Tel-Aviv University
Raymond and Beverly Sackler
Faculty of Exact Sciences

Derandomization of Families of Entropy-Preserving Functions

This thesis was submitted in partial fulfillment of the requirements for the master degree (M.Sc.) at the Tel-Aviv University

School of Computer Science

Submitted by
Eyal Kaplan

This work was supervised by Dr. Amnon Ta-Shma

Abstract

In probabilistic computations, random bits are viewed as a resource. An important goal is to consume as little of it as possible. We study functions, which decrease the number of random bits needed by an algorithms.

Loss-less condensers take as input a low entropy source, and output a source, which is statistically close to a high entropy source. This closeness is referred to as the error of the condenser. We show a general error reduction for loss-less condensers, building on the error reduction technique of Raz, Reingold and Vadhan [42].

Families of k-wise almost independent permutations, permute their input in a way that looks random, to any party, which inspects only k values of the output. We give a new method for reducing the number of random bits needed to sample a permutation from such a family. Our method relies on a pseudorandom walk generator, implied by Reingold's log-space algorithm for undirected connectivity [45, 47]. We obtain families of k-wise almost independent permutations, with an optimal number of random bits.

CONTENTS

Chapter 1

Introduction

1.1 Background

1.1.1 Probabilistic computations

In computational complexity we measure computational tasks according to the resources they consume. The classical resources are time and space. The notion of probabilistic algorithms has revolutionized computational complexity. Probabilistic algorithms use random bits, to perform a computation. Using random bits often simplifies the algorithm, and can save time resources.

The downside of probabilistic algorithms, is that they require truly random bits. In theory, this is a resource, similar to time and space. In practice, it is hard to find a distribution of truly random bits. Some physical sources are believed to be ”somewhat random”, but it is often impractical to use them [53].

The sources we often have in reality, are ”weak” random sources, which have various definitions, but in general these distributions have less entropy than a uniform distribution. One simple definition is this: A weak random source on n bits, is a source with entropy k for some $k < n$. An example of such a source, is the set of all n bit strings which have the last $n - k$ bits set to 0. The entropy of this source is k. With respect to this definition, a truly random source is a source on n bits with entropy n. Further examples of weak sources are given in [53].

A great deal of study concerns turning a weak random source into a truly random source. This operation can be formulated as follows. A transformer is a function $T : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$. The first parameter $x \in \{0,1\}^n$ is called the input of T, where x is sampled from a weak random source. The second parameter $y \in \{0,1\}^d$ is called the seed, where y is sampled from a truly random source. The output is a string of m bits. We can place various requirements over the output of T. For instance, if we demand that the output of T is distributed uniformly, then T transform a weak random source into a truly random source. This is explained in Section 1.1.2.

The seed in the above definition is necessary: It can be shown, that a deterministic function which receives a weak source, cannot output even one truly random bit (since it is fixed on at least half of the inputs, it is fixed on a weak source with $k = n - 1$ defined over these inputs).

In this work we studied two problems. One problem concerns a certain type of transformers, called loss-less condensers. We show how to improve current constructions of loss-less condensers. The necessary background and an overview of our result is given in Section 1.1.2.

The other problem concerns families of permutations, with certain pseudorandom properties. We show how to optimize the size of such families. This is further described in Section 1.1.3.

1.1.2 Extractors

We now consider a special type of transformers. If we require the output of the transformer T to be ϵ close (in statistical distance) to a truly random source on m bits, then T transforms weak random sources into (almost) truly random sources, in the following sense: If X is a weak source, then the distribution of $T(X, U)$ is ϵ close to a truly random source on m bits, where U is the uniform distribution. The distance ϵ from uniform, is referred to as the error of T. Such transformers are called extractors, and are formally defined below.

Extractors were first defined by Nisan and Zuckerman [34]. Extractors may also be thought of as pseudorandom generators, that are secured against any (powerful) party. We now define three variants of extractors, that will be used here. For a survey on extractors, refer to [53].

Let $T : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a transformer, and X a distribution with min-entropy at least k.

- If for any X, the output of $T(X, U_d)$ is ϵ close to U_m, then T is a (k, ϵ) extractor.

- If for any X, the size of the image of $T(X, U_d)$ is at least $(1 - \epsilon)2^m$, then T is a (k, ϵ) disperser.

- If for any X, the size of the image of $T(X, U_d)$ is ϵ close to 2^k, then T is a (k, ϵ) condenser.

Note that the difference between these objects is in the restriction over the outputs. The outputs of an extractor are spread evenly over the output bits, while the outputs of a disperser only cover most of the output bits. Finally, the condenser is a $1-1$ function on a small fraction (roughly 2^{k-m}) of the output bits.

Extractors have found many applications, including deterministic amplification, hardness of approximation and Ramsey graphs. For a more detailed description, refer to [12, 53].

There is a "strong" variant to each of the above objects. In this variant, the output of the transformer is augmented with the seed. The transformer is required to maintain the properties of the output, even when the seed is exposed.

We will focus in our result on the strong version for condensers, which is defined as follows. Let $T : \{0,1\}^n \times \{0,1\}^d \rightarrow \{0,1\}^m$ be a transformer. If for any input distribution X with min-entropy h, $h \leq k$, the distribution $U_d \circ T(X, U_d)$ is ϵ close to a distribution $U_d \circ D$, where for all $y \in U_d$, the distribution $D|(U_d = y)$ has at least h min-entropy, then T is a (k, ϵ) loss-less condenser.

Condensers were first defined by Raz and Reingold [41]. Since then, they have been studied and used in various works, including [41, 46, 58, 48, 12, 5, 6]. The (non-explicit) existence of loss-less condensers may be derived using the probabilistic method. A standard calculation shows that for all n, k, m and $\epsilon > 0$, a (k, ϵ) loss-less condenser $C : \{0,1\}^n \times \{0,1\}^d \rightarrow \{0,1\}^m$, with roughly $d = \log(\frac{n-k}{m-k}) + \log(\frac{1}{\epsilon}) + O(1)$ exists (see Section 2.4). This is almost tight given the lower bound of Capalbo et al. ([12], Lemma $B.2$).

In building any of the above variants, we are given the input length n, the min-entropy k and the error ϵ, and our goal is to maximize the output length, and minimize the seed length. The construction should work for any values of k and ϵ.

The reason for maximizing the output length, is that the longer the output is, the more truly random bits we get. This enables us to derandomize algorithms which need more random bits, for instance, or if we are using random bits as a private key in a cryptographic setting, it will increase the security of our scheme.

A shorter seed length is useful, for a few reasons. In a derandomization of an algorithm using a transformer, we iterate over all seeds of the transformer, for each seed calculate the output of the transformer, and use this output as a random input for the algorithm. Thus, a shorter seed means less impact of the derandomization over the running time. Another reason, is when using the transformer to build expander graphs. Here, the length of the seed corresponds to the (logarithm of the) degree of the graph.

Error reduction for transformers

An error reduction for a transformer T with an error ϵ is an explicit algorithm, which takes T as input, and outputs a transformer T', with an error $\epsilon' < \epsilon$. T' might differ from T in its output length and seed length, but not in the input length.

Motivation

There are only few explicit constructions of loss-less condensers. Ta-Shma, Umans and Zuckerman [58] show that any extractor whose correctness proof relies on a reconstructive algorithm (e.g. Trevisan's extractor [61]), and uses a short advice string, yields a loss-less condenser. The condenser is the advice string. In this way, they managed to construct, for every k, and error $\epsilon > 0$, a (k, ϵ) loss-less condenser whose output length is $poly(k/\epsilon)$, with a logarithmic seed length. While for a constant error parameter ϵ, this output length is optimal up to a polynomial factor, for a smaller error, e.g. error of order $1/n$, this is of no use (since a trivial condenser would simply leave the input unchanged).

Our purpose is to construct loss-less condensers with a better dependency of the output length on the error. For every source with entropy k, and error $\epsilon > 0$, we construct a loss-less condenser whose output length is $poly(k) \cdot poly\log(\frac{1}{\epsilon}) + \log n$, and seed length is $O(\log \frac{n}{\epsilon})$ (Corollary 2.3.2).

Our technique and main results

We improve the output length of loss-less condensers, by showing an error reduction, namely,

Theorem 1.1.1 *Suppose there exists an explicit* (k, ϵ_0) *loss-less condenser* $C : \{0,1\}^n \times \{0,1\}^m \rightarrow \{0,1\}^d$ *for some constant* ϵ_0*. Then, for every* $\epsilon > 0$*, there exists an explicit* (k, ϵ) *loss-less condenser*
$\tilde{C} : \{0,1\}^n \times \{0,1\}^{\tilde{d}} \rightarrow \{0,1\}^{\tilde{m}}$ *with*

- $\tilde{d} = d + \log n + O(\log(\frac{1}{\epsilon}))$*, and,*
- $\tilde{m} = m \cdot poly\log(\frac{1}{\epsilon}) + \log n$*.*

Error reduction was done for extractors by Raz, Reingold and Vadhan [42], and for dispersers by Gradwohl et al. [19]. Our reduction follows that of [42], but we simplify it using an idea from the second construction of [19].

The extractor error reduction of Raz, Reingold and Vadhan [42] works by repeating the following two steps. First, they concatenate two applications of the original extractor, using two dependent seeds. This gives a condenser with error parameter ϵ^2, and a source that is close to having entropy rate half. Since their aim is an extractor, they need to keep the output length short, so in the second step they apply a high min-entropy extractor on the output. These two steps are now repeated recursively, each time squaring the error.

We are interested in building condensers, and so we modify the above approach as follows. One way to go is repeating the derandomized squaring procedure, without the intermediate compression steps. However, a simpler way to go is to have only a single step, where the original condenser is run many times on dependent seeds that are chosen in a derandomized way. We do that by selecting a random input z for a disperser, and taking as our set of seeds the neighbors of z in the disperser. I.e., if $\Gamma(z)$ is the set of neighbors of z, then the new condenser basically outputs the concatenation of $C(x, y)$ for all $y \in \Gamma_g(z)$ (for the exact construction see Section 2.2.1). A similar seed derandomization was used in the disperser error reduction of Gradwohl et al. [19].

