \in \mathcal{G}_{T,k}$. This lemma does not group the bad subspaces, and keeps one vertex for each bad subspace. We conclude that in the simulating affine branching program there is a relatively small number of good vertices (but we will not have any upper bound on the number of bad vertices).

We will now sketch the proof of Lemma 5.6. Let W be a random variable distributed over good affine subspaces. Our goal is to group the good affine subspaces in the support of W into a relatively small number of good affine subspaces. The proof works by finding a subspace s such that $\Pr[W \subseteq s]$ is large and the distribution $\mathbf{E}_{W|W\subseteq s}[\mathcal{U}_{W\cap T}]$ is close to the uniform distribution over $s \cap T$. The subspaces in the support of W that are contained in s are grouped together to a new vertex labeled by s. This process is then repeated for $(W|W \nsubseteq s)$, until getting a good approximate covering of W. Observe that if s captures at least $1/M$ of the probability mass of W, then $\Pr[W \nsubseteq s] \leq 1-1/M$, and repeating the process by induction $t = O(M \cdot \log(1/\epsilon))$ times covers all but $(1-1/M)^t \leq \epsilon$ of the probability mass of W.

To find the subspace s, we apply an iterative process that defines a sequence of affine subspaces $s_0 \supseteq s_1 \supseteq s_2 \supseteq \ldots \supseteq s_{k'} = s$ where $0 \leq k' \leq k$ and such that each subspace s_i is of co-dimension i. We start with $s_0 = \{0,1\}^n$. To construct s_i from s_{i-1} we consider two cases. Let $r \in [0,n]$ be some parameter to be optimized later. For $a \in \{0,1\}^n$ and $b \in \{0,1\}$, such that a is not in the orthogonal to s_{i-1}, let $s_{a,b}$ be the subspace defined by $\{x' \in s_{i-1} : \ a \cdot x' = b\}$. The two cases that we consider are:

1. There exists a, b such that $\Pr[W \subseteq s_{a,b} | W \subseteq s_{i-1}] \geq 2^{-r}$.

2. For every a, b it holds that $\Pr[W \subseteq s_{a,b} | W \subseteq s_{i-1}] < 2^{-r}$.

In the first case, we define $s_i = s_{a,b}$ (and note that $\mathrm{codim}(s_{a,b}) = \mathrm{codim}(s_{i-1}) + 1$). If the second case applies, we end the sequence of affine subspaces and set $s = s_{i-1}$ (thus, $k' = i-1$).

Most of our effort goes into showing that indeed in this case, $\left|\mathbf{E}_{W|W\subseteq s}[\mathcal{U}_{W\cap T}] - \mathcal{U}_{s\cap T}\right|_1$ is small (Lemma 5.4). The proof of Lemma 5.4 uses Fourier analysis and is described next.

Proof of Lemma 5.4

Consider a random variable W, supported on affine subspaces in $\mathcal{G}_{T,k}$, and consider the conditional random variable $(W|W \subseteq s)$. We assume that for every linear equation $a \cdot x' = b$ (not already known to hold for s), we have $\Pr[W \subseteq s_{a,b}|W \subseteq s] < 2^{-r}$. Under this assumption, we prove that $\left|\mathbf{E}_{W|W\subseteq s}[\mathcal{U}_{W\cap T}] - \mathcal{U}_{s\cap T}\right|_1$ is small. We first consider the case where $s = \{0,1\}^n$ and then reduce the general case to this special case.

The case $s = \{0,1\}^n$. Denote $D = \mathbf{E}_W[\mathcal{U}_{W\cap T}]$. To bound $|D - \mathcal{U}_T|_1$, we use the standard Cauchy-Schwartz inequality and Parseval's identity to get

$$(|D - \mathcal{U}_T|_1)^2 \le |T| \cdot 2^n \cdot \sum_{\alpha\in\{0,1\}^n} \left(\widehat{D}(\alpha) - \widehat{\mathcal{U}}_T(\alpha)\right)^2 . \tag{1}$$

[R16] considered the case $T = \{0,1\}^n$ and showed that each individual term in the sum $\sum_{\alpha\in\{0,1\}^n} (\widehat{D}(\alpha) - \widehat{\mathcal{U}}_T(\alpha))^2$ is small, concluding that the RHS of Equation (1) is small. We cannot afford bounding each term individually, and need to rely on cancellations. To allow cancellations, it is much more convenient to bound the ℓ_1-distance between $\mathcal{U}_T$ and $\frac{D}{C}$ for some constant $C > 0$. This is the ℓ_1 distance between a distribution $\mathcal{U}_T$ and some non-negative function $\frac{D}{C}$, which is not necessarily a distribution (in fact, $\frac{D}{C}$ is a distribution only if $C = 1$). Claim 5.1 easily shows that a bound on $\left|\mathcal{U}_T - \frac{D}{C}\right|_1$ implies a bound on $|\mathcal{U}_T - D|_1$, losing only a multiplicative factor of 2 (regardless of C).

We pick the value of C to be $\mathbf{E}_W[C_W]$. Using the convolution formula, we show that many cancellations occur, and prove that

$$\begin{aligned}(|D - \mathcal{U}_T|_1)^2 &\le O\left(\left|\tfrac{D}{C} - \mathcal{U}_T\right|_1\right)^2 \le O(|T| \cdot 2^n) \cdot \sum_{\alpha\in\{0,1\}^n} \left(\frac{\widehat{D}(\alpha)}{C} - \widehat{\mathcal{U}}_T(\alpha)\right)^2 \\ &\le O(2^n) \cdot \mathop{\mathbf{E}}_{W_1,W_2}\left[\frac{C_{W_1}}{C} \cdot \frac{C_{W_2}}{C} \cdot \sum_{0\ne\beta_1\in W_1^\perp} \sum_{0\ne\beta_2\in W_2^\perp} \left|\widehat{\mathcal{U}}_T(\beta_1+\beta_2)\right|\right] ,\end{aligned} \tag{2}$$

where the expectation in the last expression is taken over independent random variables W_1, W_2 with the same distribution as W, and the summation is taken over all $\beta_1, \beta_2 \ne 0$ in the vector spaces orthogonal to W_1, W_2, respectively.

We mention that our proof for Equation (2) uses the fact that $\mathcal{U}_T$ is the uniform distribution over a set T, as we use the fact that $\mathcal{U}_T(x)^2 = \mathcal{U}_T(x)/|T|$ for all $x \in \{0,1\}^n$ (indeed, for $x \in T$ both sides equal $1/|T|^2$, and for $x \notin T$ both sides equal 0). Note that in the right hand side of Equation (2) we are taking the expected sum over at most 2^{2k} Fourier coefficients - as opposed to 2^n Fourier coefficients in the original sum. Note also the multiplicative factor $\frac{C_{W_1}}{C}\cdot\frac{C_{W_2}}{C}$ that motivated us to define the set $\mathcal{G}_{T,k}$ in the first place. Under the assumption that W is supported only on good subspaces, we get that $\frac{C_{W_1}}{C} \cdot \frac{C_{W_2}}{C} = O(1)$.

It remains to bound the right hand side of Equation (2). For this bound, we use the fact that each nonzero vector $\beta \in \{0,1\}^n$ is orthogonal to W with probability at most 2^{-r}, and the fact that T is an (ϵ, δ)-biased set.

The general case for s. In the general case, s can be any affine subspace of co-dimension k' in $\mathcal{G}_{T,k}$. Apply an invertible affine transformation to $\{0,1\}^n$ that maps s to $\{0,1\}^{n-k'} \times 0^{k'}$, and consider T and W under that transformation. Such a transformation maintains the ℓ_1 distance between distributions, as well as the Fourier spectrum of T. Thus, we have reduced the problem of a general s to the case of $s = \{0,1\}^{n-k'} \times 0^{k'}$.

Let $T' = s \cap T$. The distance between $\mathcal{U}_{s \cap T}$ and $\mathbf{E}_{W|W \subseteq s}[\mathcal{U}_{W \cap T}]$ equals the distance between $\mathcal{U}_{T'}$ and $\mathbf{E}_{W|W \subseteq s}[\mathcal{U}_{W \cap T'}]$. To analyze the distance between $\mathcal{U}_{T'}$ and $\mathbf{E}_{W|W \subseteq s}[\mathcal{U}_{W \cap T'}]$, we may forget the last k' coordinates as they are always 0, and use the case $s = \{0,1\}^{n-k'}$. The only non-trivial manipulation we did in terms of Fourier coefficients is considering $T' = s \cap T$, rather than T. We wish to show that if T is an (ϵ, δ)-biased set then T' is an $(O(\epsilon \cdot 2^k), \delta \cdot 2^k)$-biased set. This is done using Claim 5.2 that shows how to write each Fourier coefficient of $\mathcal{U}_{T'}$ as the product of C_s and an (alternating) sum of up to 2^k Fourier coefficients of $\mathcal{U}_T$, and using the fact that since $s \in \mathcal{G}_{T,k}$, we have $C_s = O(1)$.

Note that we use the fact that W is supported only on good affine subspaces both in the case $s = \{0,1\}^n$ (to bound C_{W_1}, C_{W_2} and C) and in the reduction from the general case to the case $s = \{0,1\}^n$ (to bound C_s).

Our actual proof for Lemma 5.4 combines the two cases together, as on a technical level handling the general case is not much more difficult than handling the case $s = \{0,1\}^n$.

Lower Bounds for Accurate Affine Branching Programs

Assume that we have an accurate affine branching program with a relatively small length and a relatively small number of good vertices. We prove that the probability that the computation-path reaches any bad vertex is small. A time-space lower bound follows, because the affine branching program is accurate, thus conditioned on reaching a good vertex v, x is almost uniformly distributed over $w(v) \cap T$. Since v is good it follows that $w(v)$ is of co-dimension at most k and that $C_{w(v),T}$ is close to 1. Therefore, $|w(v) \cap T| \geq \Omega\left(2^{-k} \cdot |T|\right)$. Since a good vertex is almost always reached, it is almost always the case that when the program stops, x is still close to being uniformly distributed over a large set.

Towards the end of showing that the probability that the computation-path reaches any bad vertex is small, we prove the following two lemmas. Both lemmas below are proven under the assumption that all vertices in the branching program are labeled with affine subspaces of co-dimension $\leq k$.

1. In Lemma 8.1 we prove that for every vertex v, such that the co-dimension of $w(v)$ is k, the probability of reaching v is extremely small. Since we assume that the affine branching program has a relatively small number of good vertices, we get, by a union bound, that the probability of reaching *any* good vertex of co-dimension k is small.

2. In Lemma 8.2 we prove that the probability of reaching any bad vertex is small.

In the proof of Theorem 1, we combine the above lemmas and show that the probability that the computation-path reaches any bad vertex is small, without using the assumption that all vertices in the branching program are labeled with affine subspaces of co-dimension $\leq k$. Theorem 2 is an easy corollary of Theorem 1.

3 Upper Bounds

There are two trivial protocols for parity learning over any set T, by a branching program. First, the branching program can store $O(\log |T|)$ samples, using $O(n \cdot \log |T|)$ memory bits, and find x by trying all the possibilities. Note that this is not a computationally efficient protocol, but the computational efficiency is irrelevant when considering a branching program, as we only care about the width and length of the program, which correspond to the memory size and number of samples used by the program. Second, parity learning over T can be solved by a branching program of width $O(1)$, by trying all the possibilities one after the other, using $O(|T| \cdot \log |T|)$ samples.

We next describe a third protocol for parity learning over T_ℓ that only requires memory of size $o(n)$ and $(\ell + \log n)^{O(\ell)}$ samples (for any $\ell \leq \sqrt{n/2}/\log n$). In particular, note that for $\ell \leq O(\log(n)/\log\log(n))$, the memory is of size $o(n)$ and the number of samples needed is polynomial in n. The protocol exhibits the tightness of our lower bound – indeed, Theorem 3 shows that when the number of samples is at most $\ell^{c \cdot \ell}$ for some small constant $c > 0$, the memory required to learn x is at least $n \cdot \ell^{0.99}$. The protocol proceeds as follows:

1. **Guessing a set containing the support of x.** The learner guesses a set $S \subseteq [n]$ of size n/t, where t is a parameter. This is done by randomly sampling n/t distinct coordinates in $[n]$. If $n/t \geq 2\ell$, the probability that the set S contains the support of x (the set of non-zero coordinates of x) is larger than $\Omega(t^{-\ell})$. In the next step, the learner assumes that indeed S contains the support of x.

2. **Obtaining a candidate for x.** The learner gathers $O(\ell \cdot \log n)$ samples, and stores each sample (a, b) by storing b and storing the coordinates of a that are in the set S. The learner then goes over all possible vectors x' with sparsity ℓ and support contained in S, in order to find a sparsity-ℓ vector x' that satisfies all the equations of the form $a \cdot x' = b$ induced by the stored samples. Observe that since the support of x' is contained in S, there is no need of storing the coordinates of a not in S in order to find x'. Since the samples are random, each sample gives roughly one bit of information about x. As x is specified by less than $\ell \cdot \lceil \log n \rceil$ bits, the induced set of equations has at most one solution with high probability. The solution x', if exists, is the current candidate for x (if more than one solution exists, the learner picks one of them as x' arbitrarily). Note that if the set S indeed contains the support of x, then $x' = x$ with high probability.

