Electronic Colloquium on Computational Complexity, Report No. 44 (2000)

On the Distribution of the Number of Roots of Polynomials and Explicit Logspace Extractors

Tzvika Hartman[†] Ran Raz [‡]

Abstract

Weak designs were defined in [RRV99a] and are used in constructions of extractors. Roughly speaking, a weak design is a collection of subsets satisfying some near-disjointness properties. Constructions of weak designs with certain parameters are given in [RRV99a]. These constructions are explicit in the sense that they require time and space polynomial in the number of subsets. However, the constrcuctions require time and space polynomial in the number of subsets even when needed to output only one specific subset out of the collection. Hence, the constructions are not explicit in a stronger sense.

In this work we provide constructions of weak designs (with parameters similar to the ones of [RRV99a]) that can be carried out in space logarithmic in the number of subsets. Moreover, our constructions are explicit even in a stronger sense; given an index to a subset, we output the specified subset in time and space polynomial in the size of the index. Using our constructions, we obtain extractors similar to the RRV extractors in terms of parameters and that can be evaluated in logarithmic space.

Our main construction is algebraic. In order to prove the properties of weak designs we prove some combinatorial-algebraic lemmas that may be interesting in their own right. These lemmas regard the number of roots of polynomials over finite fields. In particular, we prove that the number of polynomials (over any finite field) with k roots, vanishes exponentially in k. In other words, we prove that the number of roots of a random polynomial is not only bounded by its degree (a well known fact), but furthermore, it is concentrated exponentially around its expectation (which is 1). Our lemmas are proved by combinatorial-algebraic arguments. The main lemma is also proved by a probabilistic argument.

1 Introduction

1.1 Background

Weak designs were defined in [RRV99a]. Roughly speaking, a weak design is a collection of subsets, satisfying some near-disjointness properties. In particular, we take the exponentiations of the size of the intersections between every pair of subsets and bound its average (see definition 5).

[RRV99a] provide an explicit construction of weak designs which outputs all the subsets in time and space polynomial in the number of subsets. However, they require time and space time and space polynomial in the number of subsets even when needed to output only one subset. In other

[†]Dept. of Applied Math and Computer Science, Weizmann Institute, Rehovot 76100, Israel. E-mail: ranraz@wisdom.weizmann.ac.il. Phone: +972-8-934-3544

[‡]Dept. of Applied Math and Computer Science, Weizmann Institute, Rehovot 76100, Israel. E-mail: tzvi@wisdom.weizmann.ac.il.

ISSN 1433-8092

words, if we consider a stronger notion of explicitness (as in definition 6), where the algorithm gets as an input an index of a subset and is required to output only the specified subset, then the constructions are not explicit. This is since the time and space required are exponential in the size of the index (which is logarithmic in the number of subsets).

In this work we provide constructions of weak designs, which are explicit even in a stronger sense: The algorithm outputs a specific subset in time and space polynomial in the size of the index of the subset (the time is actually polynomial also in the size of the subsets). It hence follows that our algorithm can output all subsets in space logarithmic in the number of subsets.

The main application of weak designs in [RRV99a] is in constructing *extractors*. Extractors, defined by Nisan and Zuckerman [NZ96], are functions (which can be viewed as graphs as well) that have very strong randomness properties. Roughly speaking, these functions convert a "somewhat random" distribution (which will be defined later) into an almost random one. This is done using a small number of truly random bits as a catalyst. A large body of work has been done in giving explicit constructions of extractors, with a steady improvement in its parameters. Extractors have a wide variety of applications (see section 2.1 and also [NT98]).

A recent breakthrough was made by Trevisan [Tre99], who uses the Nisan-Wigderson pseudo-random generator [NW94], previously used only in the computational setting, in order to construct extractors (which are information-theoretic objects). For some setting of parameters, Trevisan's extractors are optimal and improve upon previous constructions. The Nisan-Wigderson generator and hence the Trevisan Extractor are based on a "combinatorial design", which is a collection of subsets of a given universe, in which the intersection between every two subsets is bounded.

The main observation in [RRV99a] is that Trevisan's construction yields an extractor even if we replace these combinatorial designs by a weaker notion of combinatorial designs. They introduced the notion of *weak design* and proved that the same construction, based on weak designs, induces an extractor as well. Since the requirements from the weak designs are weaker, they can be constructed with better parameters. Hence, the parameters of the implied extractors are in many interesting cases better than the ones of Trevisan's extractors (specifically, the number of truly random bits used by the extractor is much smaller).

A construction of weak designs is given by [RRV99a]. The properties of the design are proved by an application of the Probabilistic Method. The proof is derandomized by the Method of Conditional Probabilities. This construction, as discussed above, can be done in time polynomial in the number of subsets. However, this construction requires also polynomial space. This implies that the evaluation of the RRV extractor cannot be done in small space.

1.2 Overview of Our Results

In this work we use a different approach, an algebraic one, for constructing weak designs. This approach was introduced by Nisan and Wigderson [NW94] in order to construct standard designs (see definition 4 and lemma 2.2). By using this approach we obtain constructions of weak designs that have similar parameters to the ones constructed in [RRV99a], with the advantage that they are explicit in a stronger sense: We provide an algorithm that gets as an input an index to a subset and outputs the specified subset in time and space polynomial in the size of the index of the subset (the time is actually polynomial also in the size of the subsets). This improves upon the constructions of [RRV99a], in which the time and space required is exponential in the size of the input. Moreover, our algorithm can output *all* the subsets in space which is logarithmic in the number of subsets, whereas the RRV constructions require polynomial space. By plugging these designs into the RRV extractor we obtain an extractor which can be evaluated in space logarithmic in its input.

In order to obtain our main result, we prove some combinatorial-algebraic lemmas that may be interesting in their own right. The lemmas regard the number of roots and the number of linear factors of polynomials over finite fields. Roughly speaking, we prove that the number of polynomials (over any finite field) with k roots (or k linear factors) vanishes exponentially in k. In other words, we prove that the number of roots of a random polynomial is not only bounded by its degree (a well known and useful fact), but furthermore, is typically much smaller. In particular, it is concentrated exponentially around its average, which is 1.

The lemmas are proved by combinatorial-algebraic techniques. A slightly different version of our main lemma (which is used for our main result) is also proved by a probabilistic argument. This is done using the fact that there is a limited independence between samples generated by a polynomial. By defining the appropriate random variables, we show that this limited independence suffices to prove our result.

Our algebraic construction yields a weak design where some of the parameters have specific values (these specific values are the most interesting values for constructing extractors). In addition, we provide general tools to take a weak design with specific parameters and construct a new weak design with other parameters (improving some parameters while paying the price in other ones). These tools use techniques such as union of subsets of several designs and direct product of designs. Applying these tools to our main construction yields a more general construction, which matches, up to a constant factor, the constructions of [RRV99a] (having the advantage of being explicit in the strong sense).

1.3 Organization of the Paper

In section 2 we give the formal definitions of extractors and describe briefly previous work on extractors, with emphasis on the Trevisan and RRV extractors. In addition we define combinatorial designs and give some known constructions. We state our results, which will be proved later on and describe an application of the results. Section 3 contains some combinatorial-algebraic lemmas, which will be used in our main construction. In section 4 we prove our main result - an explicit construction of weak-designs for specific parameters. Then, in 5, we generalize this construction to get explicit constructions of weak designs with parameters that match, up to constant factors, the parameters of the probabilistic constructions in [RRV99a]. We conclude with a short discussion in section 6.

2 Preliminaries

2.1 Extractors

2.1.1 Definitions

Before defining extractors, we need to define some related notions.

Definition 1 (Statistical difference) *The* statistical difference *between two distributions X and Y over $\{0,1\}^m$ is defined as*

$$\max_D |Pr[D(X) = 1] - Pr[D(Y) = 1]|$$

where the maximum is taken over all functions (sometimes called "distinguishers") $D : \{0,1\}^m \to \{0,1\}$. We say that two distributions are ϵ − close if their statistical difference is at most ϵ.

Statistical difference is sometimes called *variation difference.*

Definition 2 (Min-entropy) *A distribution X is said to have* min-entropy k *if for all x,*

$$Pr[X = x] \leq 2^{-k}$$

It is useful to think of a distribution of min-entropy k as being uniform over a subset of size 2^k. We will denote the uniform distribution on strings of length j by U_j. Now we can define our main object (we define it as a function, a similar definition can be given by a graph):

Definition 3 (Extractor) *A function $EXT : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is called an (k, ϵ) –* extractor *if for every distribution X on $\{0,1\}^n$ with min-entropy k , $EXT(X, U_d)$ is $\epsilon -$ close to U_m.*

This function actually *extracts* m bits of randomness from the given n bits, as long as these n bits have min-entropy k. In order to do that, it needs additional d truly random bits. In extractors, we would like to minimize d (the number of additional truly random bits), while keeping m (the number of extracted "almost" random bits) as close to k (the amount of randomness in the source) as possible. It can be shown, non-constructively (by a probabilistic argument), that for every $n, k \leq n$ and $\epsilon > 0$ there exists a (k, ϵ)-extractor $EXT : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, with $m = k$ and $d = O(\log(n/\epsilon))$, i.e., it is possible to extract all the randomness of the source using only a logarithmic number of additional truly random bits. This gives an optimal non-explicit construction of extractors. However, we are interested in explicit constructions of extractors.

A family of extractors $\{EXT_i : \{0,1\}^{n_i} \times \{0,1\}^{d_i} \to \{0,1\}^{m_i}\}_{i \in \mathbb{N}}$ is called *explicit* if EXT_i can be evaluated in time poly(n_i). Notice that we did not consider the parameters d_i in the complexity measurement since an extractor is interesting mainly if d_i is smaller than n_i. For the sake of readability, we state our definitions and theorems in terms of individual extractors rather than families of extractors (although it is actually meaningless of an individual extractor being explicit).

The following is a stronger notion of explicitness: An explicit (k, ϵ)-extractor $EXT : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ will be called an explicit *logspace* extractor if EXT is computable in space $s = O(\log n)$.