1.1.3 Families of k-wise functions

A family G of functions is k-wise independent, if a function g chosen at random from G is completely indistinguishable from a function f chosen at random from the set of all functions, for any process that receives the value of either f or g at any k points. We can relax the requirement and talk about almost k-wise independence by requiring that the advantage of a distinguisher be limited by some δ.

Families of functions that are k-wise independent (or almost independent) were constructed and applied extensively in the computer science literature (see [3, 31]). There is a rather natural construction that is optimal in terms of size: Let G consist of all polynomials of degree $k-1$ over $GF[2^n]$. Then the description of each $f \in F$ is kn-bit long. It is easy to see that this is the minimum number of bits needed.

We are interested in constructing a permutation, i.e. a 1-1 function $g : \{0,1\}^n \rightarrow \{0,1\}^n$, which is indistinguishable from a random permutation for a process that examines at most k points (a variant also allows examining the inverse). In other words, we are interested in families of permutations such that restricted to k inputs their output is identical (or statistically close, up to distance δ), to that of a random permutation. For $k = 2$ the set of linear permutations ($ax + b$ where $a \neq 0$) over $GF[2^n]$ constitutes such a family. For $k > 3$ no explicit (non-trivial) construction is known for k-wise exactly independent permutations.

Once we settle on k-wise almost independent permutations, with error parameter δ, then we can hope for permutations with description length $O(kn + \log(\frac{1}{\delta}))$ [1]; this is what a random (non-explicit) construction gives (see Section 3.2.2). There are a number of proposals in the literature of constructing k-wise almost independent permutations (see Section 3.3), but the description length they obtain is in general significantly higher than this asymptotically optimal value. This paper obtains the first construction of k-wise almost independent permutations, with description length $O(kn + \log(\frac{1}{\delta}))$, for every value of k.

[1] The lower bound of kn trivially follows as in the case of functions (simply since the output of a random permutation on k fixed inputs has entropy close to kn).

Motivation

Given the simplicity of the question, and given how fundamental k-wise independent functions are, we feel that it is well motivated in its own right. Indeed, k-wise independent permutations have been receiving a growing amount of attention with various motivations and applications in mind (e.g. [20, 32]).

We give a method for "derandomizing" essentially all previous constructions of k-wise almost independent permutations. It is most effective, and easiest to describe for permutation families obtained by composition of simpler permutations. As most previous constructions fall into this category, this is a rather general method. In particular, based on any one of a few previous constructions, we obtain k-wise almost independent permutations with optimal description length, up to a constant factor.

Our technique and main results

Consider a family of permutations $\mathcal{F}$, with a rather small description length s. We denote by $\mathcal{F}^t$ the family of permutations obtained by composing any t permutations $f_1, f_2, \ldots, f_t$ in $\mathcal{F}$. Now assume that $\mathcal{F}^t$ is a family of k-wise almost independent permutations. The description length of $\mathcal{F}^t$ is $t \cdot s$ as we need to describe t independent permutations from $\mathcal{F}$. We will argue that such constructions can be derandomized in the sense that it is sufficient to consider a subset of the t-tuples of $\mathcal{F}$ functions. This will naturally reduce the overall description length.

A permutations family is closely associated with a certain type of graph, which we call the companion graph. This graph has a vertex for each k-tuple of distinct n-bit strings. There is an edge between vertices $u = (x_1, x_2, \ldots x_k)$ and $v = (y_1, y_2, \ldots y_k)$, labelled with a permutation f from $\mathcal{F}$, if $f(x_i) = y_i$ for $i = 1, 2, \ldots, k$. Evaluating a random permutation from $\mathcal{F}$ on inputs $x_1, x_2, \ldots x_k$ is equivalent to a step on the graph, starting from the vertex corresponding to $x_1, x_2, \ldots x_k$, and picking a random edge. This way, evaluating t random permutations from $\mathcal{F}$ is equivalent to taking a random walk of length t on the graph.

We use pseudorandom generators for walks on graphs. Such a generator outputs a sequence of labels. This sequence, when interpreted as a walk on a graph, has the following property: For any 'consistently labelled' undirected graph, and any start vertex, the distribution over the end vertices obtained by taking the walk, is statistically close to uniform. Such generators with sufficiently good parameters are implied by the proof that undirected connectivity is in logspace of Reingold [45], and made explicit by Reingold, Trevisan and Vadhan [47].

The generator will be used, to generate a sequence of permutations from $\mathcal{F}$ (the labels will be interpreted as permutations). We will compose these permutations, instead of choosing truly random permutations from $\mathcal{F}$.

By our assumption on $\mathcal{F}^t$, a random walk of length t on the companion graph of $\mathcal{F}$ lands on a uniformly distributed vertex. This means that the graph has some 'expansion properties', and we say that it is a good expander (essentially, it is highly connected). The quality of the graph effects the parameters of the generator. For good expanders, the generator will produce a walk of length t' which is slightly longer than t, but will use less than $t \cdot s$ random bits to generate it. Thus we manage to use less random bits for the walk. The description length will be the number of random bits used by the generator.

Now consider g' which is the composition of t' permutations $f'_1, f'_2, \ldots, f'_{t'}$ in $\mathcal{F}$, selected using the pseudorandom walk generator. Assume that the distribution on g' is not k-wise almost independent. This means that there are k evaluation points $x_1, x_2, \ldots x_k$ such that the distribution $g'(x_1), g'(x_2), \ldots g'(x_k)$ is not close enough to uniform. That is, there exists a test $\mathcal{T}$ that distinguishes $g'(x_1), g'(x_2), \ldots g'(x_k)$ from uniform. This test distinguishes between a random walk and a walk generated by the pseudorandom walk generator, thus violating the security of the generator.

Related work

There are several lines of constructions that are of particular relevance to our work. We describe them in more detail in Section 3.3. The information is summarized in Table 1.1.

Table 1.1: Summary of Results and Previous Work on k-wise δ-dependent Permutations.

Family	Description Length	Range of Queries
Feistel[4] (Luby Rackoff)	$nk + O(n)$ $O(nk \cdot \log \frac{\delta}{\delta_0})$	$k < 2^{\frac{n}{4}-O(1)}$, $\delta = \frac{k^2}{2^{n/2}}$ $k < 2^{\frac{n}{4}-O(1)}$,any δ, $\delta_0 = \frac{k^2}{2^{n/2}}$
Simple 3-Bit Permutations [10, 18, 20]	$O(n^2k(nk + lg(\frac{1}{\delta}))\lg(n))$	$k \leq 2^n - 2$
Thorp Shuffle [30, 32, 50]	$O(n^{45}k\log(\frac{1}{\delta}))$	$k \leq 2^n$
Non-Explicit Construction (Thm. 3.2.1)	$O(nk + \log(\frac{1}{\delta}))$	$k \leq 2^n$
This Work (Theorem 3.4.5)	$O(nk + \log(\frac{1}{\delta}))$	$k \leq 2^n$

1.2 Context of this work

This work can be viewed as a particular derandomization of random objects. The original use of the term derandomization, is in the context of probabilistic algorithms. To derandomize an algorithm, is to execute it with less random bits (preferably none). Tools for derandomization of space bounded algorithms include the pseudorandom generators of Nisan [33], and of Nisan and Zuckerman [34].

A stronger tool is the extractor, which allows a derandomization of an arbitrary class of algorithms, as it is secured against any strong distinguisher (this is the advantage of a statistical closeness between distributions). It has found many usages, for example a derandomization of BPP [14, 63].

The notion of extractors has developed and found many applications. For some applications, weaker notions were defined (e.g. disperser, which is sufficient for derandomization).

We deal with loss-less condensers and permutations. Loss-less condensers are a particular instance of extractors. Permutations are entropy preserving functions.

In both cases, we use derandomization to reduce the resources of the objects. For loss-less condensers, we reduce the output length. For permutation families, we reduce the description length. As in a derandomization of an arbitrary algorithm, the tradeoff is in execution time.

[4] The first row is based on 4 rounds with the first and last being pair-wise independent [32]. Analysis of related constructions [28, 36, 37] approaches $k = 2^{n/2}$, but does not go beyond. It is possible to obtain any $\delta' \leq \delta$ by the composition of independent permutations (which adds a $\log \frac{\delta}{\delta'}$ multiplicative factor.)

Indeed, the running time of our objects increases (in the case of permutation families, this impact is more evident, refer to Section 3.5).

1.3 Thesis organization

The rest of this work is organized as follows. In Chapter 2 we show an error reduction technique for loss-less condensers. In Chapter 3 we describe how to reduce the size of families of permutations.

Chapter 2

Error Reduction for Loss-Less Condensers

2.1 Preliminaries

A probability distribution D on a finite set Ω is a function $D : \Omega \rightarrow [0,1]$ such that $\sum_{x \in \Omega} D(x) = 1$. U_n is the uniform distribution on $\{0,1\}^n$. We measure distance between distributions by the statistical variation distance.

Definition 1 *(statistical distance) Let D_1, D_2 be distributions over a finite set Ω. The variation distance between D_1 and D_2 is*

$$\|D_1 - D_2\| = \frac{1}{2} \sum_{\omega \in \Omega} |D_1(\omega) - D_2(\omega)| = \max_{S \subseteq \Omega} |D_1(S) - D_2(S)|.$$

We say that D_1 and D_2 are ϵ close if $\|D_1 - D_2\| \leq \epsilon$.

Note that if two distributions are ϵ close then there is no distinguisher (not even an inefficient one) that can distinguish the distributions with advantage better than ϵ.

The support of a distribution D is the set of all x's for which $D(x) \neq 0$. A distribution D is flat over its support $A \subseteq \Omega$ if $D(a) = \frac{1}{|A|}$ for all $a \in A$.