3. **Checking the candidate x'.** If the above steps produced a candidate x', the learner uses another $O(\ell \cdot \log t)$ samples, one after the other, and checks whether the candidate x' is consistent with all of them. If x' is consistent with all of them, the learner outputs x'. Otherwise, the learner returns to Step 1 and repeats the process. Observe that since the probability that the set S contains the support of x is larger than $\Omega(t^{-\ell})$, the learner is likely to find the correct solution after $O(t^\ell)$ repetitions. Note that we used $O(\ell \cdot \log t)$ samples to check the solution, to ensure that the probability for a false solution to pass all tests would be smaller than $2^{-O(\ell \cdot \log t)} = t^{-O(\ell)}$, which ensures that with high probability we will not get a false solution even after $O(t^\ell)$ repetitions.

The protocol requires memory of size $O(\ell \cdot \log(n) \cdot n/t)$. Since a random set S of size n/t contains the support of x with probability $\Omega(t^{-\ell})$ (if $n/t \geq 2\ell$), the number of samples required is $O(t^{\ell} \cdot \ell \cdot \log n)$. By taking $t = \ell \cdot (\log n)^2$, we get a number of samples of $(\ell + \log n)^{O(\ell)}$ and memory of size $o(n)$ (for any ℓ such that $n/t \geq 2\ell$, that is, $\ell \leq \sqrt{n/2}/\log n$).

4 Preliminaries

4.1 Parity Learning

Let $T \subseteq \{0,1\}^n$ be a set. In the problem of parity learning over T, there is an unknown string $x \in T$ that was chosen uniformly at random. A learner (who knows T) tries to learn x from samples (a, b), where $a \in_R \{0,1\}^n$ and $b = a \cdot x$ (where $a \cdot x$ denotes inner product modulo 2). That is, the learning algorithm is given a stream of samples, $(a_1, b_1), (a_2, b_2) \ldots$, where each a_t is uniformly distributed over $\{0,1\}^n$ and for every t, $b_t = a_t \cdot x$.

When T is known from the context, we omit T, and refer to the problem of "parity learning over T" simply as "parity learning".

4.2 Notation

For an integer n, denote $[n] = \{1, \ldots, n\}$. For $a, x \in \{0,1\}^n$, denote by $a \cdot x$ their inner product modulo 2.

For a function $P : \Omega \to \mathbb{R}$, we denote by $|P|_1$ its ℓ_1 norm. In particular, for two functions, $P, Q : \Omega \to \mathbb{R}$, we denote by $|P - Q|_1 = \sum_{x \in \Omega} |P(x) - Q(x)|$ their ℓ_1 distance.

For a random variable X and an event E, we denote by $\mathbb{P}_X$ the distribution of the random variables X, and we denote by $\mathbb{P}_{X|E}$ the distribution of the random variable X conditioned on the event E.

For a set $S \subseteq \{0,1\}^n$, denote by $\mathcal{U}_S$ the uniform distribution over S.

For $n \in \mathbb{N}$, denote by $\mathcal{A}(n)$ the set of all affine subspaces of $\{0,1\}^n$. For an affine subspace $w \in \mathcal{A}(n)$, we define

$$\begin{aligned} w^{\perp} &= \{(\beta, b) \in \{0,1\}^n \times \{0,1\} : \ \forall x \in w, \ \beta \cdot x = b\}, \\ (w^{\perp})_1 &= \left\{\beta \in \{0,1\}^n : \ \exists b \in \{0,1\}, \ (\beta, b) \in w^{\perp}\right\}. \end{aligned}$$

4.3 Fourier Coefficients

For $\alpha \in \{0,1\}^n$, let $\chi_\alpha : \{0,1\}^n \to \{-1, 1\}$ be the character given by $\chi_\alpha(x) = (-1)^{\alpha \cdot x}$. For a function $f : \{0,1\}^n \to \mathbb{R}$, let $\hat{f}(\alpha) \in \mathbb{R}$ be the Fourier coefficient given by

$$\hat{f}(\alpha) = \mathop{\mathbf{E}}_{x \in_R \{0,1\}^n} [f(x) \cdot \chi_\alpha(x)].$$

Let D be a distribution over $\{0,1\}^n$. We view D as a function $D : \{0,1\}^n \to \mathbb{R}^+$ with $\sum_{x \in \{0,1\}^n} D(x) = 1$. The Fourier coefficients of D are given by

$$\hat{D}(\alpha) = \mathop{\mathbf{E}}_{x \in_R \{0,1\}^n} [D(x) \cdot \chi_\alpha(x)] = 2^{-n} \cdot \mathop{\mathbf{E}}_{x \sim D} [\chi_\alpha(x)].$$

4.4 Definitions

Let $n \in \mathbb{N}$, $T \subseteq \{0,1\}^n$. For $w \in \mathcal{A}(n)$ such that $w \cap T \neq \emptyset$, we define

$$C_{w,T} = \frac{|w| \cdot |T|}{2^n |w \cap T|} = \frac{(|w|/2^n) \cdot (|T|/2^n)}{|w \cap T|/2^n},$$

that measures the *correlation* between w and T. For $\epsilon > 0$, we define the set $\mathcal{B}_T(\epsilon) \subseteq \{0,1\}^n$ of *big* Fourier coefficients of $\mathcal{U}_T$ by

$$\mathcal{B}_T(\epsilon) = \left\{ \alpha \in \{0,1\}^n : \left| \widehat{\mathcal{U}}_T(\alpha) \right| > 2^{-n}\epsilon \right\}.$$

For $k \in \mathbb{N}$, $k \leq n$, we define the set $\mathcal{G}_{T,k} \subseteq \mathcal{A}(n)$ of "*good*" affine subspaces by

$$\mathcal{G}_{T,k} = \left\{ w \in \mathcal{A}(n) : \ \mathrm{codim}(w) \leq k \ \text{ and } \ w \cap T \neq \emptyset \ \text{ and } \ (w^{\perp})_1 \cap \mathcal{B}\left(2^{-(k+1)}\right) = \{\vec{0}\} \right\}.$$

4.4.1 Global Definitions

Sections 5-9 refer to a general set $T \subseteq \{0,1\}^n$ and a maximal co-dimension $k \in \mathbb{N}$. Throughout these sections, we think of T and k as fixed. For simplicity of notation, we refer to $C_{w,T}$ as C_w, to $\mathcal{B}_T(\epsilon)$ as $\mathcal{B}(\epsilon)$, and to $\mathcal{G}_{T,k}$ as $\mathcal{G}$, in all these sections.

From Section 10 on, we analyze some specific choices of T and k.

4.5 A Simple Bound on $\mathcal{B}_T(\epsilon)$

Lemma 4.1. *Let $\epsilon \in (0,1)$. Then, for any $T \subseteq \{0,1\}^n$ we have $|\mathcal{B}_T(\epsilon)| \leq \frac{2^n}{|T| \cdot \epsilon^2}$.*

Proof. Recall that

$$\mathcal{B}_T(\epsilon) = \left\{ \alpha \in \{0,1\}^n : \left| \widehat{\mathcal{U}}_T(\alpha) \right| > 2^{-n}\epsilon \right\}.$$

By Parseval's identity,

$$|\mathcal{B}_T(\epsilon)| \cdot (2^{-n}\epsilon)^2 \leq \sum_{\alpha \in \mathcal{B}_T(\epsilon)} \left(\widehat{\mathcal{U}}_T(\alpha) \right)^2 \leq \sum_{\alpha \in \{0,1\}^n} \left(\widehat{\mathcal{U}}_T(\alpha) \right)^2 = \mathop{\mathbf{E}}_{x \in_R \{0,1\}^n} \left[(\mathcal{U}_T(x))^2 \right] = \frac{1}{|T| \cdot 2^n}$$

Therefore, $|\mathcal{B}_T(\epsilon)| \leq \frac{2^n}{|T| \cdot \epsilon^2}$ □

5 Distributions over Affine Subspaces

Recall that throughout this section, we think of the set $T \subseteq \{0,1\}^n$ and the maximal co-dimension $k \in \mathbb{N}$ as fixed (see Section 4.4.1). The section studies convex combinations of uniform distributions over sets of the form $w \cap T$, where $w \in \mathcal{A}(n)$ is an affine subspace of co-dimension at most k.

5.1 Claims

We first prove some simple claims that will be used in this section.

Claim 5.1. *Let $D : \Omega \to [0,1]$ be a probability distribution. Let $F : \Omega \to \mathbb{R}^+$ be a function. Let $C = \sum_{x\in\Omega} F(x)$. If $|F - D|_1 \le \epsilon$, then the probability distribution F/C satisfies $|F/C - D|_1 \le 2\epsilon$.*

Proof. First assume that $C > 1$. Let $A = \{x \in \Omega : \ F(x)/C > D(x)\}$. It holds that $\sum_{x\in A}(F(x)/C - D(x)) \le \sum_{x\in A}(F(x) - D(x)) \le \sum_{x\in\Omega} |F(x) - D(x)| \le \epsilon$. Observe that $\sum_{x\in A}(F(x)/C - D(x)) = \sum_{x\in\Omega\setminus A}(D(x) - F(x)/C)$ as D and F/C are probability distributions. Therefore $|F/C - D|_1 \le 2\epsilon$.

The case $C < 1$ is handled similarly, by considering the set $A' = \{x \in \Omega : \ F(x)/C < D(x)\}$. □

Claim 5.2. *Let $w \in \mathcal{A}(n)$ be an affine subspace, such that $w \cap T \ne \emptyset$. For every $\alpha \in \{0,1\}^n$,*

$$\widehat{\mathcal{U}}_{w\cap T}(\alpha) = C_w \cdot \sum_{(\beta,b)\in w^\perp} (-1)^b \cdot \widehat{\mathcal{U}}_T(\alpha + \beta).$$

Proof. An easy calculation shows that the Fourier coefficients of $\mathcal{U}_w$ are:

$$\widehat{\mathcal{U}}_w(\beta) = \begin{cases} (-1)^b \cdot 2^{-n} & \text{if } (\beta,b) \in w^\perp \\ 0 & \text{otherwise} \end{cases}$$

The Fourier coefficients of $\mathcal{U}_{w\cap T}$ are:

$$\begin{aligned} \widehat{\mathcal{U}}_{w\cap T}(\alpha) &= \frac{1}{|w\cap T|} \cdot \widehat{1}_{w\cap T}(\alpha) = \frac{1}{|w\cap T|} \cdot \widehat{1_w \cdot 1_T}(\alpha) = \frac{|w|\cdot|T|}{|w\cap T|} \cdot \widehat{\mathcal{U}_w \cdot \mathcal{U}_T}(\alpha) \\ &= \frac{|w|\cdot|T|}{|w\cap T|} \cdot \sum_{\beta\in\{0,1\}^n} \widehat{\mathcal{U}}_w(\beta) \cdot \widehat{\mathcal{U}}_T(\alpha+\beta) \qquad \text{(convolution)} \\ &= C_w \cdot \sum_{(\beta,b)\in w^\perp} (-1)^b \cdot \widehat{\mathcal{U}}_T(\alpha+\beta). \end{aligned}$$

□

Claim 5.3. *For $w \in \mathcal{G}$ it holds that $C_w \in (2/3, 2)$.*

Proof. Since $\widehat{\mathcal{U}}_{w\cap T}(\vec{0}) = 2^{-n}$, Claim 5.2 applied with $\alpha = \vec{0}$ gives

$$1/C_w = 2^n \cdot \sum_{(\beta,b)\in w^\perp} (-1)^b \cdot \widehat{\mathcal{U}}_T(\beta) = 1 + 2^n \cdot \sum_{(\beta,b)\in w^\perp\setminus(\vec{0},0)} (-1)^b \cdot \widehat{\mathcal{U}}_T(\beta).$$

Since $w \in \mathcal{G}$, it holds that $|w^\perp| \le 2^k$ and $\forall \beta \in (w^\perp)_1 \setminus \{\vec{0}\} : \ \left|\widehat{\mathcal{U}}_T(\beta)\right| \le 2^{-n} \cdot 2^{-(k+1)}$. Therefore,

$$\left| \sum_{(\beta,b)\in w^\perp\setminus(\vec{0},0)} (-1)^b \cdot \widehat{\mathcal{U}}_T(\beta) \right| \le (2^k - 1) \cdot 2^{-n} \cdot 2^{-(k+1)} < 2^{-n-1}.$$

Conclude that $1/C_w \in (1/2, 3/2)$. □

5.2 Main Lemmas

Lemma 5.4. *Let $s \in \mathcal{G}$. Denote by $\mathcal{A}(s)$ the set of all affine subspaces of s. Let $W \in \mathcal{A}(s) \cap \mathcal{G}$ be a random variable. Let $r, \epsilon > 0$. Assume that for every $(a, b) \in (\{0,1\}^n \times \{0,1\}) \setminus s^{\perp}$, it holds that*

$$\Pr_W [\forall x \in W : a \cdot x = b] \leq 2^{-r}.$$

Then,

$$\left| \mathop{\mathbf{E}}_W [\mathcal{U}_{W \cap T}] - \mathcal{U}_{s \cap T} \right|_1 < 2^{k+3} \sqrt{2^{-r} |\mathcal{B}(\epsilon)| + \epsilon}.$$

Proof. Denote $\mu = \mathbb{P}_W$. We define the probability distribution ρ over $\mathcal{A}(s) \cap \mathcal{G}$ by $\rho(w) = \frac{C_w}{C} \cdot \mu(w)$, where $C = \sum_{w \in \mathcal{A}(s) \cap \mathcal{G}} C_w \cdot \mu(w)$ is a normalization factor. Since $W \in \mathcal{G}$, by Claim 5.3, for every $w \in \mathrm{supp}(\mu)$ we have $C_w \in (2/3, 2)$. Therefore also $C \in (2/3, 2)$. For any event $\mathcal{E}$ over the space $\mathcal{A}(s) \cap \mathcal{G}$, we have

$$\Pr_{w \sim \rho} [\mathcal{E}] = \sum_{w\colon \mathcal{E}(w) = \text{true}} \rho(w) = \sum_{w\colon \mathcal{E}(w) = \text{true}} \frac{C_w}{C} \cdot \mu(w) < \frac{2}{2/3} \cdot \sum_{w\colon \mathcal{E}(w) = \text{true}} \mu(w) = 3 \cdot \Pr_{w \sim \mu} [\mathcal{E}]. \quad (3)$$