2.1.2 Previous work and Applications

Extractors were first defined and constructed by Nisan and Zuckerman [NZ96]. This followed a large body of work in the late 1980's, regarding weak notions of randomness [SV86, VV85, CG88, Zuc90]. Improved constructions of extractors appeared in [WZ95, GW97, SZ98, Zuc97, NT98]. A new approach to constructing extractors was recently initiated by Trevisan [Tre99] (and then [RRV99a]), who uses the Nisan-Wigderson pseudo-random generator [NW94] for constructing extractors (these constructions will be discussed in detail in subsubsection 2.1.3).

Besides the natural application of simulating randomized algorithms with weak random sources [Zuc96], explicit constructions of extractors have a wide variety of applications, including constructing oblivious samplers [Zuc97]; constructive leader election [Zuc97]; randomness efficient error-reduction in randomized algorithms and interactive proofs [Zuc97]; explicit constructions of expander graphs, superconcentrators, and sorting networks [WZ95]; hardness of approximation [Zuc96]; pseudorandom generators for space-bounded computation [NZ96, RR99]; derandomizing BPP under circuit-complexity assumptions [STV99] and other problems in complexity theory [GZ97].

For a detailed survey on previous work on extractors and their applications, see [NT98]. The following table summarizes the best known constructions of extractors:

reference	min-entropy k	output length m	additional randomness d
[GW97]	any k	$m = k$	$d = O(n - k + \log(1/\epsilon))$
[Zuc97]	$k = \Omega(n)$	$m = (1-\alpha)k$	$d = O(\log(n/\epsilon))$
[Tre99, RRV99b]	any k	$m = k^{1-\alpha}$	$d = O(\log^2 n / \log k + \log(1/\epsilon))$
[RRV99a, RRV99b]	any k	$m = (1-\alpha)k$	$d = O(\log^2 n + log(1/\epsilon))$
[RRV99a, RRV99b]	any k	$m = k$	$d = O((\log^2 n + \log(1/\epsilon)) \cdot \log k)$
optimal	any k	$m = k$	$d = O(\log(n/\varepsilon))$

Above, α is an arbitrarily small constant.

Figure 1: Best constructions of extractors

2.1.3 The Trevisan and RRV Extractors

Trevisan's Extractor The Trevisan extractor is based on the Nisan-Wigderson pseudorandom generator [NW94]. This generator is built out of any predicate P and its security is closely related to the hardness (on the average) of P. Let $S = (S_1, \ldots, S_m)$ be a collection of subsets of $[d]$, each subset of size l, that satisfy some nearly-disjointness property (these collections are called *standard designs*, see definition 4). Let $P: \{0,1\}^l \to \{0,1\}$ be any predicate. For a string $y \in \{0,1\}^d$, define $y|_{S_i}$ to be the string in $\{0,1\}^l$ obtained by projecting y onto the coordinates specified by S_i. Then the Nisan-Wigderson generator $NW_{S,P}: \{0,1\}^d \to \{0,1\}^m$ is defined as

$$NW_{S,P}(y) = P(y|_{S_1}) \cdots P(y|_{S_m}).$$

In the "indistinguishability proof" of [NW94], it is shown that for any function $D: \{0,1\}^m \to \{0,1\}$ which distinguishes the output of $NW_{S,P}(y)$ (for uniformly selected y) from the uniform distribution on $\{0,1\}^m$, there is a "small" circuit C (or procedure of small "description size") such that $C^D(\cdot)$ (i.e., C with oracle access to D) approximates $P(\cdot)$ reasonably well. Thus, a distinguisher D with small circuit size implies a function C with small circuit size which approximates P, in contrast to the assumption that P is "hard".

Roughly speaking, the pseudorandomness property of the output is due to the fact that each bit is determined by a different subset of the collection. Therefore, if we choose subsets that are nearly disjoint, we get small dependencies between the bits of the output and the output is pseudorandom. Specifically, the size of the circuit C is bounded by $m \cdot 2^{\max_{i \neq j} |S_i \cap S_j|}$, so we just need a collection such that $\max_{i \neq j} |S_i \cap S_j|$ is small.

We now give a rough description of the Trevisan extractor

$$EXT: \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m.$$

For a string $u \in \{0,1\}^n$, let $\overline{u} \in \{0,1\}^{\overline{n}}$ be an encoding of u in an error-correcting code and define $l = \log \overline{n}$. We view $\overline{u}$ as a Boolean function $\overline{u}: \{0,1\}^\ell \to \{0,1\}$. As above, we fix a collection $S = (S_1, \ldots, S_m)$ of nearly disjoint subsets of $[d]$ of size l (with properties as above). Then the extractor is simply

$$EXT_S(u, y) = NW_{S,\overline{u}}(y) = \overline{u}(y|_{S_1}) \cdots \overline{u}(y|_{S_m}).$$

Intuitively, the function is an extractor since we use a stronger assumption (that the predicate is *"weakly" random*, instead of *computationally hard*) and obtain a stronger conclusion (that the

output is *statistically close* to, rather than *indistinguishable*, from the uniform distribution). The analysis of this extractor in [Tre99] shows that the output of this extractor is close to uniform as long as the source min-entropy required is greater than the size of the circuit built in the security reduction of [NW94]. Hence, one needs to keep this circuit size small while minimizing the number d of truly random bits needed, which is equal to the seed length of the Nisan–Wigderson generator. Specifically, by using constructions of standard designs with certain parameters (see subsection 2.2.2), Trevisan [Tre99] obtains an extractor that works for low min-entropy ($n^{\Omega(1)}$) and extracts only a small fraction ($k^{1-\alpha}$) of the source min-entropy using only $O(\log n)$ truly random bits.

The RRV Improvements In [RRV99a] there are two main improvements to Trevisan's extractors: They observed that the size of the circuit C is actually bounded by $\max_i \sum_{j<i} 2^{|S_i \cap S_j|}$. Therefore, the standard designs which are used by Nisan-Wigderson are not the best option. In particular, they defined a different combinatorial design, called *weak design* in which $\max_i \sum_{j<i} 2^{|S_i \cap S_j|}$ is small (in contrast to *standard designs*, in which $\max_{i \neq j} |S_i \cap S_j|$ is bounded). They show that with weak designs, one can achieve better parameters than is possible with the corresponding standard designs. In particular, in some cases d is much smaller, which means that the extractors needs less truly random bits (see section 2.2).

This yields two improved extractors: The first extractor extracts a constant fraction of the min entropy. The weak designs underlying this construction are constructed using an application of the Probabilistic Method, which is derandomized using the Method of Conditional Expectations (see [ASE92] and [MR95] ch. 5). A simple iteration is applied to these weak designs to construct new weak designs, which are used to construct an extractor which improves upon previous construction in case we want to extract all the min entropy of the source.

The shortcome of these probabilistic constructions of weak designs is in the space complexity. The constructions require space which is polynomial in the number of subsets (see subsection 2.2.2). This implies that computing the RRV-extractor requires polynomial space. In this work we remedy this shortcome. We provide explicit constructions of weak designs, which are constructions that require space that is logarithmic in the number of subsets. In terms of parameters, these constructions match, up to constant factors, the probabilistic constructions given in [RRV99a]. Hence, we obtain extractors whose parameters are similar to those of the RRV extractor, but can be evaluated in logarithmic space.

The second improvement of [RRV99a] (which we will not deal with in this work) is achieved by using some specific error-correcting code rather than an arbitrary one. More specifically, they use multilinear error correcting codes over finite fields. This improves the dependency on the error parameter ϵ: The number of truly random bits d depends linearly on $\log(1/\epsilon)$ (improving upon the quadratic dependence in [Tre99]). In a more recent paper [RRV99b], they introduce a general method to obtain an even better dependency on ϵ (and the new method holds for any extractor).

2.2 Combinatorial Designs

2.2.1 Definitions

Roughly speaking, a combinatorial design is a collection of subsets of a given universe such that the subsets are "nearly disjoint". The difference between the types of designs is the definition of "nearly disjoint" (i.e., the condition restricting the intersections between the subsets).

The first combinatorial design, in which the intersection of every two subsets is bounded, was defined by Nisan and Wigderson:

Definition 4 (Standard Design [NW94]) *A family of sets* $S = S_1, \ldots, S_m \subset [d]$ *is called a* standard (m, l, ρ, d)-design *if*

1. *For all* $1 \leq i \leq m$, $|S_i| = l$.
2. *For all* $1 \leq i \neq j \leq m$, $|S_i \cap S_j| \leq \log \rho$.

Standard designs underly the construction of the Nisan-Wigderson generator and the Trevisan extractor. [RRV99a] observed that a weaker notion of combinatorial design, which they call *weak design* suffices for the constructions.

Definition 5 (Weak Design [RRV99a]) [1] *A family of sets* $S = S_1, \ldots, S_m \subset [d]$ *is called a* weak (m, l, ρ, d)-design *if*

1. *For all* $1 \leq i \leq m$, $|S_i| = l$.
2. *For all* $1 \leq i \leq m$,
$$\frac{\sum_{j<i} 2^{|S_i \cap S_j|}}{m} \leq \rho$$

Notice that every standard (m, l, ρ, d)-design is in particular a weak (m, l, ρ, d)-design. The advantage of weak designs is that they can be constructed with better parameters (see subsection 2.2.2). Our results (except theorem 9) actually hold for a stronger version of weak-designs, in which the second condition is:
For all $1 \leq i \leq m$,
$$\frac{\sum_{j \neq i} 2^{|S_i \cap S_j|}}{m} \leq \rho$$
However, for the sake of simplicity, we will not give a different name to this type of weak-designs.

In the following we consider some possible notions of "explicitness" in the construction combinatorial designs. We may require from the constructing algorithm to output *all* subsets of the design. In this view, a construction will be called *explicit* if the time and space needed is polynomial in the number of subsets. Another possibility is that the algorithm will get as an input an index to a subset and an index to an element, and will output the specified element. Here explicitness means that the time and space complexity is polynomial in the size of the input (the size is the logarithm of the number of subsets plus the logarithm of the size of each subset).

We will consider the following definition of explicitness. In this definition, the algorithm gets as an input an index to a subset, and is required to output the specified subset.

Definition 6 (Explicit Combinatorial Design) *A combinatorial design* $S = S_1, \ldots, S_m \subset [d]$, *where the size of each set is* l, *is called an* explicit *combinatorial design if there exists an algorithm that, given* $1 \leq i \leq m$, *outputs the subset* S_i *in time* $T = \text{poly}(\log m, l)$ *and space* $S = O(\log m)$.