Suppose D is a distribution over $A \times B$. We denote $D = D_1 \circ D_2$, where D_1 is the distribution D induces on A and D_2 the distribution induced on B. We denote $(D_2 | D_1 = a)$ the distribution D induces on B given that $D_1 = a$.

We now define block-wise sources, lossy condensers and loss-less condensers:

Definition 2 *(block-wise source) A distribution $B = B_1 \circ B_2$ is a $((n_1, k_1), (n_2, k_2))$ block source, if B_i is a distribution over $\{0,1\}^{n_i}$, $H_\infty(B_1) \geq k_1$, and for all prefixes b_1, $H_\infty(B_2 | B_1 = b_1) \geq k_2$.*

Definition 3 *(condenser and entropy loss) Let*
$C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *be a function. We say that* C *is a* (k, ϵ) *condenser with entropy loss* Δ*, if for any flat distribution* X *over* $\{0,1\}^n$ *such that* $H_\infty(X) = h \le k$ *we have that* $U_d \circ C(X, U_d)$ *is* ϵ *close to a* $((d, d), (m, h - \Delta))$ *block source. If* $\Delta = 0$ *we say* C *is a loss-less condenser.*

We cite a simple concatenation lemma that can be found, e.g., in [57] (Lemma 1).

Lemma 2.1.1 (concatenation of condensers) *Let*
$C_1 : \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_1}$ *be a* (k, ϵ_1) *condenser, with entropy loss* Δ_1*. Let* $C_2 : \{0,1\}^n \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}$ *be a* (Δ_1, ϵ_2) *condenser, with entropy loss* Δ_2*. Define* $C_1 \circ C_2 : \{0,1\}^n \times \{0,1\}^{d_1+d_2} \to \{0,1\}^{m_1+m_2}$ *by*

$$C_1 \circ C_2(x; r_1, r_2) \quad = \quad C_1(x; r_1) \circ C_2(x; r_2).$$

Then, $C_1 \circ C_2$ *is a* $(k, \epsilon_1 + \epsilon_2)$ *condenser, with entropy loss* Δ_2*.*

We also need a loss-less condenser for low min-entropies. Such condensers were used in many previous works (e.g., [41]) and can be obtained using, for example, error correcting codes. We use Reed-Solomon codes. We interpret $x \in \{0,1\}^n$ as a degree p polynomial $\hat{x} : \mathbb{F}_q \to \mathbb{F}_q$ (and so we need $p \log(q) \ge n$ so that two different inputs are interpreted as two different polynomials). We have:

Lemma 2.1.2 *For every* $k \le n$*,* $\epsilon > 0$*, let* $q = \frac{n 2^{2k}}{\epsilon}$ *and* $p \ge \frac{n}{\log q}$*. Define* $C_{RS}(x; \alpha) = \hat{x}(\alpha)$ *for* $\alpha \in \mathbb{F}_q$*. Then,* $C_{RS} : \{0,1\}^n \times \{0,1\}^{d_{RS}} \to \{0,1\}^{d_{RS}}$ *is a* (k, ϵ) *loss-less condenser and* $d_{RS} = \log(n) + 2k + \log(\frac{1}{\epsilon})$*.*

Proof: Fix any set $X \subseteq \{0,1\}^n$ of cardinality 2^k. Any two different inputs are interpreted as two different low-degree polynomials, and the probability they collide at α is at most $\frac{p}{q}$. Thus, the probability that for α there is any collision between two elements in X is at most $2^{2k} \cdot \frac{p}{q} \leq \epsilon$, and C_{RS} is a loss-less condenser. ■

We also need dispersers:

Definition 4 *(disperser)* $D : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a* (k, ϵ) *disperser, if for any distribution* X *over* $\{0,1\}^n$ *with* $H_\infty(X) \geq k$*, the support of* $D(X, U_d)$ *is of size at least* $(1-\epsilon)2^m$*.*

2.2 Error reduction

2.2.1 The construction

Consider a loss-less condenser C with a constant error ϵ_0. Our goal is to construct a new loss-less condenser with arbitrarily small error. We use two components in our construction: a disperser and the low min-entropy loss-less condenser C_{RS} of Lemma 2.1.2.

Our construction follows in two steps. Assume that we have an input x. First, we use a disperser D to select a set of seeds for the condenser. This will be performed by selecting a random input z for the disperser, then taking as our set of seeds the neighbors of z in D, denoted by $\Gamma(z)$. For each neighbor $y \in \Gamma(z)$, we view y as a seed of C, and output $C(x, y)$. The concatenation of $C(x, y)$ for all $y \in \Gamma_g(z)$ composes the first part of the output.

Construction of function C_D:

Input : A (k, ϵ_0) loss-less condenser $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ and an error parameter $\epsilon < \epsilon_0$.

Auxiliary function : $D : \{0,1\}^{d+c\log(\frac{1}{\epsilon})} \times \{0,1\}^{d'} \to \{0,1\}^d$ a $(d, 1-2\epsilon_0)$ disperser, for some constant $c \geq 1$ that will be picked later.

Output : We define:

$$C_D(x;z) \quad = \quad C(x,y_1) \circ C(x,y_2) \circ \ldots \circ C(x,y_l)$$

where $\Gamma(z) = \{y_1, \ldots, y_l\}$.

We claim that:

Theorem 2.2.1 *Let $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a (k, ϵ_0) loss-less condenser. Then for any $\epsilon < \epsilon_0$, $C_D : \{0,1\}^n \times \{0,1\}^{d+c\log(\frac{1}{\epsilon})} \to \{0,1\}^{m'=m\cdot 2^{d'}}$ is a $(k, 2\epsilon)$ condenser with entropy loss* $\Delta_1 = \log(\frac{1}{\epsilon}) + 1$.

Having that, we augment C_D with the output of the low min-entropy loss-less condenser C_{RS} which we apply on x and on a new random seed. I.e.,

$$C_{D,C_{RS}}(x;z,\alpha) \quad = \quad C_D(x;z) \circ C_{RS}(x;\alpha).$$

By Lemma 2.1.1 (the concatenation lemma) we see that $C_{D,C_{RS}}$ is a $(k, 3\epsilon)$ loss-less condenser.

2.2.2 High-level proof

Let us see why this construction works. Fix some flat distribution X with min-entropy h. We first claim that because C is a constant-error loss-less condenser, for most of its seeds y, most of the images are obtained with roughly about the same probability $\frac{1}{|X|} = 2^{-h}$.

Definition 5 *We say a seed-image pair $(y, b) \in \{0,1\}^d \times \{0,1\}^m$ is t-heavy, if* $\Pr[C(X, y) = b] \geq (2^t + 1) \cdot 2^{-h}$*, and we say it is good otherwise.*

Let B denote the set of all seed-image pairs that are $t = \log(\frac{1}{\epsilon})$ heavy. By counting, and because C is a constant-error loss-less condenser, we see that $|B| \leq 2^{d+h}\epsilon\epsilon_0$ (for the proof see Claim 2.2.3 in Section 2.2.3).

We say that a seed y is *bad* for x, if $(y, C(x, y)) \in B$. For an input x, let B_x be the set of seeds bad for it. We say x is bad if more than $2\epsilon_0$ fraction of the seeds are bad for it, i.e., if $|B_x| \geq 2\epsilon_0 2^d$. A main observation (already in [42]) is that only an ϵ fraction of the x's are bad, and the rest are good. The intuitive reason for that is that C is a good constant-error condenser for B_x. For the proof see Claim 2.2.4 in Section 2.2.3. From now on we ignore the bad x's.

We are left with good x's, where at most $2\epsilon_0$ fraction of the seeds are bad for x. What we do now, is amplifying the success probability by trying many seeds over the same x, and choosing these seeds in a derandomized way using the disperser D. We say that z is bad for x, if $\Gamma(z) \subseteq B_x$, i.e., all the seeds $\{y_1, \ldots, y_\ell\}$ chosen, are bad for x. As D is a good disperser, it easily follows that for every good x we have that $Pr_z[z \text{ is bad for } x] \leq \epsilon$. For the proof see Claim 2.2.5 in Section 2.2.3. From now on we ignore bad z's, and concentrate on the good ones. We claim:

Claim 2.2.2 *For every good x, and z which is good for x, we have* $\Pr_X[C_D(X, z) = C_D(x, z)] \leq 2^{-(h - \log(\frac{1}{\epsilon}) - 1)}$.

Proof: Fix a good x and z that is good for x. Then, there exists a seed $y_i \in \Gamma(z)$ that is good for x, and so, in particular,

$\Pr_X[C_D(X,z) = C_D(x,z)] \leq \Pr_X[C(X,y_i) = C(x,y_i)] \leq 2^{-(h-\log(\frac{1}{\epsilon})-1)}$, where the last inequality is because y_i is good for x. ■

Altogether, we see that $U_d \circ C(X,U_d)$ is 2ϵ close to a distribution with $h - \log(\frac{1}{\epsilon}) - 1$ min-entropy, as desired.

2.2.3 Low-level lemmas

Claim 2.2.3 $|B| < 2^{d+h} \cdot \epsilon \cdot \epsilon_0$.

Proof: Let B_t be the set of all seed-image pairs that are t-heavy. On the one hand, $\Pr[U_d \circ C(X,U_d) \in B_t] \geq |B_t| 2^{-(d+h)}(2^t+1)$ because each element in B_t is heavy. On the other hand, $U_d \circ C(X,U_d)$ is ϵ_0 close to a distribution C' where $\Pr[C' \in B_t] = |B_t|2^{-(d+h)}$ (because C is ϵ_0 close to a loss-less condenser). Together, we see that $|B_t| \leq 2^{d+h-t}\epsilon_0$. Plugging $t = \log(\frac{1}{\epsilon})$ gives the desired result. ■

Claim 2.2.4 *$Pr_{x \in X}[x$ is bad$] < \epsilon$.*

Proof:

Let X' be the flat distribution over all bad x's, and assume $|X'| \geq \epsilon |X|$ and so $H_\infty(X') \geq h - \log(\frac{1}{\epsilon})$. Clearly, $H_\infty(X') \leq H_\infty(X) = h \leq k$. It follows that $U_d \circ C(X',U_d)$ is ϵ_0 close to a $((d,d),(m,h-\log(\frac{1}{\epsilon})))$ block source C'.