Let $D = \mathbf{E}_W [\mathcal{U}_{W \cap T}] = \mathbf{E}_{w \sim \mu} [\mathcal{U}_{w \cap T}]$. We show that $\left| \frac{C_s}{C} \cdot D - \mathcal{U}_{s \cap T} \right|_1 \leq \sqrt{\epsilon'}$ for $\epsilon' = 2^{2k+4} (2^{-r} |\mathcal{B}(\epsilon)| + \epsilon)$, as by Claim 5.1 this implies that $|D - \mathcal{U}_{s \cap T}|_1 \leq 2\sqrt{\epsilon'}$, as claimed. It holds that

$$\begin{aligned}
\left(\left| \tfrac{C_s}{C} \cdot D - \mathcal{U}_{s \cap T} \right|_1 \right)^2 &= \left(\sum_{x \in T} \left| \tfrac{C_s}{C} \cdot D(x) - \mathcal{U}_{s \cap T}(x) \right| \right)^2 \\
&\leq |T| \cdot \sum_{x \in T} \left(\tfrac{C_s}{C} \cdot D(x) - \mathcal{U}_{s \cap T}(x) \right)^2 && \text{(Cauchy-Schwartz)} \\
&= |T| \cdot 2^n \cdot \mathop{\mathbf{E}}_{x \in_R \{0,1\}^n} \left[\left(\tfrac{C_s}{C} \cdot D(x) - \mathcal{U}_{s \cap T}(x) \right)^2 \right] \\
&= |T| \cdot 2^n \cdot \sum_{\alpha \in \{0,1\}^n} \left(\tfrac{C_s}{C} \cdot \widehat{D}(\alpha) - \widehat{\mathcal{U}}_{s \cap T}(\alpha) \right)^2. && \text{(Parseval)}
\end{aligned}$$

The rest of the proof is devoted to showing that $\sum_\alpha \left(\frac{C_s}{C} \cdot \widehat{D}(\alpha) - \widehat{\mathcal{U}}_{s \cap T}(\alpha) \right)^2 \leq \frac{\epsilon'}{2^n |T|}$. The proof takes advantage of the fact that for every $x \in \{0,1\}^n$, it holds that $\mathcal{U}_T(x) \in \{0, 1/|T|\}$, thus we have $\mathcal{U}_T(x) \cdot \mathcal{U}_T(x) = \mathcal{U}_T(x)/|T|$. This yields the equality of the Fourier transform of the two sides of the equation. Using the convolution formula,

$$\forall \beta \in \{0,1\}^n : \sum_{\alpha \in \{0,1\}^n} \widehat{\mathcal{U}}_T(\alpha) \cdot \widehat{\mathcal{U}}_T(\alpha + \beta) = \widehat{\mathcal{U}}_T(\beta)/|T|. \quad (4)$$

It holds that

$$\begin{aligned}
&\frac{1}{C_s^2}\sum_{\alpha\in\{0,1\}^n}\left(\frac{C_s}{C}\cdot\widehat{D}(\alpha)-\widehat{\mathcal{U}}_{s\cap T}(\alpha)\right)^2\\
&=\frac{1}{C_s^2}\sum_{\alpha}\left(\frac{C_s}{C}\cdot\mathop{\mathbf{E}}_{w\sim\mu}\left[C_w\cdot\sum_{(\beta,b)\in w^\perp}(-1)^b\cdot\widehat{\mathcal{U}}_T(\alpha+\beta)\right]-C_s\cdot\sum_{(\beta,b)\in s^\perp}(-1)^b\cdot\widehat{\mathcal{U}}_T(\alpha+\beta)\right)^2\\
&\hfill\text{(by Claim (5.2))}\\
&=\frac{1}{C_s^2}\sum_{\alpha}\left(C_s\cdot\mathop{\mathbf{E}}_{w\sim\rho}\left[\sum_{(\beta,b)\in w^\perp}(-1)^b\cdot\widehat{\mathcal{U}}_T(\alpha+\beta)\right]-C_s\cdot\sum_{(\beta,b)\in s^\perp}(-1)^b\cdot\widehat{\mathcal{U}}_T(\alpha+\beta)\right)^2\\
&\hfill(\rho(w)=\mu(w)\cdot C_w/C)\\
&=\sum_{\alpha}\left(\mathop{\mathbf{E}}_{w\sim\rho}\left[\sum_{(\beta,b)\in w^\perp\setminus s^\perp}(-1)^b\cdot\widehat{\mathcal{U}}_T(\alpha+\beta)\right]\right)^2 \qquad (\text{for } w\in\mathrm{supp}(\rho):\ s^\perp\subseteq w^\perp)\\
&=\sum_{\alpha}\mathop{\mathbf{E}}_{\substack{w_1\sim\rho\\ w_2\sim\rho}}\left[\sum_{\substack{(\beta_1,b_1)\in w_1^\perp\setminus s^\perp\\(\beta_2,b_2)\in w_2^\perp\setminus s^\perp}}(-1)^{b_1+b_2}\cdot\widehat{\mathcal{U}}_T(\alpha+\beta_1)\cdot\widehat{\mathcal{U}}_T(\alpha+\beta_2)\right]\\
&=\mathop{\mathbf{E}}_{w_1,w_2}\left[\sum_{(\beta_1,b_1),(\beta_2,b_2)}(-1)^{b_1+b_2}\cdot\sum_{\alpha}\widehat{\mathcal{U}}_T(\alpha+\beta_1)\cdot\widehat{\mathcal{U}}_T(\alpha+\beta_2)\right]\\
&=\mathop{\mathbf{E}}_{w_1,w_2}\left[\sum_{(\beta_1,b_1),(\beta_2,b_2)}(-1)^{b_1+b_2}\cdot\sum_{\alpha}\widehat{\mathcal{U}}_T(\alpha)\cdot\widehat{\mathcal{U}}_T(\alpha+\beta_1+\beta_2)\right]\\
&=\mathop{\mathbf{E}}_{w_1,w_2}\left[\sum_{(\beta_1,b_1),(\beta_2,b_2)}(-1)^{b_1+b_2}\cdot\frac{1}{|T|}\cdot\widehat{\mathcal{U}}_T(\beta_1+\beta_2)\right] \qquad \text{(by Equation (4))}\\
&\le\frac{1}{|T|}\cdot\mathop{\mathbf{E}}_{w_1,w_2}\left[\sum_{(\beta_1,b_1),(\beta_2,b_2)}\left|\widehat{\mathcal{U}}_T(\beta_1+\beta_2)\right|\right] \qquad (5)
\end{aligned}$$

(In the above set of equations, whenever we take an expectation over w_1, w_2, we mean that the expectation is over $w_1\sim\rho, w_2\sim\rho$. Whenever we consider a sum over $(\beta_1,b_1),(\beta_2,b_2)$, we mean that the sum is over $(\beta_1,b_1)\in w_1^\perp\setminus s^\perp, (\beta_2,b_2)\in w_2^\perp\setminus s^\perp$.)

Denote $E=\mathbf{E}_{w_1,w_2}\left[\sum_{(\beta_1,b_1),(\beta_2,b_2)}\left|\widehat{\mathcal{U}}_T(\beta_1+\beta_2)\right|\right]$. We next upper-bound the last expression in Equation (5) by upper-bounding E. Let $\alpha\in\{0,1\}^n$, we bound the coefficient of $\widehat{\mathcal{U}}_T(\alpha)$ in E. Fix $w_1\in\mathrm{supp}(\rho)\subseteq\mathcal{G}$. Since $w_1\in\mathcal{G}$, the co-dimension of w_1 is at most k, thus $|w_1^\perp|\le 2^k$. Therefore, there are at most 2^k options for the selection of the pair (β_1,b_1). Fix $(\beta_1,b_1)\in w_1^\perp\setminus s^\perp$. Let $\beta_2=\alpha-\beta_1$ and $b_2\in\{0,1\}$. The first condition of the lemma and Equation (3) imply that $\Pr_{w_2\sim\rho}\left[(\beta_2,b_2)\in w_2^\perp\right]<3\cdot 2^{-r}$, as long as $(\beta_2,b_2)\notin s^\perp$. Since we are summing only over $(\beta_2,b_2)\notin s^\perp$, we conclude that the coefficient of $\widehat{\mathcal{U}}_T(\alpha)$ in E is smaller than $3 \cdot 2^k \cdot 2^{-r}$.

Recall that $\mathcal{B}(\epsilon) = \left\{\alpha \in \{0,1\}^n : \left|\widehat{\mathcal{U}}_T(\alpha)\right| > 2^{-n}\epsilon\right\}$ is the set of big Fourier coefficients of $\mathcal{U}_T$. For any $\alpha \in \{0,1\}^n$, we have $\left|\widehat{\mathcal{U}}_T(\alpha)\right| \le \left|\widehat{\mathcal{U}}_T(\vec{0})\right| = 2^{-n}$. Therefore, the total contribution of the Fourier coefficients $\widehat{\mathcal{U}}_T(\alpha)$ for $\alpha \in \mathcal{B}(\epsilon)$ to E is smaller than $3 \cdot 2^k \cdot 2^{-r}|B(\epsilon)| \cdot 2^{-n}$. The total contribution of the Fourier coefficients $\widehat{\mathcal{U}}_T(\alpha)$ for $\alpha \in \{0,1\}^n \setminus \mathcal{B}(\epsilon)$ to E is at most $2^{2k} \cdot 2^{-n}\epsilon$, as then $\left|\widehat{\mathcal{U}}_T(\alpha)\right| \le 2^{-n}\epsilon$ and $|w_1^\perp|, |w_2^\perp| \le 2^k$. Hence, by Equation (5), the fact that $s \in \mathcal{G}$ and Claim 5.3,

$$\sum_{\alpha \in \{0,1\}^n} \left(\frac{C_s}{C} \cdot \widehat{D}(\alpha) - \widehat{\mathcal{U}}_{s\cap T}(\alpha)\right)^2 < C_s^2 \cdot \frac{3 \cdot 2^k \cdot 2^{-r}|B(\epsilon)| + 2^{2k}\epsilon}{2^n|T|}$$
$$\le 3 \cdot 2^2 \cdot \frac{2^{2k}(2^{-r}|\mathcal{B}(\epsilon)| + \epsilon)}{2^n|T|}.$$

□

Lemma 5.5. *Let $W \in \mathcal{G}$ be a random variable. Let $\epsilon \in (0,1)$ and $r = \log(|\mathcal{B}(\epsilon)|/\epsilon)$. Then, there exists an affine subspace $s \in \mathcal{G}$ of co-dimension $k' \le k$, such that:*

1. $\Pr_W[W \subseteq s] \ge 2^{-rk'}$.
2. $\left|\mathbf{E}_{W|(W\subseteq s)}[\mathcal{U}_{W\cap T}] - \mathcal{U}_{s\cap T}\right|_1 < 2^{k+4}\sqrt{\epsilon}$.

Proof. We apply an iterative process to define a sequence of affine subspaces $s_0 \supseteq s_1 \supseteq s_2 \supseteq \ldots \supseteq s_{k'} = s$ where $0 \le k' \le k$ and such that each subspace s_i is of co-dimension i. We start with $s_0 = \{0,1\}^n$. For $i = 1, \ldots, k$ we check if there exist $a_i \in \{0,1\}^n$ and $b_i \in \{0,1\}$ such that $(a_i, b_i) \notin (s_{i-1})^\perp$ and

$$\Pr_W[\forall x \in W : a_i \cdot x = b_i \;\mid\; W \subseteq s_{i-1}] \ge 2^{-r}.$$

If this is the case, then we take $s_i = \{x \in s_{i-1} : a_i \cdot x = b_i\}$. Indeed, if s_{i-1} is an affine subspace of co-dimension $i-1$, then s_i is an affine subspace of co-dimension i. If this is not the case (that is, for every $(a_i, b_i) \in (\{0,1\}^n \times \{0,1\}) \setminus (s_{i-1})^\perp$ it holds that $\Pr_W[\forall x \in W : a_i \cdot x = b_i \mid W \subseteq s_{i-1}] < 2^{-r}$), then we halt and take $k' = i-1$ and $s = s_{k'}$. If the process did not halt after these k iterations, we take $k' = k$ and $s = s_{k'}$.

Proving the first property. We prove by induction that for every $0 \le i \le k'$ it holds that $\Pr_W[W \subseteq s_i] \ge 2^{-ri}$. For $i = 0$, it holds that $\Pr_W[W \subseteq s_0] = 1$. For $i \ge 1$ we have

$$\Pr_W[W \subseteq s_i] = \Pr_W[W \subseteq s_{i-1}] \cdot \Pr_W[\forall x \in W : a_i \cdot x = b_i \mid W \subseteq s_{i-1}] \ge 2^{-r(i-1)} \cdot 2^{-r} = 2^{-ri}.$$

Proving that $s \in \mathcal{G}$. Since $k' \le k$, the subspace s is of co-dimension at most k. Let $w \in \mathrm{supp}(W)$ be such that $w \subseteq s$. Such a w exists as $\Pr_W[W \subseteq s] \ge 2^{-rk'} > 0$. Since $w \subseteq s$, $s^\perp \subseteq w^\perp$ and since $w \in \mathrm{supp}(W) \subseteq \mathcal{G}$, it also holds that $s \in \mathcal{G}$.

