Derandomized Constructions of k-Wise (Almost) Independent Permutations[*]

Eyal Kaplan[†] Moni Naor[‡] Omer Reingold[§]

Abstract

Constructions of k-wise almost independent permutations have been receiving a growing amount of attention in recent years. However, unlike the case of k-wise independent functions, the size of previously constructed families of such permutations is far from optimal. This paper gives a new method for reducing the size of families given by previous constructions. Our method relies on pseudorandom generators for space-bounded computations. In fact, all we need is a generator, that produces "pseudorandom walks" on undirected graphs with a consistent labelling. One such generator is implied by Reingold's log-space algorithm for undirected connectivity [35, 36]. We obtain families of k-wise almost independent permutations, with an optimal description length, up to a constant factor. More precisely, if the distance from uniform for any k tuple should be at most δ, then the size of the description of a permutation in the family is $O(kn + \log \frac{1}{\delta})$.

1 Introduction

In explicit constructions of pseudorandom objects, we are interested in simulating a large random object using a succinct one and would like to capture some essential properties of the former. A natural way to phrase such a requirement is via limited access. Suppose the object that we are interested in simulating is a random function $f : \{0,1\}^n \mapsto \{0,1\}^n$ and we want to come up with a small family of functions G that simulates it. The k-wise independence requirement in this case is that a function g chosen at random from G be 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 of its choice. We can also relax the requirement and talk about *almost* k-wise independence by requiring that the advantage of a distinguisher be limited by some δ.

[*] A preliminary version of this paper appeared in Random 2005.

[†] Tel-Aviv University. email: kaplaney@post.tau.ac.il

[‡] Incumbent of the Judith Kleeman Professorial Chair, Department of Computer Science and Applied Math, The Weizmann Institute of Science, Rehovot 76100, Israel; email: moni.naor@weizmann.ac.il. Research supported in part by a grant from the Israel Science Foundation.

[§] Incumbent of the Walter and Elise Haas Career Development Chair, Department of Computer Science, Weizmann Institute of Science, Rehovot 76100, Israel omer.reingold@weizmann.ac.il. Research supported by US-Israel Binational Science Foundation Grant 2002246.

Families of functions that are k-wise independent (or almost independent) were constructed and applied extensively in the computer science literature (see [3, 25]). 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.

Suppose now that the object we are interested in constructing is a *permutation*, i.e. a 1-1 function $g : \{0,1\}^n \mapsto \{0,1\}^n$, 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. Similarly, there is an algebraic trick when $k = 3$ (we learned it from Schulman, private communication in [26], see also [40, 44]). 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 $O(kn + \log(\frac{1}{\delta}))$ [*]; this is what a random (non-explicit) construction gives (see Section 3.2). There are a number of proposals in the literature of constructing k-wise almost independent permutations (see Section 4), 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.

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. One motivation for this study is the relation between k-wise independent permutations and block ciphers [14, 26].

In block-ciphers, modelled by pseudorandom permutations, the distinguisher is not limited by the *number of calls* to the permutations but rather by its computational power. Still, the two notions are related. On one hand, some constructions of pseudorandom permutations, and most notably the Luby-Rackoff construction [20], imply explicit constructions of k-wise almost independent permutations [26] (see references therein). On the other hand, Hoory et al. [14] study a construction in terms of k-wise independence, partially with the motivation of understanding the way “cryptographic” pseudorandomness may be obtained. Furthermore, k-wise independence is sometimes sufficient for cryptographic applications, and may be easier to obtain (e.g. Pinkas [33]). Below, we illustrate one such case (partially related to a motivating example given by Black and Rogaway [6]).

Suppose that you want to permute the set of all credit card numbers to reduce fraud. You would like to construct a permutation on the set of credit card numbers (of size roughly 2^{40}, ignoring the first 4 digits). Only trusted servers will have access to the permutation. The goal is that an adversary

[*] 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). If for no other reason, $\log(\frac{1}{\delta})$ bits are needed to reduce roundoff errors. This lower bound also follows for more significant reasons, unless k-wise *exactly* independent permutations can be constructed.

who sees a limited number of permuted credit card numbers and the original numbers (say its own cards) would not be able to obtain information on any other card for which it sees only the permuted value. Furthermore, we would like to spread the permutation among the trusted servers at low cost (to save communication). This means, that the permutation should be represented by a small number of bits. Note that for this range even under cryptographic assumptions there is no ready made solution. For instance, DES is a permutation on 2^{64} values that is presumed pseudorandom, at least for sufficiently weak machines. However, it is not clear how to use it in order to construct a permutation on 2^{40} values. This example may also point out practical values for which an efficient solution is needed. While our main interest is description length, we discuss time efficiency in Section 6.