Note that the complexity is not related to the parameter d, since typically $d = \text{poly}(\log m)$.

[1] This is a slight modification of the definition in [RRV99a], where the second condition is $\sum_{j<i} 2^{|S_i \cap S_j|} \leq \rho \cdot (m-1)$

2.2.2 Known Constructions

As mentioned above, the main motivation of defining the combinatorial designs is for construction of extractors (in standard designs, the original motivation was actually for pseudorandom generators). The parameters of the extractors are hence related to the parameters of the underlying designs. In this subsection, we state some known constructions of designs and see the parameters of the extractors they imply. In general, the relation between the parameters of the design and those of the extractor is as follows. The size of the universe of the design, d, is the number of truly random bits that the extractor needs, the size of each subset, l, is (roughly) the logarithm of the length of the input, n, and the parameter ρ is usually taken as $O(k/m)$ (k is the min entropy of the source, and m is the length of the output). For the sake of simplicity, in the following discussion we ignore ϵ, the error parameter of the extractor.

In constructing extractors, we would like to keep d (the number of truly random bits), as small as possible (as mentioned above, the best we can hope for is $d = O(\log n)$). This implies that we should minimize the size of the universe, d. In the following we will see that the main contribution of introducing the notion of weak designs [RRV99a] is in the fact that the universe is small, implying an extractor which requires a small number of truly random bits. We also want to extract as much randomness as possible, i.e., we want that m will be close to k. This implies that in the designs, ρ should be as close to 1 as possible.

Now we will describe the known constructions of combinatorial designs, and see the parameters of the extractors induced by these constructions. The first construction of standard designs is based on the random construction and achieves the same parameters.

Lemma 2.1 ([NW94]) *For every m, $l \in \mathsf{N}$, and $\rho > 1$, there exists a standard (m, l, ρ, d)-design $S_1, \ldots, S_m \subset [d]$ with*

$$d = O\left(\frac{l^2 m^{1/\log \rho}}{\log \rho}\right).$$

The design can be constructed in time $\mathrm{poly}(m, 2^d)$ *and space* $\mathrm{poly}(m)$.

This standard design yields an extractor [Tre99] that for any source of min entropy k, extracts m bits using $d = O\left(\frac{\log^2 n \cdot m^{1/O(\log k/m)}}{O(\log k/m)}\right)$ additional random bits. The most interesting case is when $m = k^{1-\alpha}$ (α is an arbitrary small constant), in which $d = O\left((\log^2 n)/(\log k)\right)$ (see figure 2.1.2). The algorithm for constructing the above design chooses sequentially (by exhaustive search) m subsets of $[d]$ such that each one intersects the previous ones in at most $\log \rho$ points. A probabilistic argument (using Chernoff bounds) shows that it is possible to choose m such subsets. Note that this construction requires the same time and space complexity even if we want to output only one subset and hence, it is not explicit in the sense of definition 6. However, we want to stress that this construction implies an indeed explicit construction of extractors, since in the extractor case, complexity is measured according to the input size of the extractor.

An explicit construction of standard designs with weaker parameters is given by the following:

Lemma 2.2 ([NW94]) *For every m,$l \in \mathsf{N}$ there exists an explicit standard (m, l, ρ, d)-design $S_1, \ldots, S_m \subset [d]$ with $d = O(l^2)$ and $\rho = m$. This design is explicit in the stronger sense, i.e., given $1 \leq i \leq m$, the algorithm outputs the subset S_i in time $T = \mathrm{poly}(\log m, l)$ and space $S = O(\log m)$.*

The construction, using polynomials over finite fields, is the construction we use in our main result (section 4). Roughly speaking, each subset is identified with a polynomial over a fixed finite

field. The subsets are constructed such that the intersection between the subsets contains the values in which the two polynomials (identified with the two subsets) agree on. The proof that these subsets induce a standard design is based on the fact that the number of values that two polynomials can agree on is bounded by their maximal degree.

For weak designs, we have the following construction:

Lemma 2.3 ([RRV99a]) *For every $l, m \in \mathsf{N}$ and $\rho > 1$, there exists a weak (m, l, ρ, d)-design $S_1, \ldots, S_m \subset [d]$ with*

$$d = \left\lceil \frac{l}{\ln \rho} \right\rceil \cdot l$$

This design can be constructed in time $\mathrm{poly}(m, d)$ *and space* $\mathrm{poly}(m)$.

This construction is much better than the construction of standard designs of lemma 2.1 in terms of the parameter d. Specifically, if ρ is constant, here $d = O(l^2)$, whereas $d = O(l^2 m^{\Omega(1)})$ in the corresponding standard design. In terms of extractors, if k/m is constant, the implied extractor needs only $O(\log^2 n)$ truly random bits (see figure 2.1.2), improving upon the $m^{\Omega(1)} \log^2 n$ bits needed in Trevisan's extractor in this setting of parameters.

We want to stress that the case that ρ is constant (or in terms of extractors k/m is constant) is a very important case. It means that we can extract a constant fraction of the source min-entropy and by using iterations [WZ95] on these extractors we obtain an extractor that extracts *all* the source min-entropy.

The proof is by a probabilistic argument: The universe $[d]$ is divided into l blocks, each block of size $\lceil l/\ln \rho \rceil$. The subsets are constructed sequentially such that each subset contains exactly one element from each block. It is shown that after choosing the subsets $S_1, \ldots S_{i-1}$, for every new subset S_i, the expected value of $\sum_{j<i} 2^{|S_i \cap S_j|}$ is smaller (or equal) to ρm. Hence, with non-zero probability, there exists a set S_i satisfying the requirements. The construction is derandomized by the Method of Conditional Expectations (see [ASE92, MR95]). In order to find the i^{th} subset we must compute and store all previous $i-1$ subsets (so we can calculate the appropriate conditional expectation). Thus, the algorithm requires time and space polynomial in m even if it is required to output only one specific subset. Hence, the construction is not explicit in the sense of definition 6 (however, as in lemma 2.1 it implies an explicit extractor).

The shortcome of the parameters in the above construction is that in case ρ is very close to 1, d gets very large. Specifically, if $\rho = 1 + \gamma$ for small γ, then the above construction gives $d = O(l^2/\gamma)$. This means that the induced extractor performs poorly (in terms of the number of additional truly random bits) if we want to extract all (or almost all) the min-entropy of the source. The following construction improves d significantly in this case.

Lemma 2.4 ([RRV99a]) *For every $l, m \in \mathsf{N}$ and $3/m \leq \gamma < 1/2$, there exists a weak $(m, l, 1+\gamma, d)$-design with*

$$d = O\left(l^2 \cdot \log \frac{1}{\gamma}\right).$$

This design can be constructed in time $\mathrm{poly}(m, d)$ *and space* $\mathrm{poly}(m)$.

The implied extractor needs $O(\log^2 n \cdot \log(1/\gamma))$ additional random bits, where $1+\gamma = k/(m-1)$. The most interesting case is when we want to extract all the min entropy of the source (i.e., $m = k$), in which the extractor requires only $O(\log^2 n \cdot \log k) = O(\log^3 n)$ additional truly random bits (see figure 2.1.2). This construction is accomplished by iterating the construction of lemma 2.3 (inspired by the iterations of Wigderson and Zuckerman [WZ95] on extractors).

2.3 Statement of Our Results

In this work we provide several explicit constructions of weak designs, which relate to the interesting cases of extractors discussed above. Our main result, which will be proved in section 4, is an explicit construction of weak designs for specific values of d and ρ.

Theorem 7 (Main) *For every $m,l \in \mathsf{N}$ there exists an explicit weak (m, l, ρ, d)-design, with $d = l^2$ and $\rho = e^2$ (where e denotes the base of the natural logarithm).*

This construction implies a logspace extractor that extracts a constant fraction $(1/e^2)$ of the source min entropy with only $O(\log^2 n)$ additional truly random bits. As discussed above, extractors that extract a constant fraction of the source min entropy are very important. By applying the iterations of Wigderson and Zuckerman [WZ95] on this extractor we get an extractor that extracts all the min entropy with $O(\log^3 n)$ random bits (without using the construction of theorem 9).

For the sake of completeness, we provide some more constructions of weak designs for more general cases. The following theorem generalizes the previous one in the sense that ρ is not restricted to the value e^2. This means that the induced extractor can extract any constant fraction of the min entropy using $O(\log^2 n)$ additional random bits. It provides explicit weak designs, which (in the interesting cases) match, up to a constant factor, the probabilistic construction of lemma 2.3:

Theorem 8 *For every $m,l \in \mathsf{N}$ and $\rho > 1$ there exists an explicit weak (m, l, ρ, d)-design such that*

1. *If $\rho = O(1)$ then $d = O(l^2)$.*
2. *If $\rho \neq O(1)$ and $2^l = O(m)$ then $d = O\left(\frac{l^2}{\ln \rho}\right)$.*

The following theorem states that weak designs with the same parameters as those of lemma 2.4 can be constructed explicitly:

Theorem 9 *For every $l, m \in \mathsf{N}$ and $3/m \leq \gamma < 1/2$, there exists an explicit weak $(m, l, 1+\gamma, d)$-design with*

$$d = O\left(l^2 \cdot \log \frac{1}{\gamma}\right).$$

This implies an extractor that extracts all the min entropy with $O(\log^3 n)$ random bits. Theorems 8 and 9 are proved in section 5.

An important application of the above explicit constructions of weak designs above is several explicit constructions of extractors that are similar to the RRV extractors [RRV99a, RRV99b] in terms of parameters and that can be evaluated in logarithmic space:

Theorem 10 *For every n, k, m and ϵ such that $m \leq k \leq n$ and $\epsilon > exp(-n/(log^* n)^{O(\log^* n)})$, there are explicit logspace (k, ϵ)-extractors $EXT : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with*

1. *if $k/m = O(1)$, $d = O\left(\log^2 n + \log(1/\epsilon)\right)$.*
2. *$d = O(\log(1/\gamma) \cdot (\log^2 n + log(1/\epsilon)))$, where $1 + \gamma = k/(m-1)$ and $1/m \leq \gamma < 1/2$.*

Proof: The evaluation of the RRV-extractor (and also the Trevisan extractor) has two main steps (see description in subsection 2.1.3). The first step is applying an error correcting code, which can be done in logspace (see Appendix A in [RRV99a]). The second step is constructing a weak design. For the first extractor we use the explicit weak designs of theorem 8, whereas the second uses those

of theorem 9. To calculate the i^{th} bit in the output of the extractor we must find the projection of the subset S_i on the codeword computed in the first step. Using our explicit weak designs, we can do this in logarithmic space. Overall, our extractor can be evaluated in logarithmic space, as desired. ■

2.4 Application to Derandomization of Space Bounded Computation

Let M be a logarithmic space Turing machine (or a polynomial width branching program) and assume that some weak estimation of the state probabilities is known. In [RR99], there is a construction of a logarithmic space pseudo-random generator that uses only logarithmic number of truly random bits and outputs a sequence of k bits that look random to M. The construction is based on linear space extractors which extract all the min-entropy of the source. By using the logspace extractor of theorem 10 (which is in particular linear space) we can slightly improve k (the length of the output of the pseudo-random generator). Details follow.