Proving the second property. We first consider the case $k' = k$. In such a case we claim that the statistical distance is 0. This is true since $s = s_k$ is a subspace of co-dimension k. Since $W \in \mathcal{G}$ it is always of co-dimension at most k. Hence, the random variable $(W \mid W \subseteq s)$ must attain the subspace s with probability 1. Conclude that $\mathbf{E}_{W|(W \subseteq s)}[\mathcal{U}_{W \cap T}] = \mathcal{U}_{s \cap T}$.

Next, we consider the case $k' < k$. In this case we know that for all $(a, b) \in (\{0,1\}^n \times \{0,1\}) \setminus s^{\perp}$ we have $\Pr_W[\forall x \in W : \ a \cdot x = b \mid W \subseteq s] < 2^{-r}$. To finish the proof, we apply Lemma 5.4 with the s, r, ϵ defined above and with $W = W|(W \subseteq s)$. Using the choice of $r = \log(|\mathcal{B}(\epsilon)|/\epsilon)$, we get that $\left|\mathbf{E}_{W|(W\subseteq s)}[\mathcal{U}_{W\cap T}] - \mathcal{U}_{s \cap T}\right|_1 \le 2^{k+3}\sqrt{2^{-r}|\mathcal{B}(\epsilon)| + \epsilon} \le 2^{k+4}\sqrt{\epsilon}$, as required. □

The next lemma is the main result of this section.

Lemma 5.6. *Let $W \in \mathcal{A}(n)$ be a random variable. Let $\epsilon \in (0,1)$ and $r = \log(|\mathcal{B}(\epsilon)|/\epsilon)$. There exists a partial function $\sigma : \mathcal{A}(n) \to \mathcal{A}(n)$, such that:*

1. $\Pr_W[W \notin \text{domain}(\sigma)] \le 2^{-2n}$.
2. *For every $w \in \text{domain}(\sigma)$, $w \subseteq \sigma(w)$.*
3. *For every $s \in \text{image}(\sigma)$,*
$$\left| \underset{W|(\sigma(W)=s)}{\mathbf{E}} [\mathcal{U}_{W \cap T}] - \mathcal{U}_{s\cap T} \right|_1 < 2^{k+4}\sqrt{\epsilon}.$$
4. *For every $k' \le k$, there are at most*
$$n \cdot 2^{rk'+1}$$
elements $s \in \text{image}(\sigma) \cap \mathcal{G}$, with $\text{codim}(s) \le k'$.

Proof. For any $w \in \mathcal{A}(n) \setminus \mathcal{G}$ we map w to itself, i.e., $\sigma(w) = w$. To map subspaces in $\mathcal{G}$ we repeatedly apply Lemma 5.5. We start with the random variable $W_0 = W \mid (W \in \mathcal{G})$, and apply Lemma 5.5 on W_0. We obtain a subspace s_0 (the subspace s whose existence is guaranteed by Lemma 5.5). For every $w \subseteq s_0$, we define $\sigma(w) = s_0$.

We then define the random variable $W_1 = W_0 \mid (W_0 \not\subseteq s_0)$, and apply Lemma 5.5 on W_1. We obtain a subspace s_1 (the subspace s whose existence is guaranteed by Lemma 5.5). For every $w \subseteq s_1$ on which σ was still not defined, we define $\sigma(w) = s_1$.

In the same way, in Step i, we define the random variable $W_i = W_{i-1} \mid (W_{i-1} \not\subseteq s_{i-1})$. Note that $W_i = W \mid (W \in \mathcal{G}) \wedge (W \not\subseteq s_0) \wedge \ldots \wedge (W \not\subseteq s_{i-1})$, that is, W_i is the restriction of W to the part of $\mathcal{A}(n)$ where σ was still not defined. We apply Lemma 5.5 on W_i and obtain a subspace s_i (the subspace s whose existence is guaranteed by Lemma 5.5). For every $w \subseteq s_i$ on which σ was still not defined, we define $\sigma(w) = s_i$.

We repeat this until $\Pr_W[W \notin \text{domain}(\sigma)] \le 2^{-2n}$.

By Lemma 5.5, for every i it holds that $s_i \in \mathcal{G}$. Note that for $i' < i$, $s_{i'} \ne s_i$, because the support of W_i doesn't contain any element $w \subseteq s_{i'}$. Hence, the subspaces $s_0, s_1, \ldots$ are all different.

It remains to show that the properties in the statement of the lemma hold:

1. The first property is obvious because we continue to define σ on more and more elements repeatedly, until the first property holds.

2. The second property is obvious because we mapped w to s_i only if $w \subseteq s_i$.

3. The third property holds for $s \in \text{image}(\sigma) \cap \mathcal{G}$, by the second property guaranteed by Lemma 5.5. For $s \in \text{image}(\sigma) \setminus \mathcal{G}$, the property trivially holds as the random variable $W \mid (\sigma(W) = s)$ is supported on the singleton $\{s\}$.

4. The fourth property holds because by the first property guaranteed by Lemma 5.5, in each step where we obtain a subspace s_i of co-dimension at most k', we define σ on a fraction of at least $2^{-rk'}$ of the space that still remains. Thus, after at most $2n \cdot 2^{rk'}$ such steps we have $\Pr[W \notin \text{domain}(\sigma)] \leq (1 - 2^{-rk'})^{2n \cdot 2^{rk'}} \leq 2^{-2n}$, and we stop. Thus, the number of elements s_i, of co-dimension at most k', that we obtain in the process, is at most $2n \cdot 2^{rk'}$.

□

6 Branching Programs for Parity Learning

Recall that in the problem of parity learning, there is a string $x \in T$ that was chosen uniformly at random. A learner tries to learn x from a stream of samples, $(a_1, b_1), (a_2, b_2) \ldots$, where each a_t is uniformly distributed over $\{0,1\}^n$ and for every t, $b_t = a_t \cdot x$.

6.1 General Branching Programs for Parity Learning

In the following definition, we model the learner by a *branching program.*

Definition 6.1. Branching Program for Parity Learning: *A branching program of length m and width d, for parity learning, is a directed (multi) graph with vertices arranged in $m+1$ layers containing at most d vertices each. In the first layer, that we think of as layer 0, there is only one vertex, called the start vertex. A vertex of outdegree 0 is called a leaf. All vertices in the last layer are leaves (but there may be additional leaves). Every non-leaf vertex in the program has 2^{n+1} outgoing edges, labeled by elements $(a,b) \in \{0,1\}^n \times \{0,1\}$, with exactly one edge labeled by each such (a,b), and all these edges going into vertices in the next layer. Each leaf v in the program is labeled by a vector $\tilde{x}(v) \in \{0,1\}^n$, that we think of as the output of the program on that leaf.*

Computation-Path: *The samples $(a_1, b_1), \ldots, (a_m, b_m) \in \{0,1\}^n \times \{0,1\}$ that are given as input, define a computation-path in the branching program, by starting from the start vertex and following at Step t the edge labeled by (a_t, b_t), until reaching a leaf. The program outputs the label $\tilde{x}(v)$ of the leaf v reached by the computation-path.*

Success Probability: *The success probability of the program is the probability that $\tilde{x} = x$, where $\tilde{x}$ is the vector that the program outputs, and the probability is over $x, a_1, \ldots, a_m$ (where x is uniformly distributed over T, and $a_1, \ldots, a_m$ are uniformly distributed over $\{0,1\}^n$, and for every t, $b_t = a_t \cdot x$).*

6.2 Affine Branching Programs for Parity Learning

Next, we define *affine branching programs for parity learning.* In an affine branching program for parity learning, every vertex v is labeled by an affine subspace $w(v) \in \mathcal{A}(n)$. We will have the property that if the computation-path reaches v then $x \in w(v)$. Thus, we can interpret $w(v)$ as an affine subspace that is known to contain x.

Definition 6.2. Affine Branching Program for Parity Learning: *A branching program for parity learning is affine if each vertex v in the program is labeled by an affine subspace $w(v) \in \mathcal{A}(n)$, and the following properties hold:*

1. **Start vertex:** *The start vertex is labeled by the space $\{0,1\}^n \in \mathcal{A}(n)$.*

2. **Soundness:** *For an edge $e = (u, v)$, labeled by (a, b), denote*
$$w(e) = w(u) \cap \{x' \in \{0,1\}^n : a \cdot x' = b\}.$$
Then,
$$w(e) \subseteq w(v).$$

Given an affine branching program for parity learning, and samples $(a_1, b_1), \ldots, (a_m, b_m)$, such that, for every t, $b_t = a_t \cdot x$, it follows by induction that for every vertex v in the program, if the computation-path reaches v then $x \in w(v)$.

We remark that an affine branching program is a branching program, and as such is supposed to have an output. However, in what follows, we ignore this output, and focus instead on the affine subspace $w(v) \in \mathcal{A}(n)$ that labels the leaf v reached by the computation-path. For this reason, we sometimes define affine branching programs without specifying the output.

6.3 Accurate Affine Branching Programs for Parity Learning

For a vertex v in a branching program for parity learning, we denote by $\mathbb{P}_{x|v}$ the distribution of the random variable x, conditioned on the event that the vertex v was reached by the computation-path.

Definition 6.3. ϵ-Accurate Affine Branching Program for Parity Learning: *An affine branching program of length m for parity learning is ϵ-accurate if all the leaves are in the last layer, and the following additional property holds (where x is uniformly distributed over T, and $a_1, \ldots, a_m$ are uniformly distributed over $\{0,1\}^n$, and for every t, $b_t = a_t \cdot x$):*

3. **Accuracy:** *Let $0 \leq t \leq m$. Let V_t be the vertex in layer t, reached by the computation-path. Let y_t be a random variable uniformly distributed over the set $w(V_t) \cap T$. Then,*
$$\left|\mathbb{P}_{V_t,x} - \mathbb{P}_{V_t,y_t}\right|_1 \leq \epsilon,$$
or, equivalently,
$$\mathop{\mathbf{E}}_{V_t} \left|\mathbb{P}_{x|V_t} - \mathcal{U}_{w(V_t)\cap T}\right|_1 \leq \epsilon.$$

7 From Branching Programs to Affine Branching Programs

Recall that throughout this section, we think of the set $T \subseteq \{0,1\}^n$ and the maximal co-dimension $k \in \mathbb{N}$ as fixed (see Section 4.4.1). In this section, we show that any branching program B for parity learning can be simulated by an affine branching program P for parity learning. Roughly speaking, each vertex of the simulated program B will be represented by a set of vertices of the simulating program P. Note that the width of P will typically be significantly larger than the width of B.

More precisely, a branching program B for parity learning is simulated by a branching program P for parity learning if there exists a mapping Γ from the vertices of P to the vertices of B, and the following properties hold:

1. **Preservation of structure:** For every i, Γ maps layer i of P to layer i of B. Moreover, Γ maps leaves to leaves and non-leaf vertices to non-leaf vertices. Note that Γ is not necessarily one-to-one.

2. **Preservation of functionality:** For every edge (u,v), labeled by (a,b), in P, there is an edge $(\Gamma(u),\Gamma(v))$, labeled by (a,b), in B.

Lemma 7.1. *Assume that there exists a length m and width d branching program B for parity learning, such that: all leaves of B are in the last layer, and the success probability of B is β.*

Let $\epsilon \in (2^{-4n}, 1)$ and $r = \log(|\mathcal{B}(\epsilon)|/\epsilon)$. Let $\epsilon' = 2^{k+6}m\sqrt{\epsilon}$. Then, there exists an ϵ'-accurate length m affine branching program P for parity learning, such that:

1. *For every $k' \leq k$, the number of vertices in P, that are labeled with an affine subspace in $\mathcal{G}$ of co-dimension k', is at most*
$$n \cdot 2^{rk'+1} \cdot dm.$$

2. *The probability that the leaf reached by the computation-path of P is labeled by a subspace in $\mathcal{A}(n) \setminus \mathcal{G}$ is at least*
$$\beta - \epsilon' - \frac{2^{k+1}}{|T|}.$$

Proof. For every $0 \leq j \leq m$, let $\epsilon_j = 2^{k+6}j\sqrt{\epsilon}$. We will use Lemma 5.6 to turn, inductively, the layers of B, one by one, into layers of an ϵ'-accurate affine branching program, P. In Step j of the induction, we will turn layer j of B into layer j of P, and define the label $w(v) \in \mathcal{A}(n)$ for every vertex v in that layer of P. Formally, we will construct, inductively, a sequence of programs $B, P_0, \ldots, P_m = P$, where each program is of length m, and for every j, the program P_j differs from the previous program only in layer j (and in the edges going into layer j and out of layer j). After Step j of the induction, we will have a branching program P_j, such that, layers 0 to j of P_j form an affine branching program for parity learning. In addition, the following inductive hypothesis will hold:

Inductive Hypothesis:

Let $\mathcal{L}_j$ be the set of vertices in layer j of P_j. Let V_j be the vertex in $\mathcal{L}_j$, reached by the computation-path of P_j. Note that V_j is a random variable that depends on $x, a_1, \ldots, a_j$ (and recall that x is uniformly distributed over T, and $a_1, \ldots, a_m$ are uniformly distributed over $\{0,1\}^n$, and for every t, $b_t = a_t \cdot x$). The inductive hypothesis is that there exists a random variable U_j over $\mathcal{L}_j$, such that, if y_j is a random variable uniformly distributed over the set $w(U_j) \cap T$, then

$$\left|\mathbb{P}_{V_j,x} - \mathbb{P}_{U_j,y_j}\right|_1 \leq \tfrac{\epsilon_j}{2}. \tag{6}$$

The inductive hypothesis is equivalent to the *accuracy* requirement (see Definition 6.3) for layer j of P_j, up to a small multiplicative constant in the accuracy, but we need to assume it in this slightly different form, in order to avoid deteriorating the accuracy by a multiplicative factor in each step of the induction.