Now, on the one hand, $\Pr[U_d \circ C(X',U_d) \in B]$ is at least $2\epsilon_0$ (because for every bad x, with probability $2\epsilon_0$ a random seed y is bad for it, and $(y, C(x,y)) \in B$). On the other hand for C', $\Pr[C' \in B] \leq |B|2^{-(d+h-\log(\frac{1}{\epsilon}))}$ and so $\Pr[U_d \circ C(X',U_d) \in B] \leq \epsilon_0 + |B|2^{-(d+h-\log(\frac{1}{\epsilon}))}$. Rearranging, we see that $|B| \geq 2^{d+h}\epsilon\epsilon_0$ contradicting Claim 2.2.3. ■

Claim 2.2.5 *Let x be a good input. Then $Pr_z[\Gamma(z) \subseteq B_x] \leq \epsilon$.*

Proof: Fix any good input x. Let Z be the set of all bad z's for x. By the disperser definition it then follows that $|Z| \leq 2^d$, and hence the probability such a bad z is picked is at most $2^d/2^{d+c\log(\frac{1}{\epsilon})} \leq \epsilon$ $(c \geq 1)$. ■

2.2.4 Plugging parameters

We now choose the disperser D we plug into the construction of $C_{D,C_{RS}}$. The simplest choice is the disperser of Ta-Shma, Umans and Zuckerman [58] which has $O(\log n)$ seed length, $O(\log n)$ entropy loss and a constant error. Thus, we may take $c = 1$ in the construction of Section 2.2.1 and $d' = O(\log(d + \log(\frac{1}{\epsilon})))$ and this gives in Theorem 1.1.1 $\tilde{d} = d + \log n + O(\log(\frac{1}{\epsilon}))$ and $\tilde{m} = m \cdot poly(d + \log(\frac{1}{\epsilon})) + \log n$.

For a large $\epsilon \geq 2^{-d}$, we can improve on that by using the extractors of Capalbo et al. [12], which have seed length that is logarithmic in the entropy deficiency rather than the input length. Formally, Capalbo et. al prove:

Theorem 2.2.6 *[12] Let $\alpha > 0$ be an arbitrarily small constant. For all $0 \leq \Delta \leq (1-\alpha)n$, and $\gamma > 0$ constant, there exists a $(n - \Delta, \gamma)$ extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, with $d = O(\log \Delta)$, and $m \geq n - (1+\alpha)\Delta$.*

Using $c = 2$ in the construction of Section 2.2.1 we get the result stated in Theorem 1.1.1.

2.3 Applications

We now demonstrate that our error reduction technique can be applied to the loss-less condensers of Ta-Shma, Umans and Zuckerman [58].

Theorem 2.3.1 *[58] For all $\epsilon \in (0, \frac{1}{2})$ constant, and all $k \leq n$, there exists an explicit (k, ϵ) loss-less condenser $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, for the following choices of d and m:*

- *$d = poly \log(n)$, $m = O(k)$.*
- *For all $\alpha > 0$ constant, $d = O(\log n)$, $m = O(k^{1+\alpha})$.*

Where the implicit constants depend on ϵ.

Applying Theorem 1.1.1 we establish:

Corollary 2.3.2 *For all $\epsilon > 0$ and $k \leq n$, there exists an explicit (k, ϵ) loss-less condenser $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, for the following choices of d and m:*

- $d = poly\log(n) + O(\log(\frac{1}{\epsilon}))$, $m = O(k) \cdot poly\log(\frac{1}{\epsilon}) + \log n$.
- *For all* $\alpha > 0$ *constant,* $d = O(\log \frac{n}{\epsilon})$, $m = O(k^{1+\alpha}) \cdot poly\log(\frac{1}{\epsilon}) + \log n$.

2.4 An upper bound

Capalbo et al. [12] showed (by a reduction to an extractors lower bound) the following lower bound for loss-less condensers:

Theorem 2.4.1 *[12] Let k, m, n, ϵ be such that $k + O(1) < m < n$ and $\epsilon \leq \frac{1}{4}$. If $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a (k, ϵ) loss-less condenser, then*
$d \geq \max\{\log(\frac{n-k}{m-k}), \log(\frac{1}{\epsilon})\} + O(1)$.

We augment this with a matching non-explicit upper bound.

Lemma 2.4.2 *For every $k \leq m \leq n$ and $\epsilon > 0$, there exists a (k, ϵ) loss-less condenser $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with*
$d \leq \log(\frac{n-k+\log(\frac{1}{\epsilon})+O(1)}{m-k+\epsilon}) + \log(\frac{1}{\epsilon})$.

Proof: Ta-Shma, Umans and Zuckerman [58] showed that a (k, ϵ) loss-less condenser $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is equivalent to a
$([N = 2^n], [MD = 2^m 2^d])$ bipartite graph, which is a $(K = 2^k, (1-\epsilon)2^d)$ expander. Define a graph $G = (V_1 = [N], V_2 = [DM], E)$ with left degree D, by choosing for each $v \in N$, D random neighbors $\{m_1, \ldots, m_D\}$ from $[M]$, and connecting v to the vertices $(1, m_1), \ldots, (D, m_D) \in [DM]$. By the discussion above, it is sufficient to prove that with a positive probability, G is a $(K, (1-\epsilon)D)$ bipartite expander.

Let $X \subseteq N$ be a set of cardinality $H = 2^h$ for some $1 \leq h \leq k$. Observe that

$$\Pr[|\Gamma(X)| < (1-\epsilon)DH] \ \leq \ \binom{DH}{\epsilon DH} \cdot \left(\frac{(1-\epsilon)H}{M}\right)^{\epsilon DH}$$

because if $|\Gamma(X)| < (1-\epsilon)DH$ then there exists a set $X' \subseteq X$ of cardinality $\epsilon|X|$ such that $\Gamma(X') \subseteq \Gamma(X \setminus X')$.

By a union bound, over all such subsets X of size 2^h,

$$\binom{N}{H}\binom{DH}{\epsilon DH}\left(\frac{(1-\epsilon)H}{M}\right)^{\epsilon DH} \ \leq \ \left[\frac{eN}{H}\frac{e}{\epsilon}\left(\frac{(1-\epsilon)H}{M}\right)^{\epsilon D}\right]^{H}$$

Choosing $D = \frac{1}{\epsilon}\frac{n-h+\log(\frac{1}{\epsilon})+O(1)}{m-h+\epsilon}$ this quantity is at most 2^{-H} (we have used $1-\epsilon \leq e^{-\epsilon}$). Notice that if D is good for h (i.e., $D \geq \frac{1}{\epsilon}\frac{n-h+\log(\frac{1}{\epsilon})+O(1)}{m-h+\epsilon}$) then D is also good for $h-1$ because $\frac{a}{b} \geq \frac{a+1}{b+1}$ for $a \geq b$. Thus, summing over all possible h and subsets H, we get a geometric sum whose total is less than 1, hence a good graph exists. ■

2.5 Further work

The dependency of the seed length in our construction on the input length and error is $O(\log\frac{n}{\epsilon})$, while the correct dependency is roughly $d = \log(\frac{n-k}{m-k}) + \log(\frac{1}{\epsilon}) + O(1)$. For linear min-entropy, this difference is notable: By our non-constructive argument, there are loss-less condensers with a degree which depends only on the error - $O(\log(\frac{1}{\epsilon}))$ (i.e. take $m = n/2$, $k = n/4$). For a constant error, this is constant. However, our construction (as well as all other constructions known to us) fails to achieve a constant degree. An interesting question is what causes this barrier, and of course, how it can be broken.

Chapter 3

Derandomized Constructions of k-Wise (Almost) Independent Permutations

3.1 Preliminaries

We denote by P_n the set of all permutations over $\{0,1\}^n$. For any $f, g \in P_n$ denote by $f \circ g$ their composition (i.e., $f \circ g(x) = f(g(x))$). For $x, y \in \{0,1\}^n$, $x \oplus y$ denotes their bit-by-bit exclusive-or. Denote by $[N_k]$ the set of all k-tuples of distinct n-bit strings.

3.1.1 Random walks

A random walk on a graph starting at a vertex v is a sequence of vertices, $u_0, u_1, \ldots$ where $u_0 = v$ and for $i > 0$ the vertex u_i is obtained by selecting an edge (u_{i-1}, u_i), uniformly from the edges leaving u_{i-1}. Regular, undirected graphs, with self-loops, have the property that a random walk on the graph (starting at an arbitrary vertex) converges to the uniform distribution on the vertices. The rate of convergence is governed by the second largest (in absolute value) eigenvalue of the graph. Below we formalize these notions.

Definition 6 *(Spectral Gap) Let $G = (V, E)$ be a connected, d-regular undirected graph on n vertices. The normalized adjacency matrix of G is its adjacency matrix divided by d. Denote this matrix by $M \in M_n(\mathbb{R})$. Denote by $1 = \lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ its eigenvalues. We denote by $\lambda(G)$ the second eigenvalue in absolute value. Namely, $\lambda(G) \doteq \max\{|\lambda_2|, |\lambda_n|\}$. The spectral gap of G, is defined by gap$(G) \doteq 1 - \lambda(G)$.*

Definition 7 *(Mixing Time) Let $G = (V, E)$ be a connected, regular, undirected graph with self-loops, on n vertices. Let $M \in M_n(\mathbb{R})$ be the normalized adjacency matrix of G. A random walk on this graph is an ergodic Markov chain, whose transition matrix is M. Its stationary distribution π is the uniform distribution on the vertices. For $x \in V$, define the mixing time of the walk starting* *from x, by $\tau_x(\epsilon) = \min\{n | \|M^n 1_x - \pi\| \leq \epsilon\}$, where 1_x is the distribution concentrated on x. The mixing time of the walk is defined by $\tau(\epsilon) = \max_{x \in V} \tau_x(\epsilon)$.*

We have the following theorems, relating the mixing time of a walk with the spectral gap of the graph.