Our Technique and Main Results: 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.

Consider a family of permutations $\mathcal{F}$, with rather small description length s. We denote by $\mathcal{F}^t$ the family of permutations obtained by composing every 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.

Our first idea uses generators that fool bounded space computations for the task of choosing the subset of $\mathcal{F}^t$, as we describe below. Pseudorandomness for space-bounded computation has been a very productive area, see [27, 28]. Such pseudorandomness has been used before in the context of combinatorial constructions where space is not an *explicit* issue by Indyk [15] and by Sivakumar [43].

Let g be the composition of t uniformly and independently selected $f_1, f_2, \ldots, f_t$ in $\mathcal{F}$. Let us also consider g' which is the composition of t permutations $f'_1, f'_2, \ldots, f'_t$ in $\mathcal{F}$, selected in some other manner. Assume that the distribution on g' is not k-wise almost independent. This means that there are k inputs $x_1, x_2, \ldots x_k$ such that the distribution $g'(x_1), g'(x_2), \ldots g'(x_k)$ in not close enough to uniform. That is, there exists a test $\mathcal{T}$ that distinguishes $g'(x_1), \ldots g'(x_k)$ from uniform. On the other hand, by our assumption, $g(x_1), \ldots g(x_k)$ *is* close to uniform, therefore $\mathcal{T}$ also distinguishes $g'(x_1), \ldots g'(x_k)$ from $g(x_1), \ldots g(x_k)$. This translates to a test that distinguishes the distribution of $f'_1, f'_2, \ldots, f'_t$ from uniform. The key observation is that the distinguisher uses only space kn as a branching program (i.e., it is of width 2^{kn}). Therefore, if $f'_1, f'_2, \ldots, f'_t$ are selected by a generator that fools space-kn computations then no such distinguisher exists and g' is k-wise almost independent, with a shorter description length than $t \cdot s$.

To complete this argument let us describe the small space distinguisher for the distribution $f'_1, f'_2, \ldots, f'_t$. Consider a protocol for t parties, where party i receives h_i as input and altogether the parties want to distinguish the case that the h_i's are uniformly distributed from the case that they are distributed according to the distribution $f'_1, f'_2, \ldots, f'_t$. Party i will only be allowed to send nk bits to party $i + 1$. Such communication network is equivalent to a branching program of space nk and the known pseudorandom generators for space bounded computations work against distinguishers in this model. The distinguisher operates as follows. The first party applies h_1 to $x_1, \ldots x_k$ and sends $\vec{z}_1 = (h_1(x_1), \ldots h_1(x_k))$. At its turn, party $i > 1$ implies h_i to the sequence $\vec{z}_{i-1}$ received from party $i-1$ to obtain $\vec{z}_i$ that it sends to party $i+1$. At the end, party t evaluates $\vec{z}_t$ and outputs $\mathcal{T}(\vec{z}_t)$. We note the following facts: (1) Each $\vec{z}_i$ is kn-bit long and thus this is indeed a space kn distinguisher. (2) If the h_i's are uniformly distributed then $\vec{z}_t$ is distributed according to $g(x_1), \ldots g(x_k)$. Otherwise it is distributed according to $g'(x_1), \ldots g'(x_k)$. As $\mathcal{T}$ behaves differently on these two distributions, we obtain the correctness of our small space distinguisher.

Given an "ideal" generator the fools space bounded computations and has optimal parameters we could expect the method above to give k-wise almost independent permutations with description length $O(nk + \log(\frac{1}{\delta}) + s + \log t)$. Based on previous constructions of k-wise almost independent permutations this implies description length $O(nk + \log(\frac{1}{\delta}))$ as desired. However, applying this derandomization method with currently known generators (which are not optimal) implies description length $(nk + \log(\frac{1}{\delta}))$ *times poly-logarithmic factors*.