By assuming there are explicit linear space optimal extractors (i.e., linear space extractors which use only $d = O(logn + \log 1/\epsilon)$ truly random bits), the construction yields a pseudo-random generator with $k = 2^{\sqrt{\log n}}$. Using the linear space extractors of Ta-Shma [NT98], which uses $(\log n)^9$ truly random bits, [2] it was possible to prove an unconditional result with $k = 2^{\sqrt{\log n}/(\log\log n)^9}$. The RRV-extractor [RRV99a] uses only $(\log n)^3$ truly random bits, but is not a linear space extractor and therefore cannot be used in the construction. However, by using the logspace extractor of theorem 10, we can improve the degree of the polylog up to $k = 2^{\sqrt{\log n}/(\log\log n)^3}$. The above is formalized in the following corollary (see definitions of *layered branching program* and *estimator* in [RR99]):

Corollary 11 *Let $r = \sqrt{\log n}/(\log\log n)^3$. Let M be an $(n, 2^r)$-layered branching program, and let A be a $(2^{-3r-1}, r)$-estimator for M then there exists an RL Turing machine T such that:*

1. *The inputs for T are M and A.*
2. *T uses $O(\log n)$ random bits (and $O(\log n)$ space).*
3. *The output of T is a 2^{-r}-error pseudo-random sequence for M.*

3 Combinatorial-Algebraic Lemmas

A trivial counting shows that the number of polynomials with a bounded degree over a finite field is the size of the field to the power of the degree plus one. In this section we encounter the question of what happens to this number if we restrict the polynomials to be with a specific number of roots or a specific number of linear factors. The following lemmas provide answers to some versions of this question. The lemmas, especially lemmas 3.4 and 3.6 (which are the most interesting ones) can be viewed in the following way:

A fundamental theorem from algebra states that the number of roots of a (non-zero) polynomial is bounded by its degree. A simple observation shows that the expected number of roots of a random polynomial over a finite field is much smaller, namely, one. This can be seen by considering a uniformly selected polynomial (i.e., selecting uniformly the coefficients). For every element in the

[2] In [NT98] the number of truly random bits used by the extractors given as d =polylog n, without specifying the degree of the polynomial. However, Ta-Shma estimates the degree of this polynomial to be 9 (see [RRV99a]).

field, the probability that it is a root of the selected polynomial , which means that by evaluating the polynomial with this element we get zero, is one over the size of the field. Therefore, the expectation of the number of roots is one.

Lemma 3.4 shows that not only the expected number of roots of a random polynomial is one, but furthermore, this number is very concentrated around its expectation. Namely, the number of polynomials with many roots vanishes exponentially in the number of roots. This result is used for the explicit construction of weak designs in the next section and we believe it may be interesting in other applications as well.

We assume the reader is familiar with the following basic fact from algebra: $\alpha_1, \ldots, \alpha_k$ are roots of the polynomial $p(x)$ if and only if $\Pi_{i=1}^{k}(x - \alpha_i)$ divides $p(x)$ (It can be found in every standard text book on algebra, e.g. [Lan68] sec. 7, see also [Sud98] lec. 2).

In section 3.2 we state our lemmas and prove them using combinatorial arguments. In section 3.3 we prove lemma 3.6, which provides an upper bound which is slightly different than the one given in lemma 3.4. This proof is by a probabilistic argument.

3.1 Preliminaries

Let F be a finite field of size q. In the following, $F[x]$ is the ring of polynomials over F and $p(x) \in F[x]$. Define

- $deg(p) \stackrel{\text{def}}{=}$ the degree of $p(x)$.
- $\sharp L(p) \stackrel{\text{def}}{=}$ the number of linear factors in the factorization of $p(x)$ over F.
- $\sharp R(p) \stackrel{\text{def}}{=}$ the number of distinct roots of $p(x)$ in F (i.e., the number of elements $\alpha \in F$ such that $p(\alpha) = 0$).

Every distinct root over F implies a linear factor over F, thus, for every $p(x) \in F[x]$, it holds that $\sharp R(p) \leq \sharp L(p)$.
The following are subsets of $F[x]$:

1. $$L_q(k,d) \stackrel{\text{def}}{=} \{p(x) \in F[x] \mid (\sharp L(p) = k) \wedge (deg(p) \leq d) \wedge (p \neq \overline{0})\}$$
   ($L_q(k,d)$ is the set of non-zero polynomials over F of degree at most d that have exactly k linear factors over F. In other words, the polynomials have at most k roots, and the sum of their multiplicities is exactly k.) [3]

2. $$R_q(k,d) \stackrel{\text{def}}{=} \{p(x) \in F[x] \mid (\sharp R(p) = k) \wedge (deg(p) \leq d) \wedge (p \neq \overline{0})\}$$
   ($R_q(k,d)$ is the set of non-zero polynomials over F of degree at most d that have exactly k distinct roots in F. The multiplicity of the roots can be arbitrary.)

3. $$LR_q(k,d) \stackrel{\text{def}}{=} \{p(x) \in F[x] \mid (\sharp L(p) = k) \wedge (\sharp R(p) = k) \wedge (deg(p) \leq d) \wedge (p \neq \overline{0})\}$$
   ($LR_q(k,d)$ is the set of non-zero polynomials over F of degree at most d that have exactly k linear factors over F and exactly k distinct roots in F. An equivalent way is to say that the polynomials have exactly k distinct roots in F, all of them with multiplicity 1.)

[3] The zero polynomial is a special case in which there are q roots but no linear factors. Therefore, in these definitions,we prefer to define the sets without the zero polynomial.

Notice that by definition, $LR_q(k,d) = L_q(k,d) \cap R_q(k,d)$. Since having no linear factors over F is equivalent to having no roots in F, we get that $L_q(0,d) = R_q(0,d) = LR_q(0,d)$.

3.2 The Lemmas

The following lemmas regard the sizes of the sets defined above. In each lemma, the first part provides a formula for calculating the relevant value, whereas the second part provides an upper bound for this value. In the whole discussion, when we talk about linear factors of a polynomial, we assume, without loss of generality, that these factors are monic (i.e., in each factor, the coeffecient of x is 1). Therefore, we can identify a linear factor with an element of the field, i.e., each element $\alpha \in F$ is identified with the linear factor $(x - \alpha)$. Due to obvious reasons, we restrict ourselves to cases in which $d < q$, i.e., the degree of the polynomial is smaller than the size of the field.

Lemma 3.1

1.

$$|L_q(0,d)| = |R_q(0,d)| = |LR_q(0,d)| = \sum_{i=0}^{d}(-1)^i \cdot \binom{q}{i} \cdot (q^{d-i+1} - 1)$$

2.

$$|L_q(0,d)| = |R_q(0,d)| = |LR_q(0,d)| \leq q^{d+1} \cdot \left(\frac{1}{e} + \frac{2}{d!}\right)$$

Proof: As noticed above, $L_q(0,d) = R_q(0,d) = LR_q(0,d)$ and therefore, it suffices prove the lemma for $LR_q(0,d)$.

The formula in the first part is an inclusion-exclusion formula. The first summand is the total number of non-zero polynomials of degree at most d, namely $q^{d+1} - 1$. Now we want to subtract the number of non-zero polynomials that have at least one root (denoted by $\alpha \in F$). Every such polynomial is of the form $(x - \alpha) \cdot g(x)$, where $g(x)$ is a non-zero polynomial with degree at most $d - 1$. There are q possible values for α and $q^d - 1$ possibilities for $g(x)$. Hence, we subtract $q(q^d - 1)$. However, every polynomial with 2 roots we counted twice. Such polynomial has the form $(x - \alpha_1)(x - \alpha_2)g(x)$, where $\alpha_1, \alpha_2 \in F$ are the roots and $g(x)$ is a non-zero polynomial of degree at most $d - 2$. Hence, we add $\binom{q}{2}(q^{d-1} - 1)$. We continue this argument until we reach the maximal number of roots, namely, d. This proves the inclusion-exclusion formula in the first part of the lemma.

For the second part we will write the expression of the first part as a sum of two different sums and bound each sum separately:

$$\sum_{i=0}^{d}(-1)^i \cdot \binom{q}{i} \cdot (q^{d-i+1} - 1) = \sum_{i=0}^{d}(-1)^i \cdot \binom{q}{i} \cdot q^{d-i+1} + \sum_{i=0}^{d}(-1)^{i+1} \cdot \binom{q}{i}$$

In the following calculations, we assume d is odd. This will simplify the expressions bellow. Then we will show the case in which d is even.

First we bound

$$\sum_{i=0}^{d}(-1)^i \cdot \binom{q}{i} \cdot q^{d-i+1} = q^{d+1} \cdot \sum_{i=0}^{\frac{d-1}{2}} \frac{\binom{q}{2i}}{q^{2i}} - \frac{\binom{q}{2i+1}}{q^{2i+1}} =$$

$$q^{d+1} \cdot \sum_{i=0}^{\frac{d-1}{2}} \frac{q \cdot \frac{q!}{(2i)!(q-2i)!} - \frac{q!}{(2i+1)!(q-2i-1)!}}{q^{2i+1}} = q^{d+1} \cdot \sum_{i=0}^{\frac{d-1}{2}} \frac{\frac{q!}{(2i+1)!(q-2i)!} \cdot 2i(q+1)}{q^{2i+1}} =$$

$$q^{d+1} \cdot \sum_{i=0}^{\frac{d-1}{2}} \frac{2i \cdot \binom{q+1}{2i+1}}{q^{2i+1}} \leq q^{d+1} \cdot \sum_{i=0}^{\frac{d-1}{2}} \frac{2i}{(2i+1)!} =$$

$$q^{d+1} \cdot \sum_{i=0}^{\frac{d-1}{2}} \frac{1}{(2i)!} - \frac{1}{(2i+1)!} = q^{d+1} \cdot \sum_{i=0}^{d} \frac{(-1)^i}{i!} \leq q^{d+1} \cdot \frac{1}{e}$$

Where the last inequality is obtained since the last summand is odd.