Base Case:

In the base case of the induction, $j = 0$, we define P_0 by just labeling the start vertex of B by $\{0,1\}^n \in \mathcal{A}(n)$. Thus, the *start vertex* property in the definition of an affine branching program is satisfied. The *soundness* property is trivially satisfied because the restriction of P_0 to layer 0 contains no edges. Since we always start from the start vertex, the distribution of the random variable x, conditioned on the event that we reached the start vertex, is just $\mathcal{U}_T$, and hence the inductive hypothesis (Equation (6)) holds with $U_0 = V_0$.

Inductive Step:

Assume that we already turned layers 0 to $j-1$ of B into layers 0 to $j-1$ of P. That is, we already defined the program P_{j-1}, and layers 0 to $j-1$ of P_{j-1} satisfy the *start vertex* property, the *soundness* property, and the inductive hypothesis (Equation (6)). We will now show how to define P_j from P_{j-1}, that is, how to turn layer j of B into layer j of P.

Let $U_{j-1} \in \mathcal{L}_{j-1}$ be the random variable that satisfies the inductive hypothesis (Equation (6)) for layer $j-1$ of P_{j-1}. Let y_{j-1} be a random variable uniformly distributed over the set $w(U_{j-1}) \cap T$. Let $a \in_R \{0,1\}^n$. Let $b = a \cdot y_{j-1}$. Let $E = (U_{j-1}, V)$ be the edge labeled by (a,b) outgoing U_{j-1} in P_{j-1}. Thus, V is a vertex in layer j of P_{j-1}. Let $W = w(E)$, where $w(E)$ is defined as in the *soundness* property in Definition 6.2. That is,

$$w(E) = w(U_{j-1}) \cap \{x' \in \{0,1\}^n : a \cdot x' = b\},$$

where (a,b) is the label of E, and $w(U_{j-1})$ is the label of U_{j-1} in P_{j-1}.

Let v be a vertex in layer j of P_{j-1} (and note that v is also a vertex in layer j of B). Let

$$W_v = W|(V = v).$$

Let $\sigma_v : \mathcal{A}(n) \to \mathcal{A}(n)$ be the partial function whose existence is guaranteed by Lemma 5.6, when applied on the random variable W_v. Extend $\sigma_v : \mathcal{A}(n) \to \mathcal{A}(n)$ so that it outputs the special value $*$ on every element where it was previously undefined.

In the program P_j, we will split the vertex v into $|\text{image}(\sigma_v)|$ vertices (where $\text{image}(\sigma_v)$ already contains the additional special value $*$). For every $s \in \text{image}(\sigma_v)$, we will have a vertex (v,s). If $s \neq *$, we label the vertex (v,s) by the affine subspace s, and we label the additional vertex $(v,*)$ by $\{0,1\}^n$. For every $s \in \text{image}(\sigma_v)$, the edges going out of (v,s) (in P_j) will be the same as the edges going out of v in P_{j-1}. That is, for every edge (v,v') (from layer j to layer $j+1$) in the program P_{j-1}, and every $s \in \text{image}(\sigma_v)$, we will have an edge $((v,s),v')$ with the same label, (from layer j to layer $j+1$) in the program P_j.

We will now define the edges going into the vertices (v,s) in the program P_j. For every edge $e=(u,v)$, labeled by (a,b), (from layer $j-1$ to layer j), in the program P_{j-1}, consider the affine subspace $w = w(e) = w(u) \cap \{x' \in \{0,1\}^n : a \cdot x' = b\}$ (as in the *soundness* property in Definition 6.2), where $w(u)$ is the label of u in P_{j-1}. Let $s = \sigma_v(w)$.

In P_j, we will have the edge $(u,(v,s))$ (labeled by (a,b)), from layer $j-1$ to layer j, that is, we connect u to (v,s). Note that the edge $(u,(v,s))$ satisfies the *soundness* property in the definition of an affine branching program: If $s \neq *$, the vertex (v,s) is labeled by $s = \sigma_v(w)$ and by Poperty 2 of Lemma 5.6, $w \subseteq \sigma_v(w)$. If $s = *$, the vertex (v,s) is labeled by $\{0,1\}^n$ and hence the *soundness* property is trivially satisfied.

Proof of the Inductive Hypothesis:

Next, we will prove the inductive hypothesis (Equation (6)), for P_j. We will define the random variable $U_j \in \mathcal{L}_j$ as follows:

As before, let $U_{j-1} \in \mathcal{L}_{j-1}$ be the random variable that satisfies the inductive hypothesis (Equation (6)) for layer $j-1$ of P_{j-1}. Let y_{j-1} be a random variable uniformly distributed over the set $w(U_{j-1}) \cap T$. Let $a \in_R \{0,1\}^n$. Let $b = a \cdot y_{j-1}$. Let $E = (U_{j-1}, V)$ be the edge labeled by (a,b) outgoing U_{j-1} in P_{j-1}. Thus, V is a vertex in layer j of P_{j-1}. As before, let $W = w(E) = w(U_{j-1}) \cap \{x' \in \{0,1\}^n : a \cdot x' = b\}$. As before, for a vertex v in layer j of P_{j-1}, let $\sigma_v : \mathcal{A}(n) \to \mathcal{A}(n)$ be the partial function whose existence is guaranteed by Lemma 5.6, when applied on the random variable $W_v = W|(V=v)$, and extend $\sigma_v : \mathcal{A}(n) \to \mathcal{A}(n)$ so that it outputs the special value $*$ on every element where it was previously undefined.

We define $U_j = (V, \sigma_V(W)) \in \mathcal{L}_j$. Let y_j be a random variable uniformly distributed over the set $w(U_j) \cap T$, and let V_j be the vertex in $\mathcal{L}_j$, reached by the computation-path of P_j. We need to prove that

$$\left|\mathbb{P}_{V_j,x} - \mathbb{P}_{U_j,y_j}\right|_1 \leq 2^{k+5} j \sqrt{\epsilon}. \tag{7}$$

Let y'_j be a random variable uniformly distributed over the set $W \cap T$. Equation (7) follows by the following two equations and by the triangle inequality:

$$\left|\mathbb{P}_{U_j,y'_j} - \mathbb{P}_{U_j,y_j}\right|_1 \leq 2^{k+5} \sqrt{\epsilon}. \tag{8}$$

$$\left|\mathbb{P}_{V_j,x} - \mathbb{P}_{U_j,y'_j}\right|_1 \leq 2^{k+5} (j-1) \sqrt{\epsilon}. \tag{9}$$

Thus, it is sufficient to prove Equation (8) and Equation (9). We will start with Equation (8).

By Property 3 of Lemma 5.6, for every v in layer j of P_{j-1}, and every $s \in \text{image}(\sigma_v) \setminus \{*\}$,

$$\left| \mathop{\mathbf{E}}_{W|(V=v),(\sigma_v(W)=s)} [\mathcal{U}_{W \cap T}] - \mathcal{U}_{s \cap T} \right|_1 < 2^{k+4} \sqrt{\epsilon}.$$

By the definitions of y'_j and U_j,

$$\mathop{\mathbf{E}}_{W|(V=v),(\sigma_v(W)=s)} [\mathcal{U}_{W \cap T}] = \mathop{\mathbf{E}}_{W|(U_j=(v,s))} [\mathcal{U}_{W \cap T}] = \mathbb{P}_{y'_j|(U_j=(v,s))}.$$

By the definition of y_j,

$$\mathcal{U}_{s\cap T} = \mathbb{P}_{y_j|(U_j=(v,s))}$$

Hence

$$\left|\mathbb{P}_{y'_j|(U_j=(v,s))} - \mathbb{P}_{y_j|(U_j=(v,s))}\right|_1 < 2^{k+4}\sqrt{\epsilon}.$$

Taking expectation over U_j, and taking into account that, by Property 1 of Lemma 5.6, for every v, $\Pr(\sigma_v(W) = *) \leq 2^{-2n}$, we obtain

$$\left|\mathbb{P}_{U_j,y'_j} - \mathbb{P}_{U_j,y_j}\right|_1 = \mathop{\mathbf{E}}_{U_j}\left|\mathbb{P}_{y'_j|U_j} - \mathbb{P}_{y_j|U_j}\right|_1 < 2^{k+4}\sqrt{\epsilon} + 2\cdot 2^{-2n},$$

which proves Equation (8) as $\epsilon > 2^{-4n}$.

We will now prove Equation (9). Let $\mathcal{T}$ be the following probabilistic transformation from $\mathcal{L}_{j-1}\times\{0,1\}^n$ to $\mathcal{L}_j\times\{0,1\}^n$. Given $(u,z)\in\mathcal{L}_{j-1}\times\{0,1\}^n$, the transformation $\mathcal{T}$ chooses $a\in_R\{0,1\}^n$ and $b = a\cdot z$, and outputs (V,z), where $V\in\mathcal{L}_j$ is the vertex obtained by following the edge labeled by (a,b) outgoing u in P_j.

By the definition of the computation-path, $\mathcal{T}(V_{j-1},x)$ has the same distribution as (V_j,x). By the definition of U_j, y_j, y'_j, we have that $\mathcal{T}(U_{j-1},y_{j-1})$ has the same distribution as (U_j,y'_j). Hence, by the triangle inequality and the inductive hypothesis,

$$\left|\mathbb{P}_{V_j,x} - \mathbb{P}_{U_j,y'_j}\right|_1 = \left|\mathbb{P}_{\mathcal{T}(V_{j-1},x)} - \mathbb{P}_{\mathcal{T}(U_{j-1},y_{j-1})}\right|_1 \leq \left|\mathbb{P}_{V_{j-1},x} - \mathbb{P}_{U_{j-1},y_{j-1}}\right|_1 \leq 2^{k+5}(j-1)\sqrt{\epsilon},$$

which gives Equation (9).

Since, by induction, layers 0 to $j-1$ of P_{j-1} form an affine branching program for parity learning, and since we already saw that all the edges between layer $j-1$ and layer j of P_j satisfy the *soundness* property in the definition of an affine branching program, we have that layers 0 to j of P_j form an affine branching program for parity learning.

P is ϵ-Accurate:

We will now prove that the final branching program $P = P_m$, that we obtained, satisfies the requirements of the lemma. We already know that P is an affine branching program for parity learning.

We will start by proving that P is ϵ'-accurate. Let $0\leq t\leq m$. Let V_t be the vertex in layer t of P, reached by the computation-path of P. Let z_t be a random variable uniformly distributed over the set $w(V_t)\cap T$, We need to prove that,

$$\left|\mathbb{P}_{V_t,x} - \mathbb{P}_{V_t,z_t}\right|_1 \leq \epsilon'. \tag{10}$$

Recall that by the inductive hypothesis (Equation (6)), there exists a random variable U_t over layer t of P, such that, if y_t is a random variable uniformly distributed over the set $w(U_t)\cap T$, then

$$\left|\mathbb{P}_{V_t,x} - \mathbb{P}_{U_t,y_t}\right|_1 \leq \tfrac{\epsilon'}{2}, \tag{11}$$

and this also implies

$$\left|\mathbb{P}_{V_t} - \mathbb{P}_{U_t}\right|_1 \leq \tfrac{\epsilon'}{2}.$$

By the last inequality and since for every v in layer t of P, it holds that $\mathbb{P}_{z_t|(V_t=v)} = \mathbb{P}_{y_t|(U_t=v)}$ (since they are both uniformly distributed over $w(v)\cap T$), we have

$$\left|\mathbb{P}_{V_t,z_t} - \mathbb{P}_{U_t,y_t}\right|_1 = \left|\mathbb{P}_{V_t} - \mathbb{P}_{U_t}\right|_1 \leq \tfrac{\epsilon'}{2}. \tag{12}$$

Equation (10) follows by Equation (11), Equation (12) and the triangle inequality.

P Satisfies the Additional Properties:

We will now prove that P satisfies the two additional properties claimed in the statement of the lemma. The first property holds since Property 4 of Lemma 5.6 ensures that for every vertex in layers 1 to m of the branching program B, we obtain at most $n \cdot 2^{rk'+1}$ vertices in the branching program P that are labeled with affine subspaces in $\mathcal{G}$ of co-dimension k'.

It remains to prove the second property. Let $V_m = (V, S)$ be the vertex in layer m of P, reached by the computation-path of P. Note that V_m is a random variable that depends on $x, a_1, \dots, a_m$ (and recall that x is uniformly distributed over T, and $a_1, \dots, a_m$ are uniformly distributed over $\{0,1\}^n$, and for every t, $b_t = a_t \cdot x$).

Note that V is the vertex in layer m of B, reached by the computation-path of B (on the same $x, a_1, \dots, a_m$). This is true since P simulates B. More precisely, by the construction, if on $x, a_1, \dots, a_m$, the program P reaches (V, S), then, on the same $x, a_1, \dots, a_m$, the program B reaches V.