This leads us to our second idea: to obtain families with description length $O(nk + \log(\frac{1}{\delta}))$ we revise the above method to use a more restricted derandomization tool: we use *pseudorandom generators for walks on undirected labelled graphs*. That is walks which are indistinguishable from a random walk for any 'consistently labelled graph' and sufficient length. Such generators with sufficiently good parameters are implied by the proof that undirected connectivity is in logspace of Reingold [35], and made explicit by Reingold, Trevisan and Vadhan [36].

Adaptive vs. Static Distinguishers: Consider a distinguisher, trying to guess whether the permutation it has is random or from the family G. Assume further, that the distinguisher is allowed to make k queries to the permutation. A natural issue, is whether these queries are chosen ahead of time (statically) or adaptively, as a function of the responses the process receives. When considering *perfect* k-wise independent permutation there is no difference between the two cases, but when considering almost k-wise independent permutations there could be a large difference[†]. Nonetheless, here we shall consider the static case. This is in general enough, for at least two reasons. First, static indistinguishability up to distance $\delta 2^{-nk}$ implies adaptive indistinguishability up to distance δ. Second, a result of Maurer and Pietrzak [22] shows that composing two independently chosen k-wise almost independent permutations in the static case gives k-wise almost independent permutations with adaptive queries with similar parameters[‡].

Related Work: There are several lines of constructions that are of particular relevance to our work. We describe them in more detail in Section 4. The information is summarized in Table 1.

Another notion which has been studied quite extensively in recent years is that of *min-wise independence* introduced by Broder et al. [7]. Informally, a permutation family is k-restricted min-wise independent (or simply min-wise independent, if $k = n$), if for every distinct k elements, each element is mapped to the minimum among the images of the elements, with equal probability. The motivation for this notion stems from studying resemblance between documents on the Web

[†] One of our favorite examples is involutions (permutations where the cycle length is at most 2). A random involution is almost pairwise for the static case with $\varepsilon = O(1/2^n)$, but for the adaptive case $\varepsilon = 1 - O(1/2^n)$.

[‡] Note that this is a case where k-wise independence is different from cryptographic pseudorandomness, as was demonstrated in recent papers by Myers and Pietrzak [23, 32].

Table 1: Summary of Results and Previous Work on k-wise δ-dependent Permutations.

Family	Description Length	Range of Queries
Feistel[§] (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 [9, 13, 14]	$O(n^2 k(nk + lg(\frac{1}{\delta}))\lg(n))$	$k \leq 2^n - 2$
Thorp Shuffle [24, 26, 39]	$O(n^{45} k \log(\frac{1}{\delta}))$	$k \leq 2^n$
Non-Explicit Constructions:		
Probabilistic (Thm. 3.4)	$O(nk + \log(\frac{1}{\delta}))$	$k \leq 2^n$
Sample space existence (Thm. 3.5)	$O(nk)$	$k \leq 2^n$
This Work (Theorem 5.9)	$O(nk + \log(\frac{1}{\delta}))$	$k \leq 2^n$

(see Broder et al. [8, 7]). This notion is weaker than k-wise independence. Another definition, k-rankwise independence [16], demands that the k elements are mapped to any order with the same probability. k-rankwise independence is stronger than k-restricted min-wise independence, but weaker than k-wise independence. The best lower bound for k-restricted min-wise independence is from [17] and is roughly $n^{k/2}$. For a more extensive treatment we refer the reader to [7, 16, 17].

Organization

In Section 2 we provide notation and some basic information regarding random walks and the spectral gap of graphs. In Section 3 we define k-wise δ-dependent permutation, argue the (non-constructive) existence of small families of such permutations and study the composition of such permutations. In Section 4 we discuss some known families of permutations. Section 5 describes our general construction of a permutation family, and proves our main result. In Section 6 we describe possible extensions for future research.

2 Preliminaries and Notation

- Let P_n be the set of all permutations over $\{0,1\}^n$. We will use $N = 2^n$.
- Let x and y be two bit strings of equal length, then $x \oplus y$ denotes their bit-by-bit exclusive-or.
- For any $f, g \in P_n$ denote by $f \circ g$ their composition (i.e., $f \circ g(x) = f(g(x))$).
- For a set Ω, denote by U_Ω the uniform distribution on the elements of Ω.
- Denote by $[N_k]$ the set of all k-tuples of *distinct* n-bit strings.

[§] The first row is based on 4 rounds with the first and last being pair-wise independent [26]. Analysis of related constructions [22, 30, 31] 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.)