In case d is even, we will first bound the above sum when i goes from 0 to $d-1$. The same calculations will hold, since $d-1$ is odd. Now we add the last summand, which is $\binom{q}{d} q \leq \frac{q^{d+1}}{d!}$. Therefore our upper bound, which holds for arbitrary d, is $q^{d+1} \cdot \left(\frac{1}{e} + \frac{1}{d!}\right)$.

Now we bound

$$\sum_{i=0}^{d} (-1)^{i+1} \cdot \binom{q}{i} = \sum_{i=0}^{d} (-1)^{i+1} \cdot \left[\binom{q-1}{i-1} + \binom{q-1}{i} \right] \leq \binom{q-1}{d} \leq q^{d+1} \cdot \frac{1}{d!}$$

Where the first inequality is since we have a telescopic sum.

Overall, we get

$$|LR_q(0,d)| \leq q^{d+1} \cdot \left(\frac{1}{e} + \frac{2}{d!}\right)$$

■

Lemma 3.2

1. $$|L_q(k,d)| = |L_q(0,d-k)| \cdot \binom{k+q-1}{k}$$

2. $$|L_q(k,d)| \leq q^{d+1} \cdot \frac{2^k}{k!} \left(\frac{1}{e} + \frac{2}{d!}\right)$$

Proof: Let $p(x) \in L_q(k,d)$ (i.e., $p(x)$ is a polynomial over F of degree at most d that has exactly k linear factors over F). Therefore, $p(x)$ is of the form $f(x) \cdot g(x)$, where $f(x)$ is a polynomial of degree k which is the product of the k monic linear factors of $p(x)$ (notice that since we require that the linear factors are monic, $f(x)$ is uniquely determined) and $g(x)$ is a polynomial of degree at most $d-k$ with no linear factors. Thus, $g(x) \in L_q(0,d-k)$. Ranging over all the possibilities for $f(x)$ and $g(x)$ we get all the possibilities of $p(x)$.

$f(x)$ is the product of k monic linear factors, where each linear factor is identified with one of the q elements of F (notice that the same linear factor may appear more than once in the product). Hence, the number of possibilities for $f(x)$ is the number of ways to choose a multi-set of k elements out of a universe of size q, namely, $\binom{k+(q-1)}{k}$ (it can be viewed as the number of possibilities of throwing k identical balls into q bins). Thus, we get,

$$|L_q(k,d)| = |L_q(0,d-k)| \cdot \binom{k+q-1}{k}$$

This proves the first part of the lemma.

For the second part of the lemma we first bound

$$\binom{k+q-1}{k} \leq \frac{(k+q-1)^k}{k!} \leq \frac{(2q)^k}{k!} = q^k \cdot \frac{2^k}{k!}$$

Where the second inequality is due to the fact that the number of linear factors, k, is obviously bounded by the degree of the polynomial, d, which is smaller than q. Finally, by using the upper bound of lemma 3.1, we have

$$|L_q(k,d)| = |L_q(0,d-k)| \cdot \binom{k+q-1}{k} \leq q^{d-k+1}\left(\frac{1}{e}+\frac{2}{d!}\right) \cdot q^k \cdot \frac{2^k}{k!} = q^{d+1} \cdot \frac{2^k}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right)$$

Comment: For small values of k (relative to q), a better bound can be obtained. First we bound

$$(k+q-1)^k \leq q^k \cdot (k/q+1)^k = q^k \cdot \left[(k/q+1)^{q/k}\right]^{k^2/q} \leq q^k \cdot e^{k^2/q}$$

Which implies

$$|L_q(k,d)| \leq q^{d+1} \cdot \frac{e^{k^2/q}}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right)$$

■

Lemma 3.3

1. $$|LR_q(k,d)| = |LR_q(0,d-k)| \cdot \binom{q}{k}$$

2. $$|LR_q(k,d)| \leq q^{d+1} \cdot \frac{1}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right)$$

Proof:

Let $p(x) \in LR_q(k,d)$ (i.e., $p(x)$ is a polynomial over F of degree at most d that has exactly k distinct roots in F and exactly k linear factors over F). Denote its k distinct roots by $\alpha_1, \ldots, \alpha_k$ (for every $1 \leq j \leq k, \alpha_j \in F$) and let

$$f(x) \stackrel{\text{def}}{=} \Pi_{i=1}^{k}(x-\alpha_i)$$

Obviously, $f(x)$ divides $p(x)$ and $p(x)$ has no other linear factors rather than the factors of $f(x)$. Hence, $p(x)$ is of the form $f(x) \cdot g(x)$, where $g(x)$ is a polynomial of degree at most $d-k$ with no roots. Clearly, $g(x) \in LR_q(0,d-k)$.

$f(x)$ is the product of k distinct linear factors, where each linear factor is identified with one of the q elements of F. Hence, the number of possibilities for $f(x)$ is the number of ways to choose k distinct roots out of the q elements of F, namely, $\binom{q}{k}$. Thus, we get,

$$|LR_q(k,d)| = |LR_q(0,d-k)| \cdot \binom{q}{k}$$

The second part is proved using lemma 3.1:

$$|LR_q(k,d)| = |LR_q(0,d-k)| \cdot \binom{q}{k} \leq q^{d-k+1}\left(\frac{1}{e}+\frac{2}{d!}\right) \cdot \frac{q^k}{k!} = q^{d+1} \cdot \frac{1}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right)$$

■

Lemma 3.4

1.
$$|R_q(k,d)| = \binom{q}{k} \cdot \sum_{i=0}^{d-k} |R_q(0,d-k-i)| \cdot \binom{i+k-1}{i}$$

2.
$$|R_q(k,d)| \leq q^{d+1} \cdot \frac{1+e/d!}{k!}$$

Proof: Let $p(x) \in R_q(k,d)$ (i.e., $p(x)$ is a polynomial over F of degree at most d that has exactly k distinct roots in F) and let $f(x)$ be as in lemma 3.3. $f(x)$ divides $p(x)$ and therefore, $p(x)$ has at least k linear factors. Let i be such that $p(x)$ has exactly $k+i$ linear factors (obviously, $0 \leq i \leq d-k$). Let $g(x)$ be the product of the other i linear factors of $p(x)$, rather than the ones in $f(x)$.

Hence, $p(x)$ is of the form $f(x) \cdot g(x) \cdot h(x)$. As in lemma 3.3, the number of possibilities for $f(x)$ is $\binom{q}{k}$. $g(x)$ is a product of i linear factors, where each linear factor is one of the k distinct linear factors of $f(x)$ (notice that the same linear factor can appear more than once). Thus, the number of possibilities for $g(x)$ is $\binom{i+k-1}{i}$ (the number of ways to choose a multi-set of i elements out of a universe of size k). $h(x)$ is a polynomial of degree at most $d-k-i$ with no roots in F, therefore, $h(x) \in R_q(0,d-k-i)$. Summing over all possible values of i, we get

$$|R_q(k,d)| = \sum_{i=0}^{d-k} |R_q(0,d-k-i)| \cdot |LR_q(k,k)| \cdot \binom{i+k-1}{i} = \binom{q}{k} \cdot \sum_{i=0}^{d-k} |R_q(0,d-k-i)| \cdot \binom{i+k-1}{i}$$

Now,

$$|R_q(k,d)| = \binom{q}{k} \cdot \sum_{i=0}^{d-k} |R_q(0,d-k-i)| \cdot \binom{i+k-1}{i} \leq$$

$$\frac{q^k}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right) \cdot \sum_{i=0}^{d-k} q^{d-k-i+1} \cdot \frac{(i+k-1)^i}{i!} \leq \frac{q^{d+1}}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right) \cdot \sum_{i=0}^{d-k} q^{-i} \cdot \frac{q^i}{i!} =$$

$$\frac{q^{d+1}}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right) \cdot \sum_{i=0}^{d-k} \frac{1}{i!} \leq q^{d+1} \cdot \frac{1+e/d!}{k!}$$

Where the first inequality is by lemma 3.1. i ranges between 0 and $d-k$ and therefore $i+k-1 \leq d$. Together with the fact that $d \leq q$ we get the second inequality.

Comments:

- A slightly better bound can be obtained. By bounding $i+k-1 \leq d$ (without using the fact that $d \leq q$), we get (as above)
$$|R_q(k,d)| \leq \frac{q^{d+1}}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right) \cdot \sum_{i=0}^{d-k} \frac{(d/q)^i}{i!} \leq q^{d+1} \cdot \frac{e^{d/q}}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right)$$
- Another bound: By bounding
$$(i+k-1)^i \leq k^i \cdot \left[(i/k+1)^{k/i}\right]^{i^2/k} \leq k^i \cdot e^{i^2/k}$$
we get
$$|R_q(k,d)| \leq \frac{q^{d+1}}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right) \cdot \sum_{i=0}^{d-k} \frac{\left(\frac{k \cdot e^{i/k}}{q}\right)^i}{i!} \leq q^{d+1} \cdot \frac{e^{\frac{k \cdot e^{(d-k)/k}}{q}}}{k!}\left(\frac{1}{e}+\frac{2}{d!}\right)$$

■

3.3 A Probabilistic Argument

In this section we prove a lemma which is a slightly different upper bound than the one given by lemma 3.4 (part 2). This bound will be used in our main result (section 4). This proof uses a probabilistic argument (in contrast to the algebraic-combinatorial arguments in section 3.2). This argument may be interesting in its own right, and may be generalized to prove some bounds regarding multivariate polynomials, in which the combinatorial argument is much more complicated (see discussion below).