Theorem 3.1.1 *[55] Let $G = (V, E)$, M, π be as in Definition 7. Let $\epsilon > 0$. Let λ be the second largest eigenvalue of G. Then*

$$\frac{1}{2}\frac{\lambda}{1-\lambda}\ln(\frac{1}{2\epsilon}) \leq \tau(\epsilon) \leq \frac{1}{1-\lambda}\ln(\frac{|V|}{\epsilon}).$$

Usually, such a claim is used to bound the mixing time. However, we will be using constructions with a proven mixing time. The construction itself may also provide a bound on the spectral gap. In case it does not, we will be able to use Theorem 3.1.1, to bound the gap of the graph from below. A simple calculation using Theorem 3.1.1, shows that

$$gap(G) = \Omega(\frac{\ln(\frac{1}{2\epsilon})}{\tau(\epsilon)}).$$

The following theorem will be useful for us. It shows, that the distance of a distribution induced by a random walk, from its stationary distribution, is a sub-multiplicative function of the time. We will use this result, to obtain a composition theorem for families of permutations. Namely, if selecting one permutation from a family of permutations, induces a distribution, which is δ-close to uniform, then composing two such permutations, yields a distribution which is roughly δ^2-close to uniform.

Theorem 3.1.2 *([2] Chapter 2, Lemma 20) Let $G = (V, E)$, M, π be as in Definition 7. Define $d(t) = \max_{x \in V} \|M^t 1_x - \pi\|$. Then for all $s, t \geq 0$, $d(s+t) \leq 2d(s)d(t)$.*

3.2 The Existence of k-Wise δ-Dependent Permutations

In this section we define k-wise δ-dependent permutations, discuss their existence, and show that the distance parameter δ is reduced by the composition of such permutations.

3.2.1 Definitions

Definition 8 *Let $n, k \in \mathbb{N}$, and let $\mathcal{F} \subseteq P_n$ be a family of permutations. Let $\delta \geq 0$. The family $\mathcal{F}$ is k-wise δ-dependent if for every k-tuple of distinct elements $(x_1, \ldots, x_k) \in [N_k]$, the distribution $(f(x_1), f(x_2), \ldots, f(x_k))$, for $f \in \mathcal{F}$ chosen uniformly at random is δ-close to $U_{[N_k]}$. We refer to a k-wise 0-dependent family of permutations as k-wise independent.*

We are mostly interested in explicit families of permutations, meaning that both sampling uniformly at random from $\mathcal{F}$ and evaluating permutations from $\mathcal{F}$ can be done in polynomial time. The parameters we will be interested in analyzing are the following:

Description Length The description length of a family $\mathcal{F}$ is the number of random bits, used by the algorithm for sampling permutations uniformly at random from $\mathcal{F}$. Alternatively, we may consider the size of $\mathcal{F}$, which is the number of permutations in $\mathcal{F}$, denoted $|\mathcal{F}|$. In all of our applications, the description length of a family $\mathcal{F}$ equals $O(\log(|\mathcal{F}|))$. By allowing $\mathcal{F}$ to be a multi-set we can assume without loss of generality that the description length is exactly $\log(|\mathcal{F}|)$.

Time Complexity The time complexity of a family $\mathcal{F}$ is the running time of the algorithm for evaluating permutations from $\mathcal{F}$.

Our main goal would be to reduce the description length of constructions of k-wise δ-dependent permutations. Still, we would take care to keep time complexity as efficient as possible. See additional discussion in Section 3.5.

3.2.2 Non-explicit construction

We note the following non-explicit families of permutations. Our goal would be to obtain families of size which is as close as possible to that obtained by the non-explicit argument below.

The following theorem follows by the approximation method of Azar, Motwani and Naor [4]. They provide a general way, to approximate an arbitrary distribution, over a finite abelian group.

More specifically, Azar, Motwani and Naor prove ([4] Theorems 3.2, 3.5) that for any probability distribution F defined over a finite abelian group G, and $\epsilon > 0$, there exists a probability distribution F' over some sample space Ω, of support size $O(\frac{\lg |G|}{\epsilon^2})$, such that $\|F' - F\| \leq |G|\frac{\epsilon}{2}$.

Here, the group is the set of n bit strings, and we set $\epsilon = \frac{2\delta}{2^{nk}}$. We have:

Theorem 3.2.1 (Existence of k-wise δ-dependent Distribution) *Let $n \in \mathbb{N}$. For all $1 \leq k \leq 2^n$ and $\delta > 0$, there exists a distribution, that is δ-close to a family $\mathcal{F} \subseteq P_n$ of k-wise permutations, of support size $O(\frac{2^{2nk}nk}{\delta^2})$.*

3.2.3 Composition of permutations

Some of the permutations families we will inspect, require several compositions, to get a distribution close to uniform. In fact, as we argue below, composing permutations is an effective method for reducing the distance parameter δ. This motivates the following definition.

Definition 9 *Let $\mathcal{F} \subseteq P_n$. The t'th power of $\mathcal{F}$, denoted by $\mathcal{F}^t \subseteq P_n$, is $\{ f_1 \circ \ldots \circ f_t \mid f_1, \ldots, f_t \in \mathcal{F} \}$.*

Remark 1 *Let $\mathcal{F} \subseteq P_n$. Observe that $|\mathcal{F}^t| = |\mathcal{F}|^t$, and that the time complexity of $\mathcal{F}^t$ is essentially t times the time complexity of $\mathcal{F}$.*

As we will see, starting with a family $\mathcal{F}$ which is δ-dependent results in $\mathcal{F}^t$ which is only $O(\delta)^t$-dependent. Increasing the description length and time complexity linearly, pays off in an exponential decay of the error.

We now state our composition theorem.

Theorem 3.2.2 *Let $\mathcal{F}$ be a k-wise δ-dependent family. Then, $\mathcal{F}^2$ is a k-wise $2\delta^2$-dependent family.*

The proof of Theorem 3.2.2 uses the companion graph of the permutations family $\mathcal{F}$, described in Subsection 1.1.3 (for the exact definition, refer to Subsection 3.4.1). Observe, that a step on the companion graph is equivalent to computing a permutation from $\mathcal{F}$.

Proof: Let $\mathcal{F}$ be a k-wise δ-dependent family. This means, that after taking one random step on its companion graph, the distance from a uniform distribution is δ. Let $d(t)$ be as in Theorem 3.1.2. Then $d(1) = \delta$, and since by Theorem 3.1.2, $d(2) \leq 2d(1)^2 = 2\delta^2$, we conclude that $\mathcal{F}^2$ is a k-wise $2\delta^2$-dependent family. ■

Theorem 3.2.2 has the following corollary, which follows by a simple induction.

Corollary 3.2.3 *Let $\mathcal{F}$ be a k-wise δ-dependent family. Then, for any $\ell \in \mathbb{N}$, $\mathcal{F}^\ell$ is a k-wise $(\frac{1}{2}(2\delta)^\ell)$-dependent family.*

3.3 Short Survey of Explicit Constructions

As mentioned in the introduction, for $k = 2$ the set of linear permutations is a good construction (see also [32]).There are no known k-wise exactly independent permutations, whether algebraic or not. The rest of our discussion will therefore focus on k-wise almost independent permutations. We now survey some known constructions yielding k-wise almost independent permutations with reasonable parameters.

3.3.1 Feistel based constructions

In their famed work Luby and Rackoff [26] showed how to construct pseudorandom permutations from pseudorandom functions. The construction is based on the Feistel Permutation: For any function $f \in \{0,1\}^{n/2} \to \{0,1\}^{n/2}$ the Feistel Permutation is defined by $(L, R) \to (R, L \oplus f(R))$, where $|L| = |R| = n/2$. The construction uses a composition of several such permutations.

Naor and Reingold [32] construct a family of k-wise δ-dependent permutations, where the description of each permutation is $kn + O(n)$ bits with $\delta = k^2/2^{n/2}$ (note that the size is optimal up to the additive $O(n)$ term). The analysis is useless when k is larger than $2^{n/4}$.

There are Feistel constructions of k-wise δ-dependent permutations, for k up to $2^{n/2}$ (see Naor and Reingold [32], Patarin [35, 36, 37], and Maurer and Pietrzak [27]).

Feistel permutations approach yields succinct k-wise δ-dependent permutation as long as k is not too large and δ is not too small, and is probably the method of choice for this range. To reduce the dependency δ one can use Theorem 3.2.2 and obtain a permutation with description size $O(kn \log(1/\delta)$ (or even $O(k \log(1/\delta))$ for certain ranges of k and δ). The Feistel method is not known to be useful for k larger than $2^{n/2}$.

3.3.2 Card shuffling

Consider a process for shuffling cards. Each round (shuffle) in such a procedure selects a permutation on the locations of the $N = 2^n$ cards of a deck (selected from some collection of basic permutations). Starting at an arbitrary ordering of the cards, we are interested at how long does it take to get the deck into a (close to) random position. In other words, a card shuffling defines a Markov chain on the state of the deck, and the goal is to bound its mixing time.

The riffle shuffle models one of the most common "real life" shuffling techniques. Loosely, in each shuffle, the deck is split roughly in the middle, into two sides. Then, cards are dropped sequentially, from both sides, and form a new deck. The mathematical model for this shuffle is due to Gilbert, Shannon and Reeds. Aldous and Diaconis [1] provide a convenient implementation which we shall now describe. Let us view the deck of cards as the set of n bit strings, where each card is a string in $\{0, 1\}^n$. One round of the shuffle consists of two stages: Assign and reorder. In the assign stage, each of the $N = 2^n$ cards is assigned a random bit 0 or 1. In the reorder stage, the cards assigned with 0 are placed at the top, while preserving their internal order. After $O(\log N) = O(n)$ such rounds, the deck is close to uniform, see [1].