Since the success probability of B is β,

$$\Pr[\tilde{x}(V) = x] = \beta,$$

where $\tilde{x}(V)$ is the label of V in B. Let y_m be a random variable uniformly distributed over the set $w(V_m) \cap T$, where $w(V_m)$ is the label of V_m in P. Since P is ϵ'-accurate,

$$|\mathbb{P}_{V,x} - \mathbb{P}_{V,y_m}|_1 \le |\mathbb{P}_{V,S,x} - \mathbb{P}_{V,S,y_m}|_1 = |\mathbb{P}_{V_m,x} - \mathbb{P}_{V_m,y_m}|_1 \le \epsilon'.$$

Thus,

$$\Pr[y_m = \tilde{x}(V)] \ge \Pr[x = \tilde{x}(V)] - \epsilon' = \beta - \epsilon'.$$

If $w(V_m) \in \mathcal{G}$ then, by Claim 5.3, it holds that $C_{w(V_m)} \in (2/3, 2)$. In addition, $\mathrm{codim}(w(V_m)) \le k$, thus $|w(V_m)| \ge 2^{n-k}$. Therefore,

$$|w(V_m) \cap T| = \frac{|w(V_m)| \cdot |T|}{2^n} \cdot \frac{1}{C_{w(V_m)}} \ge 2^{-n-1} \cdot |w(V_m)| \cdot |T| \ge 2^{-k-1} \cdot |T|.$$

Since y_m is uniformly distributed over the set $w(V_m) \cap T$, we get that

$$\beta - \epsilon' \le \Pr[y_m = \tilde{x}(V)] \le \frac{2^{k+1}}{|T|} + \Pr[w(V_m) \in \mathcal{A}(n) \setminus \mathcal{G}].$$

That is,

$$\Pr[w(V_m) \in \mathcal{A}(n) \setminus \mathcal{G}] \ge \beta - \epsilon' - \frac{2^{k+1}}{|T|}.$$

$\square$

8 Bounds for Affine Branching Program

Recall that throughout this section, we think of the set $T \subseteq \{0,1\}^n$ and the maximal co-dimension $k \in \mathbb{N}$ as fixed (see Section 4.4.1).

Lemma 8.1. *Let P be a length m affine branching program for parity learning, such that, for every vertex u of P, $\mathrm{codim}(w(u)) \le k$. Let v be a vertex of P, such that, $\mathrm{codim}(w(v)) = k$. Then, the probability that the computation-path of P reaches v is at most*

$$m^k \cdot 2^{-k(n-2k)}.$$

Proof. Let $s = (w(v)^{\perp})_1$. That is,

$$s = \{a \in \{0,1\}^n : \exists b \in \{0,1\} \; \forall x' \in w(v) : a \cdot x' = b\}.$$

Let $V_0, \ldots, V_m$ be the vertices on the computation-path of P. Note that $V_0, \ldots, V_m$ are random variables that depend on $x, a_1, \ldots, a_m$. For every $0 \le i \le m$, let $S_i = (w(V_i)^{\perp})_1$. That is,

$$S_i = \{a \in \{0,1\}^n : \exists b \in \{0,1\} \; \forall x' \in w(V_i) : a \cdot x' = b\}.$$

By the *soundness* property in Definition 6.2, for every $1 \le i \le m$,

$$S_i \subseteq \operatorname{span}(S_{i-1} \cup \{a_i\}). \tag{13}$$

For every $0 \le i \le m$, let $Z_i = \dim(S_i \cap s)$. Note that $Z_0 = 0$, and by Equation (13), for every $1 \le i \le m$, $Z_i \le Z_{i-1} + 1$. If the computation-path of P reaches v then for some $1 \le i \le m$, $Z_i = k$. Thus, if the computation-path of P reaches v, there exist k indices $i_1 < \ldots < i_k \in [m]$, such that, the following event, denoted by $E_{i_1,\ldots,i_k}$, occurs:

$$E_{i_1,\ldots,i_k} = \bigwedge_{j \in [k]} (Z_{i_j - 1} = j - 1) \wedge (Z_{i_j} = j).$$

(In particular, $E_{i_1,\ldots,i_k}$ occurs if for every j, we take i_j to be the first i such that $Z_i = j$). We will bound the probability that the computation-path of P reaches v, by bounding $\Pr[E_{i_1,\ldots,i_k}]$, and taking the union bound over (less than) m^k possibilities for $i_1, \ldots, i_k \in [m]$.

Fix $i_1 < \ldots < i_k \in [m]$. For $r \in \{0, \ldots, k\}$, let

$$E_{i_1,\ldots,i_r} = \bigwedge_{j \in [r]} (Z_{i_j - 1} = j - 1) \wedge (Z_{i_j} = j).$$

Thus,

$$\Pr[E_{i_1,\ldots,i_k}] = \prod_{j \in [k]} \Pr[E_{i_1,\ldots,i_j} \mid E_{i_1,\ldots,i_{j-1}}].$$

We will show how to bound $\Pr[E_{i_1,\ldots,i_j} \mid E_{i_1,\ldots,i_{j-1}}]$.

$$\begin{aligned}
\Pr[E_{i_1,\ldots,i_j} \mid E_{i_1,\ldots,i_{j-1}}] &= \Pr[(Z_{i_j - 1} = j - 1) \wedge (Z_{i_j} = j) \mid E_{i_1,\ldots,i_{j-1}}] \\
&= \Pr[(Z_{i_j - 1} = j - 1) \wedge (Z_{i_j - 1} < Z_{i_j}) \mid E_{i_1,\ldots,i_{j-1}}] \\
&\le \Pr[(Z_{i_j - 1} < Z_{i_j}) \mid E_{i_1,\ldots,i_{j-1}} \wedge (Z_{i_j - 1} = j - 1)]. \qquad (14)
\end{aligned}$$

Note that the event $E_{i_1,\ldots,i_{j-1}} \wedge (Z_{i_j - 1} = j - 1)$ that we condition on, on the right hand side, depends only on $x, a_1, \ldots, a_{i_j - 1}$. We will bound the probability for the event $(Z_{i_j - 1} < Z_{i_j})$, conditioned on any event that depends only on $x, a_1, \ldots, a_{i_j - 1}$.

More generally, fix $1 \le i \le m$, and let E'_i be the event $(Z_{i-1} < Z_i)$. Let E' be any event that depends only on $x, a_1, \ldots, a_{i-1}$. Without loss of generality, we can assume that the event E' just fixes the values of $x, a_1, \ldots, a_{i-1}$. We will show how to bound $\Pr[E'_i \mid E']$.

Thus, we fix $x, a_1, \ldots, a_{i-1}$ and we will bound $\Pr[E'_i]$ (conditioned on $x, a_1, \ldots, a_{i-1}$). By Equation (13), if E'_i occurs then $\dim(S_{i-1} \cap s) < \dim(S_i \cap s) \le \dim(\operatorname{span}(S_{i-1} \cup \{a_i\}) \cap s)$, and hence $S_{i-1} \cap s \subsetneq \operatorname{span}(S_{i-1} \cup \{a_i\}) \cap s$. Thus, there exists $a' \in s$ such that $a' \in \operatorname{span}(S_{i-1} \cup \{a_i\})$ but $a' \notin S_{i-1}$, which implies that $a' = a_i \oplus a$ for some $a \in S_{i-1}$. In other words, $a_i = a \oplus a'$

where $a \in S_{i-1}$ and $a' \in s$, i.e., $a_i \in \operatorname{span}(s \cup S_{i-1})$. The event $a_i \in \operatorname{span}(s \cup S_{i-1})$ occurs with probability at most $2^{\dim(s)+\dim(S_{i-1})-n} \leq 2^{2k-n}$ (since a_i is uniformly distributed and independent of $x, a_1, \ldots, a_{i-1}$). Thus,

$$\Pr[E'_i \mid E'] \leq 2^{2k-n}.$$

In particular, by Equation (14),

$$\Pr[E_{i_1,\ldots,i_j} \mid E_{i_1,\ldots,i_{j-1}}] \leq 2^{2k-n}.$$

Hence,

$$\Pr[E_{i_1,\ldots,i_k}] \leq \prod_{j\in[k]} 2^{2k-n} = 2^{-k(n-2k)}.$$

By the union bound, the probability that the computation-path of P reaches v is at most

$$m^k \cdot 2^{-k(n-2k)}.$$

□

Lemma 8.2. *Let P be a length m affine branching program for parity learning, such that, for every vertex v of P, $\operatorname{codim}(w(v)) \leq k$. Then, the probability that the computation-path of P passes through at least one vertex labeled by a subspace in $\mathcal{A}(n) \setminus \mathcal{G}$ is at most*

$$\frac{m \cdot 2^{3k+2}}{|T|}.$$

Proof. Let $\gamma = 2^{-(k+1)}$. Let $V_0, \ldots, V_m$ be the vertices on the computation-path of P. Note that $V_0, \ldots, V_m$ are random variables that depend on $x, a_1, \ldots, a_m$. Let $S_i = (w(V_i)^{\perp})_1$. By the soundness property in Definition 6.2, for every $1 \leq i \leq m$, we have $S_i \subseteq \operatorname{span}(S_{i-1} \cup \{a_i\})$. For $0 \leq i \leq m$, let E_i be the event that the vertex V_i is labeled by a subspace in $\mathcal{A}(n) \setminus \mathcal{G}$. Recall that

$$\mathcal{G} = \left\{ w \in \mathcal{A}(n) : \ \operatorname{codim}(w) \leq k \ \text{ and } \ w \cap T \neq \emptyset \ \text{ and } \ (w^{\perp})_1 \cap \mathcal{B}(\gamma) = \{\vec{0}\} \right\}.$$

Recall that by Definition 6.2, for every vertex v in the program, if the computation-path reaches v then $x \in w(v)$ and in particular $w(v) \cap T \neq \emptyset$. Thus, $w(V_i)$ surely satisfies the second condition in the definition of $\mathcal{G}$. The first condition in the definition of $\mathcal{G}$ is also surely satisfied by $w(V_i)$, as we assumed that for every vertex v of P, $\operatorname{codim}(w(v)) \leq k$. It follows that E_i is equivalent to $(w(V_i)^{\perp})_1 \cap \mathcal{B}(\gamma) \neq \{\vec{0}\}$. That is, E_i is the event that S_i contains a non-zero vector from $\mathcal{B}(\gamma)$. For $1 \leq i \leq m$, we bound $\Pr[E_i | \neg E_0, \ldots, \neg E_{i-1}]$. It is enough to bound the probability of E_i, for any fixed choice of $x, a_1, \ldots, a_{i-1}$ under which E_{i-1} does not occur (as $E_0, \ldots, E_{i-1}$ only depend on $x, a_1, \ldots, a_{i-1}$). Since we assume that E_{i-1} does not occur, S_{i-1} does not contain any non-zero vector from $\mathcal{B}(\gamma)$. Thus, E_i occurs only if $a_i \oplus a \in \mathcal{B}(\gamma)$ for some vector $a \in S_{i-1}$. As there are at most 2^k vectors in S_{i-1} and as the probability that $a_i \oplus a \in \mathcal{B}(\gamma)$ for a fixed vector, a, is $|\mathcal{B}(\gamma)|/2^n$ we get

$$\Pr[E_i | \neg E_0, \ldots, \neg E_{i-1}] \leq 2^{k-n} \cdot |\mathcal{B}(\gamma)|.$$

Note that $\Pr[E_0] = 0$, hence

$$\begin{aligned} \Pr[E_0 \vee \ldots \vee E_m] &\leq \Pr[E_0] + \Pr[E_1 | \neg E_0] + \ldots + \Pr[E_m | \neg E_0, \ldots, \neg E_{m-1}] \\ &\leq m \cdot 2^{k-n} \cdot |\mathcal{B}(\gamma)|. \end{aligned}$$

By Lemma 4.1, we have $|\mathcal{B}(\gamma)| \leq \frac{2^n}{|T| \cdot \gamma^2} = \frac{2^{n+2k+2}}{|T|}$. Thus,

$$\Pr[E_0 \vee \ldots \vee E_m] \leq m \cdot 2^{k-n} \cdot |\mathcal{B}(\gamma)| \leq \frac{m \cdot 2^{3k+2}}{|T|} .$$

□

9 Time-Space Lower Bounds for Parity Learning

Recall that throughout this section, we think of the set $T \subseteq \{0,1\}^n$ as fixed (see Section 4.4.1). The maximal co-dimension $k \in \mathbb{N}$ will be set in the proof of Theorem 2.

Theorem 1. *Let B be a branching program of length at most m and width at most d for parity learning. Assume for simplicity and without loss of generality that all leaves of B are in the last layer. Let $\epsilon \in (2^{-4n}, 1)$ and $r = \log(|\mathcal{B}(\epsilon)|/\epsilon)$. Then, the success probability of B is at most*

$$2^{k+6} m\sqrt{\epsilon} + \frac{m \cdot 2^{3k+3}}{|T|} + m^{k+1} \cdot 2^{-k(n-r-2k)} \cdot 2nd.$$

Proof. Let B be a branching program of length m and width d for parity learning. Assume for simplicity and without loss of generality that all leaves of B are in the last layer. Denote by β the success probability of B.

By Lemma 7.1, there exists a length m affine branching program P for parity learning, such that:

1. For every $k' \leq k$, the number of vertices in P, that are labeled with an affine subspace in $\mathcal{G}$ of co-dimension k', is at most
$$n \cdot 2^{rk'+1} \cdot dm. \tag{15}$$

2. The probability that the leaf reached by the computation-path of P is labeled by a subspace in $\mathcal{A}(n) \setminus \mathcal{G}$ is at least
$$\beta - 2^{k+6} m\sqrt{\epsilon} - \frac{2^{k+1}}{|T|}. \tag{16}$$

Let E be the event that the computation-path of P passes through at least one vertex labeled by a subspace in $\mathcal{A}(n) \setminus \mathcal{G}$. We next upper bound $\Pr[E]$. We partition the event E into two sub-events: Assume that E occurs. Let V be the first vertex along the computation-path labeled by a subspace in $\mathcal{A}(n) \setminus \mathcal{G}$.