The idea is to view the number of roots of a random polynomial as a sum of random variables. We will use the fact that there is some independence between the random variables to prove that the probability that a polynomial has k roots vanishes exponentially in k. Our technical tool will be the following lemma:

Lemma 3.5 ([SSS93], Theorem 1) *Let $X_1, \ldots, X_q \in \{0,1\}$ be random variables, $X = \sum_{i=1}^{q} X_i$ and $\mu = E[X]$. For every $k > 1$, if the X_i's are d-wise independent for*

$$d \geq h = h(q, \mu, k) \stackrel{\text{def}}{=} \left\lceil \frac{\mu(k-1)}{1-\mu/q} \right\rceil$$

then,

$$Pr(X \geq \mu k) \leq \frac{\binom{q}{h}(\mu/q)^h}{\binom{\mu k}{h}}$$

Intuitively, this lemma states that for proving that a probability vanishes exponentially (as in the Chernoff-Hoeffding bounds) it is enough to show limited independence (and not necessary full independence).

Lemma 3.6

$$|R_q(k,d)| \leq q^{d+1} \cdot \frac{1}{k!}$$

Proof: Denote the q elements of F by $a_1, \ldots, a_q$. Let $p(x) \in F[x]$ be a random polynomial (i.e., a polynomial whose coefficients where chosen uniformly from F) of degree at most d. For each $a_i \in F$ define an indicator random variable

$$X_i = \begin{cases} 1 & \text{if } p(a_i) = 0 \\ 0 & \text{otherwise} \end{cases}$$

Obviously, for every i, $E[X_i] = \frac{1}{q}$. Let $X = \sum_{i=1}^{q} X_i$ and $\mu = E[X] = 1$. It is well known that polynomials of degree d generate $(d+1)$-wise independent samples (see, e.g., [N92] ch. 16), and therefore the X_i's are $(d+1)$-wise independent. We want to upper bound the probability that the polynomial has at least k roots. Since $k \leq d < q$ we have

$$h = h(q,1,k) = \left\lceil \frac{k-1}{1-1/q} \right\rceil = \left\lceil \frac{(q-1)(k-1)+(k-1)}{q-1} \right\rceil = k \leq d.$$

Hence, we can apply lemma 3.5 and get

$$Pr(X \geq k) \leq \frac{\binom{q}{k}(1/q)^k}{\binom{k}{k}} \leq \frac{1}{k!}$$

This means that the fraction of polynomials with at least k roots is at most $\frac{1}{k!}$. Formally, [4]

$$|R_q(k,d)| \leq q^{d+1} \cdot \frac{1}{k!}$$

which proves the lemma. [5] ■

A similar probabilistic argument can be used to prove an analogous result regarding multivariate polynomials. In this case, the algebraic-combinatorial argument is much more complicated. Roughly speaking, this is because in multivariate polynomials, it is not true that a root implies a linear factor, as in the univariate case (this fact is the basis of the algebraic-combinatorial arguments in the univariate case).

As above, the random variables are d-wise independent, but here d is the common degree of all the variables in the polynomial. Since the number of roots k, can be much more than d (it is bounded by dq^{n-1}, where n is the number of variables), we cannot use lemma 3.5. Instead, we use a different lemma, which requires less independence between the random variables:

Lemma 3.7 ([SSS93], Lemma 3) *Let $X_1, \ldots, X_q \in \{0,1\}$ be random variables , $X = \sum_{i=1}^{q} X_i$ and $\mu = E[X]$. For every $k > 1$, if the X_i's are d-wise independent for $d < h(q,\mu,k)$ then,*

$$Pr(X \geq \mu k) \leq \frac{\binom{q}{d}(\mu/q)^d}{\binom{\mu k}{d}}$$

This yields the following result: The fraction of multivariate polynomials over F of common degree at most d with n variables that have at least k roots is bounded by

$$\left(\frac{q^{n-1}}{k}\right)^d.$$

4 Main Result

In this section we prove our main result (theorem 7), which is an explicit construction of weak-designs with $d = l^2$ and $\rho = e^2$. The construction is essentially the same as the explicit construction of standard designs with weak parameters in lemma 2.2. Our contribution is a finer analysis, based on the lemmas of section 3, which shows that the same construction actually yields a weak design with better parameters. Roughly speaking, each subset is identified with a polynomial over a fixed finite field. The subsets are constructed such that the intersection between the subsets contains the values in which the two polynomials (identified with the two subsets) agree on. The proof that these subsets induce a weak design that the number of values that two polynomials can agree on is bounded by their maximal degree. This fact suffices for proving the standard designs property, in which we have to bound the intersection between every two subsets.

However, a much more powerful statement can be said on the number of roots of a polynomial: not only that the number is bounded by its degree, but the average number of roots is 1 and, furthermore, the average of the exponentiation of the number of roots is constant (as formalized in lemmas 3.6 and 3.4). This resembles the definition of weak designs and explains, intuitively, why the construction yields a weak design. We restate our main result and then prove it:

[4] The fact that lemma 3.5 works only for $k > 1$ does not bother us since this bound is trivial for $k = 0, 1$.

[5] We actually proved a slightly stronger result than the one stated in the lemma: we bounded the number of polynomials with *at least* k roots (and not *exactly* k roots).

Theorem 12 (theorem 7, restated) For every [6] $m,l \in \mathsf{N}$ there exists an explicit weak (m, l, ρ, d)-design, with $d = l^2$ and $\rho = e^2$ (where e denotes the base of the natural logarithm).

Proof: *The Construction:* Let $F = GF(l)$ (recall that we assume that l is a prime power). Since $d = l^2$, we can identify, in the natural way, each number in the range $[d]$ with an ordered pair of elements in F. Thus, constructing the design is equivalent to constructing subsets of the set $F^2 = \{< a, b >: a, b \in F\}$.

The subsets will be defined as follows: Given a polynomial $p(x) \in F[x]$, define the subset

$$S_p \stackrel{\text{def}}{=} \{< a, p(a) >: a \in F\} \subseteq F^2$$

The weak design will consist of all such subsets which are indexed by polynomials over F of degree at most $\frac{\log m}{\log l} - 1$ (we assume, for simplicity, that $\frac{\log m}{\log l}$ is an integer). Formally, let $deg \stackrel{\text{def}}{=} \frac{\log m}{\log l} - 1$ and let

$$P \stackrel{\text{def}}{=} \{p(x) \in F[x] : deg(p) \leq deg\}$$

The weak design S is defined as $S \stackrel{\text{def}}{=} \{S_p\}_{p \in P}$.

In the following claim we show that S is a weak-design:

Claim 4.1 *S is a weak (m, l, ρ, d)-design.*

Proof: We have to show that the properties of weak designs hold:

- All sets S_p are subsets of F^2, which is identified with $[d]$.
- The size of each subset S_p is l.
- The number of subsets is the size of P (i.e., the number of polynomials over F with degree at most deg), namely,
$$|P| = l^{\,deg+1} = l^{\,(\frac{\log m}{\log l} - 1)+1} = m$$
- For every $p \in P$, we have to show that [7]
$$\sum_{q \in P,\ q \neq p} 2^{|S_p \cap S_q|} \leq \rho \cdot m$$

  The key idea is that an element $< a, b > \in F^2$ is in the intersection of S_p and S_q if and only if $p(a) = q(a) = b$, i.e., p and q agree on a. Obviously, two polynomials agree on a value if and only if this value is a root of the difference polynomial. Hence, the size of the intersection of S_p and S_q is equal to the number of distinct roots of the polynomial $p - q$. Formally, $|S_p \cap S_q| = \sharp R(p - q)$.

  Therefore, ranging over all the polynomials q such that $q \in P,\ q \neq p$ and considering the size of $S_p \cap S_q$ (as needed to calculate the above sum) is equivalent to ranging over all the polynomials r (which represent the difference polynomials) such that $r \in P,\ r \neq \overline{0}$ and considering $\sharp R(r)$. Hence, we have reduced the problem of calculating the sizes of the intersections into the

[6] We assume, without loss of generality ,that l is a prime power. If l is not a prime power, pick, e.g., the smallest power of 2 which is greater than l. This, at most, doubles the value of l.

[7] As noted above, we show how to construct a weak-design of the stronger version, in which for all $1 \leq i \leq m$, $\frac{\sum_{j \neq i} 2^{|S_i \cap S_j|}}{m} \leq \rho$.

problem of counting how many polynomials are there with each possible number of distinct roots. This problem is solved in lemma 3.6.

For every $p \in P$,

$$\sum_{q \in P, q \neq p} 2^{|S_p \cap S_q|} = \sum_{q \in P, q \neq p} 2^{\sharp R(p-q)} = \sum_{r \in P, r \neq \overline{0}} 2^{\sharp R(r)} = \sum_{k=0}^{deg} 2^k \cdot |R_l(k, deg)| \leq$$

$$\sum_{k=0}^{deg} 2^k \cdot (l^{deg+1} \cdot \frac{1}{k!}) = m \cdot \sum_{k=0}^{deg} \frac{2^k}{k!} \leq e^2 \cdot m = \rho \cdot m$$

Where the first inequality is obtained by using the upper bound in lemma 3.6. Using the bound in the first comment in the proof of lemma 3.4 we can have $\rho = e^2 \cdot e^{d/q} \left(\frac{1}{e} + \frac{2}{d!}\right)$, which, in most cases, is a better bound. However, for the sake of simplicity, we prefered to state the theorem using the bound of lemma 3.6.

■

Proving the following claim completes the proof of the lemma:

Claim 4.2 *The above construction is explicit.*

Proof: We have to describe an algorithm that given $1 \leq i \leq m$, outputs the set S_i in time $T = \text{poly}(\log m)$ and space $S = O(\log m)$. Enumerate all polynomials over F of degree at most deg by enumerating each one of the $deg + 1$ coefficients separately (of course any other standard enumeration will do as well). Given i, divide its $\log m$ bits into $\frac{\log m}{\log l} = deg + 1$ parts, each part consists of $\log l$ bits. The k^{th} part represents the k^{th} coefficient of the i^{th} polynomial. For every $a \in GF(l)$, we pair a with the evaluation of the polynomial on a and translate the pair to an element in the range $[d]$. The set S_i consists of all these values.Obviously, this algorithm runs in time $T = \text{poly}(\log m)$ and space $S = O(\log m)$. ■

Comment: The above construction actually works only for cases in which $m \leq l^{l-1}$, since the degree of the polynomials must be smaller than the size of the field (this is a necessary condition for the lemmas of section 3). [8] However, if we want more subsets, we can take again the same subsets. Say we need $m = c \cdot l^{l-1}$ subsets for some constant c, then for every $1 \leq i \leq m$

$$\sum_{j \neq i} 2^{|S_i \cap S_j|} \leq \rho \cdot l^{l-1} c + c 2^l \leq (\rho + (2/l)^l) l^{l-1} c$$

This means we can increase the number of subsets to an arbitrary number, paying only an exponentially additive cost in the value of ρ. ■

5 The General Constructions

In order to obtain the explicit constructions of weak designs as stated in theorems 8 and 9, we first (in section 5.1) develop some tools for manipulating weak designs. Then, in section 5.2, we combine these tools with the explicit construction of theorem 7 to get the desired constructions.