The random bits cost of this procedure is quite high. We would need 2^n bits per round, total of

$O(n2^n)$ bits. Observe, that this is of the order of the number of bits needed to select a permutation, uniformly at random (and certainly much more than desired for k-wise independent permutations). An even more troubling difficulty with using this shuffle, is that it is not “oblivious” in the sense that the location of each card is determined by looking at many random bits. For instance, if the ith card is assigned a value of 0, it can still be in any of the first i position after the reorder stage, depending on how many of the first $i-1$ cards are also assigned a 0. As we shall see below, this does not completely preclude the applicability of such a process for generating k-wise independent permutations, but a more straightforward idea is to use an oblivious shuffle.

3.3.3 Oblivious card shuffling

An “old” proposal by Naor [50, page 17], [32] for the construction of k-wise almost independent permutations was to utilize “oblivious” card shuffling procedure. Briefly, a shuffle is oblivious if the location of a card, after each round, is easy to trace and is determined by only a few random bits, say $O(1)$. An excellent example is the Thorp Shuffle [60]. Here the deck is divided into two halves, and these two halves are interleaved in a more local manner than in the riffle shuffle. In the Thorp shuffle, each time we pick one card from each half. With equal probability, the card from the first half is dropped first, and otherwise the card from the second half is dropped first. This means, that the location of a card, after one round, depends on a single bit. It is therefore oblivious, in the sense described above. It was conjectured in [1] that the mixing time of the Thorp Shuffle is $O(n^2)$, but the problem remained open for many years. Recently Morris [30] provided the first *poly*(n) bound on its mixing time.

Definition 10 *(Thorp Shuffle) Let $n \in \mathbb{N}$. Given a deck of 2^n cards, one stage of the shuffle is determined by 2^{n-1} bits that we will view as a random function $g : \{0,1\}^{n-1} \to \{0,1\}$. View the location of each card as an n-bit string according to the lexical order. Card at location (σ, x) where $\sigma \in \{0,1\}$ and $x \in \{0,1\}^{n-1}$ moves to location $(x, \sigma \oplus g(x))$.*

Theorem 3.3.1 *[30] The mixing time for the Thorp shuffle is $O(n^{44})$.*

The idea of Naor [50, page 17], [32] is the following: When using such a card shuffle to construct a k-wise almost independent permutation, all we care for is the final locations of k cards. If we replace the random function g by a k-wise independent function, then this will not change the distribution on the k final locations.

3.3.4 Simple 3-bit permutations

A very intriguing method for generating k-wise δ-dependent permutation was explored first by Gowers [18] and then (with some variation) by Hoory et al. [20] and Brodsky and Hoory [10]. The idea is to pick a few bit positions (actually 3) and chose a permutation on the resulting small cube. In the Hoory et al. variation only a single bit is changed as a function of the other bits. This is reminiscent of a shuffle, but there is no chance that the shuffle will converge in reasonable time (as we invest too few bits in each shuffle). This approach is treated more formally in the Section 3.4.4 and it works very well with the derandomized walk approach, since the underlying set of permutations considered is the simplest and hence the description length of simple permutations is quite short. What this line of research shows is that a composition of not too many simple permutations yields a k-wise almost independent permutation.

3.4 Main results

In this section we give a method for reducing the description length of previous constructions of k-wise δ-dependent permutations. As discussed in the introduction, this method is particularly suited to constructions based on composition of permutations. We apply this method to the simple 3-bit permutations of [10, 18, 20] to obtain k-wise δ-dependent permutations with description length $O(nk + \log(\frac{1}{\delta}))$, for all $k \leq 2^n - 2$. To obtain our results for all $k \leq 2^n$, we apply this method to the Thorp Shuffle [60].

3.4.1 Permutation families and random walks on graphs

We associate with a family $\mathcal{F}$ of permutations a graph as follows:

Definition 11 *(Companion Graph) Let $\mathcal{F} \subseteq P_n$ be a family of permutations. For $k \in \mathbb{N}$, define the companion (multi-)graph of $\mathcal{F}$, $G_{\mathcal{F},k} = (V, E)$ by:*

- $V = [N_k]$.
- $E = \{ (i, \sigma(i)) \mid i \in [N_k], \sigma \in \mathcal{F} \}$.
- *Each edge $(i, \sigma(i)) \in E$ is labelled by σ.*

Remark 2 *For an element $x = (x_1, \ldots, x_k) \in [N_k]$, and a permutation $\sigma \in \mathcal{F}$, we abbreviate $\sigma(x)$ for $(\sigma(x_1), \ldots, \sigma(x_k))$.*

All of our families of permutations of Section 3.3 closed under taking an inverse of a permutation and always include the identity permutation. We summarize the properties of the companion graph in the following proposition:

Proposition 1 *Let $\mathcal{F} \subseteq P_n$ be a family of permutations, which is closed under taking an inverse and contains the identity permutation. Let $k \in \mathbb{N}$. Then, the companion graph $G_{\mathcal{F},k}$, is an undirected, $|\mathcal{F}|$-regular, consistently labelled graph, with self-loops.*

Remark 3 *A consistently labelled graph has the property that for any vertex w, any two incoming edges to w are labelled with different labels.*

Assume that $\mathcal{F}$ is such that $\mathcal{F}^t$ is a family of k-wise δ-dependent permutations. This means that the distribution over the vertices we reach, by taking a walk of length t, starting at any vertex of $G_{\mathcal{F},k}$, is δ-close to uniform. Simply, traversing an edge is the same as applying the permutation that is the label of this edge. Taking t random edges is the same as applying the composition of t randomly chosen permutations.

Derandomizing the family $\mathcal{F}^t$ will mean that instead of composing independently chosen permutations from $\mathcal{F}$, we will select the permutations with some dependencies. Equivalently, we will take a pseudorandom walk instead of a random one. We will use a pseudorandom generator, to generate this walk. We will require that the seed of the generator be sufficiently small and that the number of labels the generator outputs will not be too large (hopefully, not much larger than t). Such a generator was given by Reingold, Trevisan and Vadhan [45, 47].

3.4.2 Pseudorandom walk generators

We now discuss generators for pseudorandom walks on graphs. We will refer to graphs with the following parameters:

Definition 12 *(Parameters for a Graph) Let $G = (V, E)$ be a connected, undirected d-regular graph, on m vertices. Then G is an (m, d, λ)-graph if $\lambda \leq \lambda(G)$.*

Definition 13 *(Pseudorandom Walk) Let $G = (V, E)$ be a d-regular graph where each node labels its adjacent edges in $[d]$. Let $\mathcal{A}$ be a distribution over*

$$\vec{a} = a_1, a_2, \ldots a_\ell \in [d]^\ell.$$

We say that $\mathcal{A}$ is δ-pseudorandom for G, if for every $u \in V$, the distribution on the possible end vertices of a walk in G, which starts from u, and follows the edge labels in $\vec{a}$ is δ-close to uniform when $\vec{a}$ is distributed according to $\mathcal{A}$.

Note that if G is an (m, d, λ) graph, λ is sufficiently smaller than 1 and the walk is sufficiently long, then we expect a (truly) random walk to end in vertex that is close to being uniformly distributed no matter where the walk started. We are now ready to state the parameters of the best known construction of pseudorandom walk generators.

Theorem 3.4.1 *[45, 47][Pseudorandom Walk Generator] For every $m, d \in \mathbb{N}$, $\delta, \epsilon > 0$, there is* *a pseudorandom walk generator PRG where*

$PRG_{m,d,\delta,\epsilon} : \{0,1\}^r \to [d]^\ell$*, with the following parameters:*

- *Seed length* $r = O(\log(md/\epsilon\delta))$*.*
- *Walk length* $\ell = poly(1/\epsilon) \cdot \log(md/\delta)$*.*
- *Computable in space* $O(\log(md/\epsilon\delta))$ *and time* $poly(1/\epsilon, \log(md/\delta))$*.*

such that for every consistently labelled $(m, d, 1-\epsilon)$*-graph* G*, the output of* $PRG(U_r)$ *is* δ*-pseudorandom for* G*, where* U_r *is the uniform distribution on* $\{0,1\}^r$*.*

Remark 4 *The generator of [47] is more general as it also applies to certain types of directed graphs (with in-degree of each vertex equals its out-degree). Here, only undirected regular graphs are relevant. Furthermore, the time-complexity of the generator is only implicit in [47].*

3.4.3 Derandomizing compositions of permutation families

By Proposition 1, the companion graph $G_{\mathcal{F},k}$, is regular and consistently labelled. As argued above, if $\mathcal{F}^t$ (for t not too large) is k-wise almost independent then the random walk on $G_{\mathcal{F},k}$ has small mixing time. By Theorem 3.1.1, this implies a bound on the eigenvalue gap ε. Therefore, Theorem 3.4.1 gives a way to generate a pseudorandom walk for $G_{\mathcal{F},k}$ with $PRG_{m,d,\delta,\epsilon}$ with $m = |[N]_k|$ and $d = |\mathcal{F}|$. The idea is to use each seed $s \in \{0,1\}^r$ of the pseudorandom generator PRG, to define a new permutation σ_s, which is the composition of permutations from $\mathcal{F}$. Theorem 3.4.2 formalizes this approach. For simplicity, we assume that the bound on the eigenvalue gap is given, rather than deducing it by Theorem 3.1.1 (as in the discussion above).

An advantage we have, which affects the parameters of our results (especially the description length), is that the efficiency of the generator of [47] depends on the spectral gap of the initial graph. Since we are using families of permutations for which the companion graph is known to be of good expansion, we manage to achieve non-trivial parameters in the families we construct.

The following theorem describes the family of permutations we achieve.