- Let E_1 be the event that E occurs and $\operatorname{codim}(w(V)) \leq k$.
- Let E_2 be the event that E occurs and $\operatorname{codim}(w(V)) > k$.

We note that whenever E occurs, all the vertices on the computation-path reached prior to V are labeled with subspaces in $\mathcal{G}$, and, in particular, with subspaces of co-dimension at most k.

We first upper bound $\Pr[E_1]$. Assume without loss of generality that every vertex u of P has $\operatorname{codim}(w(u)) \le k$. Otherwise, we can remove all vertices with co-dimension greater than k, without changing $\Pr[E_1]$. By Lemma 8.2,

$$\Pr[E_1] \le \frac{m \cdot 2^{3k+2}}{|T|}.$$

We next upper bound $\Pr[E_2]$. Let V' be the predecessor of V on the computation-path. As mentioned above, $w(V') \in \mathcal{G}$ and $\operatorname{codim}(w(V')) \le k$. In addition, $\operatorname{codim}(w(V')) \ge k$, by the soundness property of Definition 6.2. We get that $w(V') \in \mathcal{G}$ and $\operatorname{codim}(w(V')) = k$. Let $\mathcal{V}$ be the set of vertices v such that $w(v) \in \mathcal{G}$ and $\operatorname{codim}(w(v)) = k$. Thus, if E_2 occurs, then $V' \in \mathcal{V}$. For every vertex $v \in \mathcal{V}$, let E_v be the event that v is a vertex on the computation-path, and all vertices on the computation-path prior to v are of co-dimension at most k. We have that

$$\Pr[E_2] = \Pr[E_2 \wedge (V' \in \mathcal{V})] = \sum_{v\in\mathcal{V}} \Pr[E_2 \wedge (V' = v)] \le \sum_{v\in\mathcal{V}} \Pr[E_v].$$

We now upper bound $\Pr[E_v]$ for every $v \in \mathcal{V}$. Assume without loss of generality that every vertex u of P has $\operatorname{codim}(w(u)) \le k$. Otherwise, we can remove all vertices with co-dimension greater than k, without changing $\Pr[E_v]$ for any $v \in \mathcal{V}$. By Lemma 8.1, $\Pr[E_v] \le m^k \cdot 2^{-k(n-2k)}$ for every $v \in \mathcal{V}$. By Equation (15), $|\mathcal{V}| \le n \cdot 2^{rk+1} \cdot dm$. Thus,

$$\Pr[E_2] \le m^{k+1} \cdot 2^{-k(n-r-2k)} \cdot 2nd.$$

By Equation (16) we have that

$$\beta - 2^{k+6} m\sqrt{\epsilon} - \frac{2^{k+1}}{|T|} \le \Pr[E] \le \Pr[E_1] + \Pr[E_2].$$

That is,

$$\begin{aligned}\beta &\le 2^{k+6} m\sqrt{\epsilon} + \frac{2^{k+1}}{|T|} + \frac{m \cdot 2^{3k+2}}{|T|} + m^{k+1} \cdot 2^{-k(n-r-2k)} \cdot 2nd \\ &\le 2^{k+6} m\sqrt{\epsilon} + \frac{m \cdot 2^{3k+3}}{|T|} + m^{k+1} \cdot 2^{-k(n-r-2k)} \cdot 2nd.\end{aligned}$$

□

Theorem 2. *Let B be a branching program of length at most m and width at most d for parity learning, where $8m \le |T|^{1/30}$. Assume for simplicity and without loss of generality that all leaves of B are in the last layer. Assume that the success probability of B is at least $1/m$. Then,*

$$\log(d) \ge \tfrac{1}{2} \cdot \log\left(\tfrac{2^n}{|\mathcal{B}(1/(8m)^6)|}\right) \cdot \log(m) - \log(4n).$$

Proof. Let $k = \log(m)$, $\epsilon = 1/(8m)^6$. By Theorem 1, the success probability of B is at most

$$\begin{aligned}&2^{k+6} m\sqrt{\epsilon} + \frac{m \cdot 2^{3k+3}}{|T|} + m^k \cdot 2^{-k(n-\log(|\mathcal{B}(\epsilon)|/\epsilon)-2k)} \cdot 2nmd \\ &\le 1/(4m) + 1/(4m) + 2^{-\log(m)\cdot(n-\log|\mathcal{B}(\epsilon)|-\log(1/\epsilon)-3\log(m))} \cdot 2nmd \\ &= 1/(2m) + 2^{-\log(m)\cdot(n-\log|\mathcal{B}(\epsilon)|-9\log(m)-18)} \cdot 2nmd.\end{aligned}$$

Since we assumed that the success probability is at least $1/m$, we get

$$1/(2m) \leq 2^{-\log(m)\cdot(n-\log|\mathcal{B}(\epsilon)|-9\log(m)-18)} \cdot 2nmd.$$

Equivalently,

$$\begin{aligned}\log(d) &\geq -\log(4nm^2) + \log(m)\cdot(n - \log|\mathcal{B}(\epsilon)| - 9\log(m) - 18)\\ &= \log(m)\cdot\left(\log\left(\tfrac{2^n}{|\mathcal{B}(1/(8m)^6)|}\right) - 9\cdot\log(m) - 20\right) - \log(4n)\\ &\geq \log(m)\cdot\left(\log\left(\tfrac{2^n}{|\mathcal{B}(1/(8m)^6)|}\right) - 9\cdot\log(8m)\right) - \log(4n)\end{aligned}$$

Next we show that $9\log(8m) \leq \frac{1}{2}\cdot\log\left(\frac{2^n}{|\mathcal{B}(1/(8m)^6)|}\right)$. Using Lemma 4.1, $|\mathcal{B}(\epsilon)| \leq \frac{2^n}{|T|\cdot\epsilon^2}$. Thus,

$$|\mathcal{B}(1/(8m)^6)| \leq \frac{2^n\cdot(8m)^{12}}{|T|}$$

As we assumed $(8m)^{30} \leq |T|$, we get

$$\log\left(\tfrac{2^n}{|\mathcal{B}(1/(8m)^6)|}\right) \geq \log\left(\tfrac{|T|}{(8m)^{12}}\right) \geq \log((8m)^{18}) = 18\log(8m)$$

as promised. Overall, we get

$$\begin{aligned}\log(d) &\geq \log(m)\cdot\left(\log\left(\tfrac{2^n}{|\mathcal{B}(1/(8m)^6)|}\right) - 9\cdot\log(8m)\right) - \log(4n)\\ &\geq \frac{1}{2}\cdot\log(m)\cdot\log\left(\tfrac{2^n}{|\mathcal{B}(1/(8m)^6)|}\right) - \log(4n).\end{aligned}$$

□

10 Applications

10.1 Sparse Parities

In this section, we give a time-space tradeoff for parity learning over the set T_ℓ containing all vectors of Hamming weight exactly ℓ. Formally, for $x \in \{0,1\}^n$, we denote $\mathrm{wt}(x) = \sum_{i=1}^n x_i$. Let $\ell \in [n]$. We define the set T_ℓ by

$$T_\ell = \{x \in \{0,1\}^n \;:\; \mathrm{wt}(x) = \ell\}\;.$$

We will use the following lemma that is proved in Appendix A.

Lemma 10.1. *Let $\epsilon \in (0,1]$ and $\ell \in \mathbb{N}$. Assume that $\epsilon \geq \left(\frac{8\ell}{n}\right)^{\ell/2}$. Then,*

$$|\mathcal{B}_{T_\ell}(\epsilon)| \leq 2e^{-\epsilon^{2/\ell}\cdot n/8}\cdot 2^n.$$

The following theorem is a corollary of Theorem 2, and the main result of this section.

Theorem 3. *Let $\ell \in [n]$. Let B be a branching program of length at most m and width at most d for parity learning over T_ℓ, where $8m \leq \left(\frac{n}{8\ell}\right)^{\ell/30}$. Assume for simplicity and without loss of generality that all leaves of B are in the last layer. Assume that the success probability of B is at least $1/m$. Then,*

$$\log(d) \geq n\cdot\frac{\log(m)}{16\cdot(8m)^{12/\ell}} - \log(4mn).$$

In particular,

1. *There exists a sufficiently small constant $c \in (0,1)$ such that for $\ell \le cn$ and $m = 2^\ell$, we have*
$$\log(d) \ge \Omega(n\ell).$$

2. *For $\ell \le n^{0.9}$ and $m = \ell^{0.0001\cdot\ell}$, we have*
$$\log(d) \ge \Omega(n \cdot \ell^{0.99}).$$

Proof. Let $\epsilon = 1/(8m)^6$. Since $8m \le \left(\frac{n}{8\ell}\right)^{\ell/30} \le \left(\frac{n}{8\ell}\right)^{\ell/12}$, we have that $\epsilon \ge \left(\frac{8\ell}{n}\right)^{\ell/2}$, thus by Lemma 10.1
$$|\mathcal{B}_{T_\ell}(\epsilon)| \le 2e^{-\epsilon^{2/\ell}\cdot n/8} \cdot 2^n.$$

By Theorem 2, since $8m \le \left(\frac{n}{8\ell}\right)^{\ell/30} \le \binom{n}{\ell}^{1/30} = |T_\ell|^{1/30}$,
$$\begin{aligned}\log(d) &\ge \frac{1}{2}\cdot\log\left(\tfrac{2^n}{|\mathcal{B}_{T_\ell}(\epsilon)|}\right)\cdot\log(m) - \log(4n)\\ &\ge \frac{1}{2}\cdot\log\left(e^{\epsilon^{2/\ell}\cdot n/8}/2\right)\cdot\log(m) - \log(4n)\\ &\ge \frac{1}{2}\cdot\epsilon^{2/\ell}\cdot(n/8)\cdot\log(m) - \log(4mn)\\ &= n\cdot\frac{\log(m)}{16\cdot(8m)^{12/\ell}} - \log(4mn).\end{aligned}$$

□

10.2 Small Biased Sets

Theorem 4. *Let $T \subseteq \{0,1\}^n$ be an ϵ-biased set, for $\epsilon \in (2^{-n/2}, 1/2)$. Let B be a branching program of length at most m and width at most d for parity learning over T, where $8m = (1/2\epsilon)^{1/15}$. Assume for simplicity and without loss of generality that all leaves of B are in the last layer. Assume that the success probability of B is at least $1/m$. Then,*
$$\log(d) \ge \Omega(n\cdot\log(1/\epsilon)).$$

Proof. Since T is an ϵ-biased set, by Parseval's identity and as $\epsilon \in (2^{-n/2}, 1)$,
$$\begin{aligned}\frac{1}{|T|\cdot 2^n} &= \mathop{\mathbf{E}}_{x\in_R\{0,1\}^n}\left[(\mathcal{U}_T(x))^2\right] = \sum_{\alpha\in\{0,1\}^n}\left(\widehat{\mathcal{U}}_T(\alpha)\right)^2\\ &\le (2^n-1)\cdot(2^{-n}\epsilon)^2 + (2^{-n})^2 \le 2^{-n}\epsilon^2 + 2^{-2n} \le 2\cdot 2^{-n}\epsilon^2.\end{aligned}$$

This implies, $|T| \ge 1/(2\epsilon^2)$, and hence, $8m \le |T|^{1/30}$. Since T is an ϵ-biased set, and $\epsilon \le 1/(8m)^6$, it holds that $|\mathcal{B}_T(1/(8m)^6)| = 1$. Using Theorem 2, as $8m = (1/2\epsilon)^{1/15}$,
$$\log(d) \ge \frac{1}{2}\cdot n\cdot\log(m) - \log(4n) \ge \Omega(n\cdot\log(1/\epsilon)).$$

□

References

[ADR02] Yonatan Aumann, Yan Zong Ding, Michael O. Rabin: Everlasting security in the bounded storage model. IEEE Transactions on Information Theory 48(6): 1668-1680 (2002) 4

[AGHP92] Noga Alon, Oded Goldreich, Johan Håstad, René Peralta: Simple Construction of Almost k-wise Independent Random Variables. Random Struct. Algorithms 3(3): 289-304 (1992) 4

[AR99] Yonatan Aumann, Michael O. Rabin: Information Theoretically Secure Communication in the Limited Storage Space Model. CRYPTO 1999: 65-79 4

[CM97] Christian Cachin, Ueli M. Maurer: Unconditional Security Against Memory-Bounded Adversaries. CRYPTO 1997: 292-306 4

[DM04] Stefan Dziembowski, Ueli M. Maurer: On Generating the Initial Key in the Bounded-Storage Model. EUROCRYPT 2004: 126-137 4

[M92] Ueli M. Maurer: Conditionally-Perfect Secrecy and a Provably-Secure Randomized Cipher. J. Cryptology 5(1): 53-66 (1992) 4

[NN93] Joseph Naor, Moni Naor: Small-Bias Probability Spaces: Efficient Constructions and Applications. SIAM J. Comput. 22(4): 838-856 (1993) 4

[R16] Ran Raz: Fast Learning Requires Good Memory: A Time-Space Lower Bound for Parity Learning. Electronic Colloquium on Computational Complexity (ECCC) 23: 19 (2016) 1, 2, 3, 4, 5, 6, 8

[S14] Ohad Shamir: Fundamental Limits of Online and Distributed Algorithms for Statistical Learning and Estimation. NIPS 2014: 163-171 3