[8] This comment is irrelevant for weak designs that are used in the construction of extractors, since in that case, m (which is the number of extracted almost uniform bits) is $O(2^l)$ (l is the logarithm of the length of the source).

5.1 Tradeoff Between the Values of ρ and d

In the previous section we constructed an explicit weak (m, l, ρ, d)-design for specific values of d and ρ ($d = l^2$ and $\rho = e^2$). However, theorem 8 is stated for every $\rho > 1$ and with $d = O\left(\frac{l^2}{\ln \rho}\right)$. In this section we supply tools to decrease the value of ρ while increasing the value of d, and vice versa.

5.1.1 Decreasing the Value of ρ

The following lemma is a tool for decreasing the value of ρ, i.e., given an explicit weak-design with some value of ρ we show how to construct an explicit weak-design with a smaller value of ρ. This is done while maintaining the value of l. A smaller ρ means that the bound on the average of the exponentiations of the intersections between the subsets is smaller. To accomplish this goal, we are allowed to use a bigger universe (i.e., increase the value of d).

The idea of the construction (inspired by the "packing" of weak designs in [RRV99a]) is to take an ordered union of weak-designs, each design on a different disjoint subset of the universe (this is possible since we are allowed to use a bigger universe). Roughly speaking, the parameter ρ is getting smaller since in the "unioned" weak-design, two subsets that are not taken from the same "original" weak-design, are disjoint. Two subsets taken from the same "original" weak-design have relatively big intersection (since it is related to the original ρ, which is big). Nevertheless, there are few such pairs and therefore they don't contribute too much to the total sum. Overall, we get a small ρ, as desired.

Lemma 5.1 *If for $l, m, d \in \mathsf{N}$ and $\rho' > 1$ there exists an explicit weak (m, l, ρ', d)-design, then for every $1 < \rho \leq \rho'$ there exists an explicit weak (mn, l, ρ, dn)-design, where $n = \frac{\rho'-1}{\rho-1}$.*

Proof: For simplicity, assume that n is an integer. We view $[dn]$ as the disjoint union of n blocks $B_1, \ldots B_n$ each of size d. Our weak design $S_1, \ldots, S_{mn}$ will be defined as follows: For each $t \in [n]$ we let $S_{(t-1)m+1}, \ldots, S_{tm} \subset B_t$ be a copy of a weak (m, l, ρ', d)-design (as given in the lemma). In other words, we take an ordered union of n weak (m, l, ρ', d)-designs, using disjoint subsets of the universe for each. In our new design, the size of each set is l, the number of sets is mn and the size of the universe is dn.

Now we have to show that for all $1 \leq i \leq mn, \sum_{j \neq i} 2^{|S_i \cap S_j|} \leq \rho \cdot mn$. For every $t \in [n]$, denote the collection of subsets $\{S_{(t-1)m+1}, \ldots, S_{tm}\}$ by $\widehat{S_t}$. Notice that if $i \in \widehat{S_t}$ then for every $j \notin \widehat{S_t}$, S_i and S_j are disjoint and

$$\sum_{j \in \widehat{S}_t, j \neq i} 2^{|S_i \cap S_j|} \leq \rho' \cdot m$$

since $\widehat{S_t}$ is a weak (m, l, ρ', d)-design. Thus, for every $t \in [n]$ and every $i \in \widehat{S_t}$, we have

$$\sum_{j \neq i} 2^{|S_i \cap S_j|} = \sum_{j \in \widehat{S}_t, j \neq i} 2^{|S_i \cap S_j|} + \sum_{j \notin \widehat{S}_t} 2^{|S_i \cap S_j|} \leq$$

$$\rho' \cdot m + (n-1)m = (\frac{\rho' - 1}{n} + 1) \cdot mn = \rho \cdot mn$$

The design remains explicit: Given the index i of a subset, we output the $i (\mathrm{mod} n)^{th}$ subset over the block $B_{\lceil i/n \rceil}$ (the subset can be constructed explicitly since the weak-design in each block is explicit).

■

5.1.2 Decreasing the Value of d

The following lemma is a tool for decreasing the value of d, while increasing the value of ρ. The idea of the construction is quite similar to the idea in lemma 5.1, although the opposite goal is achieved. The reason for this is that in the two constructions we indeed increase the value of d, however, in the following construction we increase simultaneously the values of d and l by the same factor. Hence, the value of d is decreased relative to l. It is achieved by taking direct products of many weak-designs.

For simplifying the following lemma, we will define a slightly stronger version of weak-designs: A *full weak* (m, l, ρ, d)*-design* is a weak-design (see definition 5), where the second condition is replaced by the following one: For all $1 \leq i \leq m$,

$$\frac{\sum_{j=1}^{m} 2^{|S_i \cap S_j|}}{m} \leq \rho$$

Obviously, every full weak-design is in particular a weak-design.

Lemma 5.2 *If for* $l, m, d \in \mathsf{N}$ *and* $\rho' > 1$ *there exists an explicit full weak* (m, l, ρ', d)*-design, then for every* $\rho \geq \rho'$ *there exists an explicit full weak* $(m^n, l \cdot n, \rho, d \cdot n)$*-design, where* $n = \frac{\ln \rho}{\ln \rho'}$.

Proof: We assume, for simplicity, that n is an integer. We view $[dn]$ as the disjoint union of n blocks $B_1, \ldots B_n$ each of size d. For each $t \in [n]$ let $S^t = S_1^t, \ldots, S_m^t \subset B_t$ be an explicit full weak (m, l, ρ', d)-design . Our design, S, will be the direct product of all the n designs:

$$S \stackrel{\text{def}}{=} \bigotimes_{t=1}^{n} S^t.$$

In other words, each subset in S is a union of n subsets, each one from a different "small" design. We now show that S is an explicit full weak $(m^n, l \cdot n, \rho, d \cdot n)$-design. It is easy to verify that each subset is of size $l \cdot n$, the number of subsets is m^n and the size of the universe is dn. To complete the proof we have to show that for all $1 \leq i \leq m^n$, $\sum_{j=1}^{m} 2^{|S_i \cap S_j|} \leq \rho \cdot m^n$.

We will index each subset in S by an n-tuple, $(i_1, \ldots, i_n) \in \{1, \ldots, m\}^n$, such that $S_{(i_1,\ldots,i_n)} = \bigcup_{t=1}^{n} S_{i_t}^t$. For all $i = (i_1, \ldots, i_n)$,

$$\sum_{j=(j_1,\ldots,j_n)} 2^{|S_i \cap S_j|} = \sum_{j_1} \cdots \sum_{j_n} 2^{\sum_{k=1}^{n} |S_{i_k} \cap S_{j_k}|} =$$

$$\sum_{j_1} \cdots \sum_{j_n} \prod_{k=1}^{n} 2^{|S_{i_k} \cap S_{j_k}|} = \sum_{j_1} \cdots \sum_{j_{n-1}} \prod_{k=1}^{n-1} 2^{|S_{i_k} \cap S_{j_k}|} \sum_{j_n} 2^{|S_{i_n} \cap S_{j_n}|} \leq$$

$$\sum_{j_1} \cdots \sum_{j_{n-1}} \prod_{k=1}^{n-1} 2^{|S_{i_k} \cap S_{j_k}|} \rho' \cdot m \leq \sum_{j_1} \cdots \sum_{j_{n-2}} \prod_{k=1}^{n-2} 2^{|S_{i_k} \cap S_{j_k}|} (\rho')^2 \cdot m^2 \leq$$

$$\leq \cdots \leq (\rho')^n \cdot m^n = \rho \cdot m^n$$

Where the inequalities are obtained by the fact that each coordinate is a full weak (m, l, ρ', d)-design.

The design remains explicit: The algorithm gets an index i to a subset. First we find the n-tuple $(i_1, \ldots, i_n)$ which i represents. The j^{th} element is from the $k \stackrel{\text{def}}{=} \lceil j/n \rceil$ design, specifically, it is the $j(\text{mod}n)^{th}$ element of the subset $S_{i_k}^k$. Our subset contains all such elements for $1 \leq j \leq l \cdot n$. Since n is smaller than the logarithm of the number of subsets (m^n), the construction remains explicit. ■

5.2 The Constructions

We now have the tools for proving theorems 8 and 9:

Theorem 13 (theorem 8, restated) *For every $m,l \in \mathsf{N}$ and $\rho > 1$ there exists an explicit weak (m, l, ρ, d)-design such that*

1. *If $\rho = O(1)$ then $d = O(l^2)$.*
2. *If $\rho \neq O(1)$ and $2^l = O(m)$ then $d = O\left(\frac{l^2}{\ln \rho}\right)$.*

Proof: We prove each part separately:

1. If ρ is a constant bigger than e^2, this part follows trivially from theorem 7, since a weak (m, l, e^2, l^2)-design is also a weak (m, l, ρ, l^2)-design for $\rho \geq e^2$. Otherwise, given $l, m \in \mathsf{N}$ and $\rho < e^2$, we let $n = \frac{e^2-1}{\rho-1} = O(1)$. By applying lemma 5.1 on an explicit weak $(\frac{m}{n}, l, e^2, l^2)$-design (guaranteed by theorem 7) we get the desired explicit weak $(m, l, \rho, O(l^2))$-design.
2. Given $l, m \in \mathsf{N}$ and $\rho > e^2$, we let $\alpha > 1$ be a constant such that $2^l \leq \alpha m$ and let $n = \frac{\ln \rho}{\ln(e^2+\alpha)} = O(\ln \rho)$. We start with an explicit full $\left(m^{1/n}, \frac{l}{n}, e^2 + \alpha, O((\frac{l}{n})^2)\right)$-design (whose existence will be proved later on). By applying lemma 5.2 on this design we get the desired explicit (full) weak $\left(m, l, \rho, O(\frac{l^2}{\ln \rho})\right)$-design.