Theorem 3.4.2 *Let $\mathcal{F} \subseteq P_n$ be a family of size $d = |\mathcal{F}|$, and $G_{\mathcal{F},k}$ be its companion graph. Suppose that gap$(G_{\mathcal{F},k}) = \epsilon$, where ϵ may be a function of n and k. Then, there exists $\mathcal{F}' \subseteq P_n$, such that $\mathcal{F}'$ is a k-wise δ-dependent family, with following properties.*

- *The description length of $\mathcal{F}'$ is $O(nk + \log(\frac{d}{\epsilon\delta}))$.*
- *If the time complexity of any permutation in $\mathcal{F}$ is bounded by $\xi(n,k)$, then the time complexity of $\mathcal{F}'$ is poly$(1/\epsilon, n, k, \log(\frac{d}{\delta})) \cdot \xi(n,k)$.*

Proof:

We apply Theorem 3.4.1 on the companion graph of $\mathcal{F}$. Following Proposition 1 we know that $G_{\mathcal{F},k}$ fits the requirements there. Let $r = O(\log(\frac{2^{nk} \cdot d}{\epsilon\delta}))$ and $\ell = poly(1/\epsilon) \cdot \log(\frac{2^{nk} \cdot d}{\delta})$ be as in Theorem 3.4.1. For a string $s \in \{0,1\}^r$, we define $\sigma_s \in P_n$ as follows. Let $\vec{w} = PRG_{2^{nk},d,\delta,\epsilon}(s) \in [d]^\ell$. Then $\vec{w} = \tau_1, \tau_2, \ldots, \tau_\ell$, where for all $1 \leq i \leq \ell$, $\tau_i \in \mathcal{F}$. We let $\sigma_s = \tau_\ell \circ \ldots \circ \tau_1$.

Next define a permutations family $\mathcal{F}' \subseteq P_n$ by

$$\mathcal{F}' = \{\, \sigma_s \mid s \in \{0,1\}^r \,\}.$$

We now show that $\mathcal{F}'$ is a k-wise δ-dependent family. By Theorem 3.4.1, for any starting vertex $u \in V(G_{\mathcal{F},k})$, the pseudorandom walk starting at u and following the labels of $PRG_{2^{nk},d,\delta,\epsilon}(U_r)$ reaches a vertex that is δ-close to uniform. Observe that picking a random $\sigma_s \in \mathcal{F}'$ and applying it to any value $A \in V(G_{\mathcal{F},k}) = [N_k]$ is exactly as taking a random walk on $G_{\mathcal{F},k}$ according to the output of $PRG_{2^{nk},d,\delta,\epsilon}$ with a random seed s. Therefore, the output of a uniform σ_s on any such $A \in [N_k]$, is δ-close to uniform. We can conclude that $\mathcal{F}'$ is k-wise δ-dependent.

The description length of $\mathcal{F}'$ is $|r| = O(\log(\frac{2^{nk}d}{\epsilon\delta})) = O(nk + \log(\frac{d}{\epsilon\delta}))$. The time complexity of $\mathcal{F}'$ depends on the time complexity of running the generator, and of running permutations from $\mathcal{F}$. This can be bounded by $poly(1/\epsilon, n, k, \log(\frac{d}{\delta})) \cdot \xi(n,k)$. ■

3.4.4 Particular Derandomization – 3-bit Permutations

We now provide a formal definition and analysis of simple 3-bit permutations, mentioned in Section 3.3.4.

Definition 14 (Simple Permutations) *[20] Let $w \leq n$. For $i \in [n], J = \{j_1, \dots, j_w\} \subseteq [n] \smallsetminus \{i\}$, and a function $f \in \{0,1\}^w \to \{0,1\}$, denote by $\sigma_{i,J,f}$ the permutation*

$$\sigma_{i,J,f}(x_1, \dots, x_n) \doteq (x_1, \dots, x_{i-1}, x_i \oplus f(x_{j_1}, \dots, x_{j_w}), x_{i+1} \dots, x_n)$$

The following simple permutation family $\mathcal{F}_w$ is defined by

$$\mathcal{F}_w = \{\sigma_{i,J,f} | i \in [n], J \subseteq [n] \smallsetminus \{i\}, |J| = w, f \in \{0,1\}^w \to \{0,1\}\}.$$

We denote by $\mathcal{F}_2$ the simple permutations family $\mathcal{F}_w$ for $w = 2$.

Theorem 3.4.3 *[10] For all $2 \leq k \leq 2^n - 2$, ${\mathcal{F}_2}^t$ is k-wise δ-dependent , for $t = O(n^2k(nk + log(\frac{1}{\delta})))$. Furthermore, $gap(G_{\mathcal{F},k}) = \Omega(\frac{1}{n^2k})$.*

Evaluating $\sigma_{i,J,f} \in \mathcal{F}_2$ takes $O(n)$ time. The size of $\mathcal{F}_2$ is $O(n^3)$, and the size of ${\mathcal{F}_2}^t$ is $O(n^3)^t = n^{O(n^2k(nk+log(\frac{1}{\delta})))}$. It follows that ${\mathcal{F}_2}^t$ has description length $O(n^2k(nk + log(\frac{1}{\delta}))\log(n))$, and time complexity $O(n^3k(nk + log(\frac{1}{\delta})))$.

Combining Theorems 3.4.3 and 3.4.2 we obtain the main result of this paper, for $k \leq 2^n - 2$.

Theorem 3.4.4 *For all $k \leq 2^n - 2$, there exists $\mathcal{F} \subseteq P_n$, such that $\mathcal{F}$ is k-wise δ-dependent. $\mathcal{F}$ has description length $O(nk + \log(\frac{1}{\delta}))$, and time complexity $poly(n, k, \log(\frac{1}{\delta}))$.*

Proof: Consider the permutations family $\mathcal{F}_2$. The size of $\mathcal{F}_2$ is $d = O(n^3)$, and the spectral gap of its companion graph is $\epsilon = \Omega(\frac{1}{n^2k})$. Applying Theorem 3.4.2 on $\mathcal{F}_2$, we get a permutations family $\mathcal{F}'$, whose description length is $O(nk + \log(\frac{d}{\epsilon\delta})) = O(nk + \log(\frac{1}{\delta}))$.

Since the time complexity of any permutation in $\mathcal{F}_2$ is $O(n)$, it follows that the time complexity of $\mathcal{F}'$ is $poly(n, k, \log(\frac{1}{\delta}))$. ■

3.4.5 Particular Derandomization – Thorp Shuffle

In order to obtain the main result of this paper, for all $k \leq 2^n$, we analyze the Thorp Shuffle [60], described in Subsection 3.3.3.

Theorem 3.4.5 *For all $k \leq 2^n$, there exists $\mathcal{F} \subseteq P_n$, such that $\mathcal{F}$ is k-wise δ-dependent. $\mathcal{F}$ has description length $O(nk + \log(\frac{1}{\delta}))$, and time complexity poly$(n, k, \log(\frac{1}{\delta}))$.*

Proof: We denote by $\mathcal{F}$ the permutations family implementing the Thorp shuffle, as explained in Definition 10. Let us analyze its size, and the spectral gap of its companion graph.

By the idea of Naor (explained following Theorem 3.3.1) the size of the permutations family for the Thorp shuffle is $d = O(2^{nk})$.

By Theorem 3.3.1, the mixing time of the Thorp shuffle for a constant error is $O(n^{44})$. Using Lemma 3.1.2, we see that the mixing time for error ζ, $\tau(\zeta) = O(n^{44} \lg \frac{1}{\zeta})$. As described in the discussion following Theorem 3.1.1, $gap(G) = \Omega(\frac{\ln(\frac{1}{2\zeta})}{\tau(\zeta)})$, and so the spectral gap of the companion graph is $\epsilon = gap(G) = \Omega(\frac{1}{n^{44}})$.

Applying Theorem 3.4.2 on $\mathcal{F}$, we get a permutations family $\mathcal{F}'$, whose description length is $O(nk + \log(\frac{d}{\epsilon\delta})) = O(nk + \log(\frac{1}{\delta}))$.

Since the time complexity of any permutation in $\mathcal{F}$ is easily seen to be $poly(n, k)$, it follows that the time complexity of $\mathcal{F}'$ is $poly(n, k, \log(\frac{1}{\delta}))$. ∎

3.5 Further work

One issue that we have not resolved, is coming up with k-wise permutations where the time complexity of evaluating at a given point is small. Note that even for k-wise independent functions this issue is not completely resolved; the basic construction based on polynomials is expensive and some lower and upper bounds are given by Siegel [54]. In general, the transformation we propose via the random walks does not preserve the time complexity of evaluating permutations in $\mathcal{F}$: When the composed permutation is stored in its succinct form, we do not know how to evaluate it at a given point without first ‘decompressing’ and representing explicitly as a composition of ℓ permutations in $\mathcal{F}$.

In order to maintain the complexity of evaluation, we need a generator with ‘random access’ properties. In such a generator, evaluating the ith bit of its output, does not entail computing all bits up to i. The Nisan generator [33] has some aspects of this nature, but is sub-optimal in other parameters.

ACKNOWLEDGMENTS

First and foremost, I would like to thank Dr. Amnon Ta-Shma, for his guidance, endless knowledge and expertise in the field, sharp observations, devotion and energy. This thesis could not have been completed without his help.

I would like to extend my gratitude to Prof. Omer Reingold and Prof. Moni Naor of the Weizmann Institute of Science, who invited me to join their research on pseudorandom permutations, and spent a lot of time and patience guiding me.

Last but not least, I would like to thank my family, for their love and support.

REFERENCES

[1] D. Aldous and P. Diaconis, *Shuffling cards and stopping times*, American Mathematical Monthly, vol. 93, 1986, pp. 333-348.

[2] D. Aldous and J. A. Fill, *Reversible Markov chains and random walks on graphs*, http://www.stat.berkeley.edu/users/aldous/RWG/book.html.