[SVW15] Jacob Steinhardt, Gregory Valiant, Stefan Wager: Memory, Communication, and Statistical Queries. Electronic Colloquium on Computational Complexity (ECCC) 22: 126 (2015) 2, 3

[V03] Salil P. Vadhan: Constructing Locally Computable Extractors and Cryptosystems in the Bounded-Storage Model. J. Cryptology 17(1): 43-77 (2004) (also in Crypto 2003) 4

[VV16] Gregory Valiant, Paul Valiant: Information Theoretically Secure Databases. Electronic Colloquium on Computational Complexity (ECCC) 23: 78 (2016) 3

A The Fourier Spectrum of Slices of The Hypercube

For $x \in \{0,1\}^n$, we denote $\mathrm{wt}(x) = \sum_{i=1}^{n} x_i$. Let $\ell \in [n]$. We next analyze the Fourier spectrum of the indicator function of the set

$$T_\ell = \{x \in \{0,1\}^n \ : \ \mathrm{wt}(x) = \ell\} \ .$$

It suffices to analyze the Fourier coefficients of weight at most $n/2$, due to the following simple identity:

Fact A.1. *For $\alpha \in \{0,1\}^n$,*

$$\widehat{\mathcal{U}_{T_\ell}}(\alpha) = (-1)^\ell \cdot \widehat{\mathcal{U}_{T_\ell}}(\alpha \oplus \vec{1}).$$

The fact holds as

$$\mathop{\mathbf{E}}_{x \in_R T_\ell}[\chi_\alpha(x)] = \mathop{\mathbf{E}}_{x \in_R T_\ell}[\chi_{\alpha\oplus\vec{1}}(x) \cdot \chi_{\vec{1}}(x)] = (-1)^\ell \cdot \mathop{\mathbf{E}}_{x \in_R T_\ell}[\chi_{\alpha\oplus\vec{1}}(x)].$$

Lemma A.2. *Let $\alpha \in \{0,1\}^n$ with $\mathrm{wt}(\alpha) = \frac{n(1-\delta)}{2} \leq \frac{n}{2}$, for some $\delta \in [0,1]$. Let $j_{\min} = \max\{0, \lceil(\ell - \delta n)/2\rceil\}$ and $j_{\max} = \min\{\lfloor \ell/2 \rfloor, \mathrm{wt}(\alpha)\}$. For $j \in \{j_{\min}, \ldots, j_{\max}\}$, define*

$$a_j = \frac{\binom{\delta n}{\ell - 2j} \cdot \binom{(n-\delta n)/2}{j}}{\binom{n}{\ell}}.$$

Then,

$$\widehat{\mathcal{U}_{T_\ell}}(\alpha) = 2^{-n} \cdot \sum_{j=j_{\min}}^{j_{\max}} (-1)^j \cdot a_j.$$

In addition,

$$\left|\widehat{\mathcal{U}_{T_\ell}}(\alpha)\right| \leq 2^{-n} \cdot \max_{j \in \{j_{\min},\ldots,j_{\max}\}} \{a_j\}.$$

Proof. We note that $\widehat{\mathcal{U}_{T_\ell}}(\alpha) = 2^{-n} \cdot \mathbf{E}_{x \in_R T_\ell}[\chi_\alpha(x)]$ and compute $\mathbf{E}_{x \in_R T_\ell}[\chi_\alpha(x)]$. We assume that α is of the form $\alpha = (01)^{(n-\delta n)/2} \circ 0^{\delta n}$ (that is, α starts with $(n - \delta n)/2$ pairs of 01 followed by δn 0's). This assumption is without loss of generality as $\mathbf{E}_{x \in_R T_\ell}[\chi_\alpha(x)]$ only depends on the weight of α.

Let X be a random variable uniformly distributed over T_ℓ. Let E be the event that for every $i \in \{1, \cdots, (n - \delta n)/2\}$ it holds that $X_{2i-1} = X_{2i}$. Observe that by a symmetry argument,

$$\mathop{\mathbf{E}}_X[\chi_\alpha(X) \mid \neg E] = 0.$$

The reason is that whenever E does not hold, there exists $i \in \{1, \cdots, (n - \delta n)/2\}$ such that $X_{2i-1} \neq X_{2i}$, and the case where $(X_{2i-1}, X_{2i}) = (0,1)$ cancels the case where $(X_{2i-1}, X_{2i}) = (1,0)$ in the expectation.

For $j \in \{j_{\min}, \ldots, j_{\max}\}$, let E_j be the event that E occurs and $\sum_{i=1}^{(n-\delta n)} X_i = 2j$. If E_j occurs, then

$$\chi_\alpha(X) = (-1)^j.$$

The first assertion follows as

$$\mathop{\mathbf{E}}_{x \in_R T_\ell}[\chi_\alpha(x)] = \sum_{j=j_{\min}}^{j_{\max}} (-1)^j \cdot \Pr[E_j] = \sum_{j=j_{\min}}^{j_{\max}} (-1)^j \cdot \frac{|\{x : x \in E_j\}|}{|T_\ell|} = \sum_{j=j_{\min}}^{j_{\max}} (-1)^j \cdot \frac{\binom{(n-\delta n)/2}{j} \cdot \binom{\delta n}{\ell - 2j}}{\binom{n}{\ell}}.$$

To prove the second assertion, consider the sequence $(a_{j_{\min}}, \ldots, a_{j_{\max}})$. The ratio between two consecutive elements in the sequence is given by

$$\frac{a_j}{a_{j-1}} = \frac{\binom{\delta n}{\ell-2j} \cdot \binom{(n-\delta n)/2}{j}}{\binom{\delta n}{\ell - 2(j-1)} \cdot \binom{(n-\delta n)/2}{j-1}} = \frac{(\ell - 2j + 2) \cdot (\ell - 2j + 1) \cdot ((n - \delta n)/2 - j + 1)}{(\delta n - \ell + 2j) \cdot (\delta n - \ell + 2j - 1) \cdot j}.$$

Observe that the right hand side is a decreasing function of j. Thus, the sequence $(a_{j_{\min}}, \dots, a_{j_{\max}})$ is unimodal, i.e., there exists i such that

$$a_{j_{\min}} \le a_{j_{\min}+1} \le \dots \le a_{i-1} \le a_i \ge a_{i+1} \ge \dots \ge a_{j_{\max}}.$$

The second assertion follows by the following claim:

Claim A.3. *Let $(a_0, \dots, a_{j'})$ be a non-negative unimodal sequence. Then,*

$$\left|\sum_{j=0}^{j'} (-1)^j \cdot a_j\right| \le \max_{j \in \{0,\dots,j'\}} \{a_j\}.$$

Proof. Since $(a_0, \dots, a_{j'})$ is a unimodal sequence, there exists i such that

$$a_0 \le a_1 \le \dots \le a_{i-1} \le a_i \ge a_{i+1} \ge \dots \ge a_{j'}.$$

Assume that $(-1)^i = 1$. The other case is similar. Assume that j' is even, otherwise add a 0-element at the end of the sequence that will not change the sum and the fact that the sequence is unimodal.

It holds that

$$\sum_{j=0}^{j'} (-1)^j \cdot a_j = (a_0 - a_1) + (a_2 - a_3) + \dots + (a_{i-2} - a_{i-1}) \quad + \quad a_i$$
$$+ (-a_{i+1} + a_{i+2}) + (-a_{i+3} + a_{i+4}) + \dots + (-a_{j'-1} + a_{j'}) \le a_i,$$

where the last inequality holds as each bracketed pair is non-positive.

It also holds that

$$\sum_{j=0}^{j'} (-1)^j \cdot a_j = a_0 + (-a_1 + a_2) + (-a_3 + a_4) + \dots + (-a_{i-1} + a_i) \quad - \quad a_i$$
$$+ (a_i - a_{i+1}) + (a_{i+2} - a_{i+3}) + \dots + (a_{j'-2} - a_{j'-1}) + a_{j'} \ge -a_i,$$

where the last inequality holds as a_0, $a_{j'}$, and each bracketed pair are non-negative. □

□

Lemma A.4. *Let $\alpha \in \{0,1\}^n$ with $\mathrm{wt}(\alpha) = \frac{n(1-\delta)}{2}$ for some $\delta \in (-1,1)$. Then,*

$$\left|\widehat{\mathcal{U}_{T_\ell}}(\alpha)\right| \le 2^{-n} \cdot \left(|\delta| + \sqrt{\tfrac{\ell}{n - |\delta| \cdot n}}\right)^{\ell}.$$

Proof. Assume that $\delta \in [0,1)$. The case where $\delta \in (-1,0]$ follows from the case $\delta \in [0,1)$ using Fact A.1. As in Lemma A.2, let $j_{\min} = \max\{0, \lceil(\ell - \delta n)/2\rceil\}$ and $j_{\max} = \min\{\lfloor \ell/2 \rfloor, \mathrm{wt}(\alpha)\}$. For $j \in \{j_{\min}, \dots, j_{\max}\}$, let

$$a_j = \frac{\binom{\delta n}{\ell - 2j} \cdot \binom{(n-\delta n)/2}{j}}{\binom{n}{\ell}} = \frac{\binom{\delta n}{\ell - 2j} \cdot \binom{n - \delta n}{2j}}{\binom{n}{\ell}} \cdot \frac{\binom{(n-\delta n)/2}{j}}{\binom{n-\delta n}{2j}}.$$

We bound each of the two terms in the right hand side of the above equation separately.

$$\begin{aligned}\frac{\binom{\delta n}{\ell-2j}\cdot\binom{n-\delta n}{2j}}{\binom{n}{\ell}} &= \frac{(\delta n)\cdots(\delta n-\ell+2j+1)}{n\cdots(n-\ell+2j+1)}\cdot\frac{(n-\delta n)\cdots(n-\delta n-2j+1)}{(n-\ell+2j)\cdots(n-\ell+1)}\cdot\binom{\ell}{2j}\\ &\le \delta^{\ell-2j}\cdot\frac{(n-\delta n)\cdots(n-\delta n-2j+1)}{(n-\ell+2j)\cdots(n-\ell+1)}\cdot\binom{\ell}{2j}\\ &\le \delta^{\ell-2j}\cdot\binom{\ell}{2j}, \qquad \text{(since } 2j\ge 2j_{\min}\ge \ell-\delta n\text{)}\end{aligned}$$

and

$$\frac{\binom{(n-\delta n)/2}{j}}{\binom{n-\delta n}{2j}} = \frac{\binom{(n-\delta n)/2}{j}\cdot\binom{(n-\delta n)/2}{j}}{\binom{n-\delta n}{2j}}\cdot\frac{1}{\binom{(n-\delta n)/2}{j}} \le \frac{1}{\binom{(n-\delta n)/2}{j}} \le \left(\frac{2j}{n-\delta n}\right)^j.$$

Overall we get,

$$\begin{aligned} a_j &\le \delta^{\ell-2j}\cdot\binom{\ell}{2j}\cdot\left(\frac{2j}{n-\delta n}\right)^j \le \delta^{\ell-2j}\cdot\binom{\ell}{2j}\cdot\left(\frac{\ell}{n-\delta n}\right)^j \qquad \text{(as } j\le j_{\max}\le\lfloor \ell/2\rfloor\text{)}\\ &= \delta^{\ell-2j}\cdot\binom{\ell}{2j}\cdot\left(\sqrt{\tfrac{\ell}{n-\delta n}}\right)^{2j} \le \left(\delta+\sqrt{\tfrac{\ell}{n-\delta n}}\right)^{\ell}.\end{aligned}$$

The assertion follows from Lemma A.2. □

Recall that

$$\mathcal{B}_{T_\ell}(\epsilon) = \left\{\alpha\in\{0,1\}^n : \left|\widehat{\mathcal{U}}_{T_\ell}(\alpha)\right| > 2^{-n}\epsilon\right\}.$$

Next, we prove Lemma 10.1.

Lemma (Lemma 10.1 Restated). *Let $\epsilon\in(0,1]$ and $\ell\in\mathbb{N}$. Assume that $\epsilon\ge\left(\frac{8\ell}{n}\right)^{\ell/2}$. Then,*

$$|\mathcal{B}_{T_\ell}(\epsilon)| \le 2e^{-\epsilon^{2/\ell}\cdot n/8}\cdot 2^n.$$

Proof. Let $\delta_0=\epsilon^{1/\ell}/2\le 1/2$. For $|\delta|\le\delta_0$ and $\alpha\in\{0,1\}^n$ with $\mathrm{wt}(\alpha)=\frac{n(1-\delta)}{2}$, it holds that

$$\left|\widehat{\mathcal{U}}_{T_\ell}(\alpha)\right| \le 2^{-n}\cdot\left(|\delta|+\sqrt{\tfrac{\ell}{n-|\delta|\cdot n}}\right)^{\ell} \le 2^{-n}\cdot\left(\delta_0+\sqrt{\tfrac{2\ell}{n}}\right)^{\ell} \le 2^{-n}\cdot\left(\epsilon^{1/\ell}/2+\epsilon^{1/\ell}/2\right)^{\ell} = 2^{-n}\epsilon.$$

Therefore, $\mathcal{B}_{T_\ell}(\epsilon)\subseteq\left\{\alpha:\ \mathrm{wt}(\alpha)=\frac{n(1-\delta)}{2},\ |\delta|>\delta_0\right\}$. Using Chernoff,

$$|\mathcal{B}_{T_\ell}(\epsilon)| \le 2e^{-\delta_0^2\cdot n/2}\cdot 2^n = 2e^{-\epsilon^{2/\ell}\cdot n/8}\cdot 2^n.$$

□