   It remains to prove the existence of the explicit full weak $\left(m^{1/n}, \frac{l}{n}, e^2 + \alpha, O((\frac{l}{n})^2)\right)$-design used above. Since $2^{l/n} \leq \alpha^{1/n} m^{1/n} \leq \alpha m^{1/n}$, the above design is guaranteed by the following simple claim:

   Claim 5.3 *For every $m', l' \in \mathsf{N}$ such that $2^{l'} = O(m')$ and for $\rho' = e^2 + \alpha$ (α is such that $2^{l'} \leq \alpha m'$) there exists an explicit full weak (m', l', ρ', d')-design where $d' = l'^2$.*

   Proof: The construction is the same as in the proof of theorem 7. For all $1 \leq i \leq m'$,

$$\sum_j 2^{|S_i \cap S_j|} = \sum_{j \neq i} 2^{|S_i \cap S_j|} + 2^{|S_i \cap S_i|} \leq e^2 m' + 2^{l'} \leq (e^2 + \alpha) m' = \rho' m'$$

   ■

■

Theorem 14 (theorem 9, restated) *For every $l, m \in \mathsf{N}$ and $3/m \leq \gamma < 1/2$, there exists an explicit weak $(m, l, 1 + \gamma, d)$-design with*

$$d = O\left(l^2 \cdot \log \frac{1}{\gamma}\right).$$

Proof: The proof follows the construction of weak-designs via "packing" in [RRV99a]. The only difference is that we use explicit weak $(m, l, 2, O(l^2))$-designs, rather than non-explicit ones. This makes the construction explicit, as desired. For the sake of completeness, we restate the construction.

For simplicity, assume that $1+\gamma = 1/(1-2^{-h})$ and $m = 2^q/(1+\gamma)$. Let $d_0 = O(l^2)$ and let $d = h \cdot d_0 = O(l^2 \cdot \log(1+\gamma))$. We view $[d]$ as the disjoint union of h blocks $B_1, \ldots, B_h$ each of size d_0. For each $t \in [h]$, let $m_t = 2^{q-t}$ and $n_t = \sum_{s=1}^{t-1} m_s$, so $\sum_t m_t = m$.

Now we define our weak design $S_1, \ldots, S_m$. For each $t \in [h]$, we let $S_{n_t+1}, \ldots, S_{n_t+m_t} \subset B_t$ be an explicit weak $(m_t, l, 2, d_0)$-design as guaranteed by theorem 8. In other words, we take the ordered union of h weak $(m_t, l, 2, d_0)$-designs using disjoint subsets of the universe for each. The number of sets is m, the size of the universe is d, and each set is of size l, so we only need to check that for all $i \in [m]$, $\sum_{j<i} 2^{|S_i \cap S_j|} < \rho \cdot (m-1)$. For $i \in \{n_t+1, \ldots, n_t+m_t\}$, S_i is disjoint from any S_j for any $j \leq n_t$ and

$$\sum_{j=n_t+1}^{i-1} 2^{|S_i \cap S_j|} \leq 2 \cdot (m_t - 1)$$

since $S_{n_t+1}, \ldots, S_{n_t+m_t}$ is a weak $(m_t, l, 2, d_0)$-design. Thus, we have

$$\begin{aligned} \sum_{j<i} 2^{|S_i \cap S_j|} &= \sum_{j=1}^{n_t} 2^{|S_i \cap S_j|} + \sum_{j=n_t+1}^{i-1} 2^{|S_i \cap S_j|} \\ &\leq n_t + 2 \cdot (m_t - 1) \\ &= 2^q - 2 < (1+\gamma)(m-1), \end{aligned}$$

as desired.

We now describe an algorithm that constructs explicitly the design: Given an index i of a subset, we find the value of $t \in [h]$ such that $n_t < i \leq n_{t+1}$. Our subset S_i is a subset of B_t indexed by $i - n_t$ in the t^{th} design, and it can be computed by the algorithm of claim 4.2. Scanning all $t \in [h]$ takes time $T = O(h) = O(\log \frac{1}{\gamma}) = O(\log m)$, where the last equality is due to the fact that $3/m \leq \gamma$. Therefore, the algorithm runs in time $\text{poly}(\log m)$ and space $O(\log m)$ and the construction remains explicit. ∎

6 Final Discussion and Open Problems

The main contribution of this work is the realization that the number of roots of a polynomial is not only bounded by its degree (a fact used in a wide variety of applications), but, typically, is much smaller. Specifically, the number of roots is exponentially concentrated around its average, which is 1. In this work, we used this fact to construct explicit weak-designs. However, this fact may be interesting in other applications as well.

For example, the Reed-Solomon error correcting code (see, e.g., [MS77]) is based on the fact that the number of roots of a polynomial is bounded by its degree. This implies that the minimal Hamming distance between every two codewords is $n-d$, where n is the length of the codeword and d is the degree of the polynomial that generated the code. In the shed of our results, the following observation can be made: The average Hamming distance between two codewords is $n-1$, and furthermore, for every codeword the number of codewords which are close to it is exponentially small.

Another possible way to construct explicit weak-designs for the case that d is smaller than $O(l^2)$ and ρ is big, is by considering multivariate polynomials, analogously to the univariate polynomials used in the construction in the proof of theorem 7. For every multivariate polynomial with n variables $p(x_1, \ldots, x_n)$ we define the subset

$$S_p \stackrel{\text{def}}{=} \{< a_1, a_2, \ldots, a_n, p(a_1, \ldots, a_n) >: \forall 1 \leq i \leq n \; a_i \in F\} \subseteq F^{n+1}$$

As in the univariate case, the quality of the design is related to the number of multivariate polynomials over F of common degree at most d with n variables that have at least k roots. The upper bound presented in section 3.3) does not suffice for proving the weak designs property. Proving a better bound may obtain the result.

Another open problem is to find explicit construction of standard designs (and maybe of other combinatorial designs as well). The constructions may be based on algebraic structures (as our construction), or may be derived from the explicit construction with weak parameters in lemma 2.2 by combinatorial techniques (e.g., the union of subsets and the direct product in section 5.1).

References

[ASE92] Noga Alon, Joel H. Spencer, and Paul Erdős. *The Probabilistic Method.* Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley and Sons, Inc., 1992.

[CG88] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, April 1988.

[GW97] Oded Goldreich and Avi Wigderson. Tiny families of functions with random properties: A quality-size trade-off for hashing. *Random Structures & Algorithms*, 11(4):315–343, 1997.

[GZ97] Oded Goldreich and David Zuckerman. Another proof that BPP $\subseteq$ PH (and more). *Electronic Colloquium on Computational Complexity* Technical Report TR97-045, September 1997. `http://www.eccc.uni-trier.de/eccc`.

[Lan68] Serge Lang. *Algebraic Structures.* Addison-Wesely, 1968.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms.* Cambridge University Press, 1995.

[MS77] Florence J. MacWilliams and Neil J. A. Sloane. *The Theory of Error-Correcting Codes.* North-Holland, 1977.

[N92] Joseph (Seffi) Naor. Probabilistic Methods in Computer Science. Lecture notes, Technion, Spring 1992. Available at `http://www.uni-paderborn.de/fachbereich/AG/agmadh/WWW/english/scripts.html`

[NT98] Noam Nisan and Amnon Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Sciences*, 1998. To appear in STOC '96 special issue.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, October 1994.

[NZ96] Noam Nisan and David Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, February 1996.

[RR99] Ran Raz and Omer Reingold. On recycling the randomness of the states in space bounded computation. *Proceedings of the 31st ACM Symposium on Theory of Computing*, Atlanta GA, May 1999.

[RRV99a] Ran Raz, Omer Reingold and Salil Vadhan. Extracting all the randomness and reducing the error in Trevisan's extractors. *Proceedings of the 31st ACM Symposium on Theory of Computing*, Atlanta GA, May 1999.

[RRV99b] Ran Raz, Omer Reingold and Salil Vadhan. Error reduction for extractors. In *40th Annual Symposium on Foundations of Computer Science*, October 1999. IEEE.

[SSS93] Jeanette P. Schmidt, Alan Siegel and Aravind Srinivasan. Chernoff-Hoeffding bounds for applications with limited independence. *Proceedings of the 4th Annual ACM Symposium on Discrete Algorithms*, 1993, pp. 331-340.

[STV99] Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. In *Proceedings of the 31th Annual ACM Symposium on Theory of Computing*, Atlanta GA, May 1999. ACM.

[Sud98] Madhu Sudan. Algebra and Computation. Lecture notes, MIT, Fall 98. Available at `http://theory.lcs.mit.edu/~madhu/FT98/course.html`

[SV86] Miklos Santha and Umesh V. Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of Computer and System Sciences*, 33(1):75–87, August 1986.

[SZ98] Aravind Srinivasan and David Zuckerman. Computing with very weak random sources. To appear in *SIAM Journal on Computing*, 1998. Preliminary version in *FOCS '94*.

[Tre99] Luca Trevisan. Constructions of extractors using pseudo-random generators. In *Proceedings of the 31th Annual ACM Symposium on Theory of Computing*, Atlanta GA, May 1999. ACM.

[VV85] Umesh V. Vazirani and Vijay V. Vazirani. Random polynomial time is equal to slightly-random polynomial time. In *26th Annual Symposium on Foundations of Computer Science*, pages 417–428, Portland, Oregon, 21–23 October 1985. IEEE.

[WZ95] Avi Wigderson and David Zuckerman. Expanders that beat the eigenvalue bound: Explicit construction and applications. Technical Report CS-TR-95-21, University of Texas Department of Computer Sciences, 1995. To appear in *Combinatorica*.

[Zuc90] David Zuckerman. General weak random sources. In *Proceedings of the 31th Symposium on Foundations of Computer Science*, pages 534–543, 1990. IEEE.

[Zuc96] David Zuckerman. Simulating BPP using a general weak random source. *Algorithmica*, 16(4/5):367–391, October/November 1996.

[Zuc97] David Zuckerman. Randomness-optimal oblivious sampling. *Random Structures & Algorithms*, 11(4):345–367, 1997.

ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'