[3] N. Alon and J. Spencer, **The Probabilistic Method**, Wiley, New York, 1992.

[4] Y. Azar, R. Motwani and J. Naor, *Approximating probability distributions using small sample spaces*, Combinatorica, vol. 18(2), 1998, pp. 151-171.

[5] B. Barak, R. Impagliazzo and A. Wigderson, *Extracting randomness using few independent sources*, FOCS 2004, pp. 384-393.

[6] B. Barak, G. Kindler, R. Shaltiel, B. Sudakov and A. Wigderson, *Simulating independence: New constructions of condensers, Ramsey graphs, dispersers, and extractors*, STOC 2005, pp. 1-10.

[7] A. Bar-Noy, J. Naor and B. Schieber, *Pushing dependent data in clients-providers-servers systems*, Wireless Networks, vol. 9(5), 2003, pp. 421-430.

[8] J. Black and P. Rogaway, *Ciphers with arbitrary finite domains*, Topics in Cryptology - CT-RSA 2002, LNCS, vol. 2271, Springer, 2002, pp. 114-130.

[9] A. Z. Broder, M. Charikar, A. M. Frieze and M. Mitzenmacher, *Min-wise independent permutations*, Proc. of the 90th Annual ACM Symposium on Theory of Computing, 1998, pp. 327-336.

[10] A. Brodsky and S. Hoory, *simple permutations mix even better*, Arxiv math.CO/0411098.

[11] P. J. Cameron, *Finite permutation groups and finite simple groups*, Bull. London Math. Soc., vol. 13, 1981, pp. 1-22.

[12] M. R. Capalbo, O. Reingold, S. P. Vadhan and A. Wigderson, *Randomness conductors and constant-degree lossless expanders*, joint session: STOC 2002, and IEEE Conference on Computational Complexity 2002.

[13] B. Chor and O. Goldreich, *Unbiased bits from sources of weak randomness and probabilistic communication complexity*, SIAM Journal on Computing, vol. 17(2), 1988, pp. 230-261.

[14] A. Cohen and A. Wigderson, *Dispersers, deterministic amplification, and weak random sources (extended abstract)*, FOCS 1989, pp. 14-19.

[15] A. C. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan and M. Strauss, *Fast, small-space algorithms for approximate histogram maintenance*, STOC 2002, pp. 389-398.

[16] O. Goldreich, S. Goldwasser and A. Nussboim, *On the implementation of huge random objects*, FOCS 2003, pp. 68-79.

[17] O. Goldreich and A. Wigderson, *Tiny families of functions with random properties: A quality-size trade-off for hashing*, Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR94-002, 1994, Revised December 1996.

[18] W. T. Gowers, *An almost m-wise independent random permutation of the cube*, Combinatorics, Probability and Computing, vol. 5(2), 1996, pp. 119-130.

[19] R. Gradwohl, G. Kindler, O. Reingold and A. Ta-Shma, *On the error parameter of dispersers*, RANDOM 2005.

[20] S. Hoory, A. Magen, S. Myers and C. Rackoff, *Simple permutations mix well*, The 31st International Colloquium on Automata, Languages and Programming (ICALP), 2004.

[21] P. Indyk, *Stable distributions, pseudorandom generators, embeddings and data stream computation*, FOCS 2000, pp. 189-197.

[22] T. Itoh, Y. Takei and J. Tarui, *On permutations with limited independence*, SODA 2000, pp. 137-146.

[23] T. Itoh, Y. Takei and J. Tarui, *On the sample size of k-restricted min-wise independent permutations and other k-wise distributions*, STOC 2003, pp. 710-719.

[24] A. Lubotzky, R. Phillips and P. Sarnak, *Ramanujan graphs*, Combinatorica, vol. 8(3), 1988, pp. 261-277.

[25] D. Koller and N. Megiddo, *Constructing small sample spaces satisfying given constraints*, SIAM J. Discrete Math., vol. 7(2), 1994, pp. 260-274.

[26] M. Luby and C. Rackoff, *How to construct pseudorandom permutations and pseudorandom functions*, SIAM J. Comput., vol. 17, 1988, pp. 373-386.

[27] U. M. Maurer and K. Pietrzak, *The security of many-round Luby-Rackoff pseudo-random permutations*, EUROCRPYT 2003, LNCS vol. 2656, Springer, pp. 544-561.

[28] U. M. Maurer and K. Pietrzak, *Composition of random systems: When two weak make one strong*, First Theory of Cryptography Conference, TCC 2004, LNCS vol. 2951, Springer, pp. 410-427.

[29] S. Myers, *Black-box composition does not imply adaptive security*, Advances in Cryptology - EUROCRYPT 2004, LNCS, vol. 3027, Springer, pp. 189-203.

[30] B. Morris, *On the mixing time for the Thorp shuffle*, STOC 2005, pp. 403-412.

[31] R. Motwani and P. Raghavan, **Randomized Algorithms**, Cambridge University Press, New York, 1995.

[32] M. Naor, O. Reingold, *On the construction of pseudorandom permutations: Luby-Rackoff revisited*, Journal of Cryptology, vol. 12(1), Springer-Verlag, 1999, pp. 29-66.

[33] N. Nisan, *Pseudorandom generators for space-bounded computation*, Combinatorica 12(4), 1992, pp. 449-461.

[34] N. Nisan and D. Zuckerman, *Randomness is linear in space*, Journal of Computer and System Sciences, vol. 52(1), 1996, pp. 43-52.

[35] J. Patarin, *Improved security bounds for pseudorandom permutations*, 4th ACM Conference on Computer and Communications Security, 1997, pp. 142-150.

[36] J. Patarin, *Luby-Rackoff: 7 rounds are enough for $2^{n(1-\epsilon)}$ security.* CRYPTO 2003, pp. 513-529.

[37] J. Patarin *Security of random Feistel schemes with 5 or more rounds*, CRYPTO 2004, pp. 106-122.

[38] K. Pietrzak, *Composition does not imply adaptive security*, Advances in Cryptology, CRYPTO 2005, LNCS, Springer, to appear.

[39] B. Pinkas, *Communication preserving cryptographic protocols*, PhD dissertation, 1999, Weizmann Institute of Science.

[40] J. Radhakrishnan and A. Ta-Shma, *Tight bounds for depth-two superconcentrators*, FOCS 1997, pp. 585-594.

[41] R. Raz and O. Reingold, *On recycling the randomness of states in space bounded computation*, STOC 1999, pp. 159-168.

[42] R. Raz, O. Reingold and S. P. Vadhan, *Error reduction for extractors*, FOCS 1999, pp. 191-201.

[43] R. Raz, O. Reingold and S. P. Vadhan, *Extracting all the randomness and reducing the error in Trevisan's extractors*, Journal of Computation and System Sciences, vol. 65(1), 2002, pp. 97-128.

[44] E. G. Rees, *Notes on Geometry*, Springer, Berlin, 1983.

[45] O. Reingold, *Undirected ST-Connectibvity in log-space*, STOC 2005, pp. 376-385.

[46] O. Reingold, R. Shaltiel and A. Wigderson, *Extracting randomness via repeated condensing*, FOCS 2000, pp. 22-31.

[47] O. Reingold, L. Trevisan and S. Vadhan, *Pseudorandom walks in biregular graphs and the RL vs. L problem*, Electronic Colloquium on Computational Complexity (ECCC), Technical Report TR05-022, accepted on February 2005.

[48] O. Reingold, S. P. Vadhan and A. Wigderson, *Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors*, Electronic Colloquium on Computational Complexity (ECCC), vol. 8(18), 2001.

[49] D. J. S. Robinson, **A course in the theory of groups – 2nd ed.**, Springer-Verlag, New York, 1996.

[50] S. Rudich, *Limits on the provable consequences of one-way functions*, PhD Thesis, U. C. Berkeley.

[51] A. Russell and H. Wang, *How to fool an unbounded adversary with a short key*, EUROCRYPT 2002, pp. 133-148.

[52] M. Saks, A. Srinivasan, S. Zhou and D. Zuckerman, *Low discrepancy sets yield approximate min-wise independent permutation families*, Information Processing Letters, vol. 73, 2000, pp. 29-32.

[53] R. Shaltiel, *Recent developments in explicit constructions of extractors*, Bulletin of the EATCS, vol. 77, 2002, pp. 67-95.

[54] A. Siegel, *On universal classes of extremely random constant-time hash functions*, SIAM Journal on Computing, vol. 33(3), 2004, pp. 505-543.

[55] A. Sinclair, *Improved bounds for mixing rates of Markov chains and multicommodity flow*, Combinatorics, Probability and Computing, vol. 1(4), Cambridge University Press, 1992, pp. 351-370.

[56] D. Sivakumar, *Algorithmic derandomization via complexity theory*, STOC 2002, pp. 619-626.

[57] A. Ta-Shma, *Storing information with extractors*, Information Processing Letters, vol. 83(5), 2002, pp. 267-274.

[58] A. Ta-Shma, C. Umans and D. Zuckerman, *Loss-less condensers, unbalanced expanders and extractors*, STOC 2001, pp. 143-152.

[59] A. Ta-Shma, D. Zuckerman and S. Safra, *Extractors from Reed-Muller codes*, FOCS 2001, pp. 638-647.

[60] E. Thorp, *Nonrandom shuffling with applications to the game of Faro*, Journal of the American Statistical Association, vol. 68, 1973, pp. 842-847.

[61] L. Trevisan, *Construction of extractors using pseudo-random generators*, STOC 1999, pp. 141-148.

[62] D. Zuckerman, *Linear degree extractors and the inapproximability of max clique and chromatic number*, Electronic Colloquium on Computational Complexity (ECCC), 2005.

[63] D. Zuckerman, *Simulating BPP using a general weak random source*, Algorithmica, vol. 16(4/5), 1996, pp. 367-391.

[64] D. Zuckerman, *Randomness-optimal oblivious sampling*, Random Structures and Algorithms, vol. 11(4), 1997, pp. 345-367.