2.1 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}. Undirected graphs that are connected, regular, and have self-loops in each vertex, 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 2.1 (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 2.2 (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 2.3 *[41] Let $G = (V, E)$, M, π be as in Definition 2.2. 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 2.3 in order to bound the gap of the graph from below. A simple calculation using Theorem 2.3 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 (Theorem 3.8). 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 $O(\delta^2)$-close to uniform.

Theorem 2.4 *([2] Chapter 2, Lemma 20) Let $G = (V, E)$, M, π be as in Definition 2.2. 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 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. Most of this paper concentrates on permutations over bit strings and we consider more general domains in Section 6.2

3.1 Definitions

The output of a k-wise almost independent permutation on any k inputs is δ-close to random, where "closeness" is measured by statistical variation distance between distributions.

Definition 3.1 (Statistical Distance) *Let D_1, D_2 be distributions over a finite set Ω. The variation distance between D_1 and D_2 is*

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

We say that D_1 and D_2 are δ-close if $\|D_1 - D_2\| \leq \delta$.

Remark 3.2 Note that if two distributions are δ-close then there is no distinguisher (not even an inefficient one) that can distinguish the distributions with advantage better than δ.

Definition 3.3 *Let $n, k \in \mathbb{N}$, and let $\mathcal{F} \subseteq P_n$ be a family of permutations (we allow repetitions). 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 the permutation efficient in terms of time complexity. See additional discussion in Section 6.

3.2 Non-Explicit Constructions

We show the existence of *non-explicit* families of permutations that are k-wise almost independent. Our goal in the other sections would be to obtain families of size which is as close as possible to that obtained by the non-explicit arguments below. The first idea for showing the existence of families of k-wise δ-dependent is simply to consider a probabilistic construction, i.e. a random collection of permutations of a certain size. The following theorem follows by the approximation method of Azar, Motwani and Naor [4]. They provide ([4] Theorem 3.1) a general way to approximate an arbitrary distribution over a finite set Γ. Their point is that the weighted average of for ℓ different weights can be approximated to within ϵ simultaneously by a sample space of size $O(\frac{\log \ell}{\epsilon^2})$ and uniform distribution over the support. Consider the sample space Γ consisting of all permutations and D is the uniform distribution. To specify the requirements of k-wise δ-dependency we need for all $(x_1, \ldots, x_k), (y_1, \ldots, y_k) \in [N_k]$ an approximation that should be within $\delta/|[N_k]|$. We get the following:

Theorem 3.4 *Let $n \in \mathbb{N}$. For all $1 \leq k \leq 2^n$ and $\delta > 0$ there exists a family of permutations $\mathcal{F}$ that is k-wise δ-dependent and is of size $O(\frac{nk2^{2nk}}{\delta^2})$.*

The existence (even with a non-explicit construction) of *exact* k-wise family of permutations is unknown. Nonetheless, we show that there exist a distribution on permutations, which is k-wise independent and has a small support. The construction follows a result by Koller and Megiddo [19], which we briefly describe below.

Their idea for constructing a small sample space for a given object was to consider the set of constraints it induces in terms of values of subsets. Then argue that if a sample space satisfying these constraints exists, then there exists an assignment where the number of non-zero points is no larger than the number of constraints.

In the case of k-wise independent permutations, we are defining a probability distribution over permutations π, i.e. for each permutation we want to assign a probability p_π. For every two k-tuples $\bar{x} = (x_1, x_2, \ldots x_k) \in [N_k]$ and $\bar{y} = (y_1, y_2, \ldots y_k) \in [N_k]$ we have the constraint that the probability that the chosen permutation π satisfies $y_i = \pi(x_i)$ for $1 \leq i \leq k$ is exactly $1/\binom{N}{k}$. Let $C_{\bar{x},\bar{y}} = \{\pi | y_i = \pi(x_i) \ \forall 1 \leq i \leq k\}$. One can write for each $\bar{x}, \bar{y} \in [N_k]$ this requirement as a linear constraint in the p_π's:

$$\sum_{\pi \in C_{\bar{x},\bar{y}}} p_\pi = \frac{1}{\binom{N}{k}}.$$

These $\binom{N}{k}^2$ constraints plus the constraint $\sum_\pi p_\pi = 1$ completely characterize k-wise independence. We know that there is an assignment satisfying all these constraints: simply make all $p_\pi = 1/N!$. As Koller and Megiddo [19] argue, this means that there is also a non-negative assignment, where the number of non-zero values is at most the number of constraints; since it is non-negative it defines a probability distribution. Unfortunately, we do not know how to construct this distribution, or to sample from it in time polynomial in n and k. By the above discussion, we have the following:

Theorem 3.5 (Existence of k-wise Independent Distribution) *There exists a distribution on permutations which is k-wise independent (i.e. for any k points the value of the chosen permutation is uniform in $[N_k]$) and the size of the support of the distribution is at most 2^{2nk}.*

3.3 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 3.6 *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 3.7 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 Theorem 3.6 will show, starting with a family $\mathcal{F}$ which is δ-dependent results in $\mathcal{F}^t$ which is only $(O(\delta))^t$-dependent. Therefore, increasing the description length and time complexity linearly, pays off in an exponential decay of the error. We now state our composition theorem.

Theorem 3.8 *Let $\mathcal{F}$ be a k-wise δ-dependent family. Then, $\mathcal{F}^2$ is a k-wise $2\delta^2$-dependent family. Furthermore, for every $\ell \in \mathbb{N}$, $\mathcal{F}^\ell$ is a k-wise $(\frac{1}{2}(2\delta)^\ell)$-dependent family.*

The proof of Theorem 3.8 uses a certain type of graph which is associated with a permutation family $\mathcal{F}$. The graph, which we call a *companion graph*, has a vertex for each k-tuple of $[N_k]$. For every two k-tuples $\bar{x} = (x_1, x_2, \ldots x_k) \in [N_k]$ and $\bar{y} = (y_1, y_2, \ldots y_k) \in [N_k]$ and every permutation $\sigma \in \mathcal{F}$ such that $y_i = \sigma(x_i)$ for $1 \leq i \leq k$ we have an edge in the companion graph between $\bar{x}$ and $\bar{y}$. This edge is labelled by σ. More formally:

Definition 3.9 (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 3.10 For an element $\bar{x} = (x_1, \ldots, x_k) \in [N_k]$, and a permutation $\sigma \in \mathcal{F}$, we abbreviate $\sigma(\bar{x})$ for $(\sigma(x_1), \ldots, \sigma(x_k))$.

Observe, that a step on the companion graph is equivalent to evaluating a permutation from $\mathcal{F}$ on the elements of the k-tuple.

Proof: (of Theorem 3.6) 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 2.4. Then $d(1) = \delta$, and since by Theorem 2.4, $d(2) \leq 2d(1)^2 = 2\delta^2$, we conclude that $\mathcal{F}^2$ is a k-wise $2\delta^2$-dependent family. Applying Theorem 2.4 inductively we have that $d(t) \leq (\frac{1}{2}(2\delta)^\ell)$. Therefore, $\mathcal{F}^\ell$ is a k-wise $(\frac{1}{2}(2\delta)^\ell)$-dependent family. □

4 Short Survey of Explicit Constructions

As mentioned in the Introduction, for $k = 2$ the set of linear permutations is a good construction (see also [26]), and for $k = 3$ using sharply 3-transitive permutation groups[*] (as suggested by Leonard Schulman (private communication)) is a good construction. Unfortunately, from the classification of finite simple groups it follows that for $k \geq 6$ there are no k-transitive groups over $[n]$ other than the symmetric group S_n and the alternating group A_n and there are only few such groups for $k = 4$ and $k = 5$ (see [10, 37]). To conclude, for $k \geq 4$ any small family of k-wise independent permutations is *not* a permutation group (i.e. is not closed under composition and inverse). This is a major hurdle in providing efficient algebraic constructions of k-wise independent permutations, for $k \geq 4$. Note also that from Theorem 3.8 (Composition Theorem) we can also conclude that a (non-trivial) permutation group cannot even be k-wise δ-dependent for any $\delta < 1/2$: since the error can be reduced sufficiently to imply k-transitivity and if the set of permutations is a group, then it is preserved under composition.

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.

4.1 Feistel Based Constructions

In their famed work, Luby and Rackoff [20] showed how to construct pseudorandom permutations from pseudorandom functions. The construction is based on the *Feistel Permutation*: For any function $f \in \{0,1\}^{n/2} \mapsto \{0,1\}^{n/2}$ the Feistel Permutation is defined by $(L, R) \mapsto (R, L \oplus f(R))$, where $|L| = |R| = n/2$. The construction uses a composition of several such permutations.

Naor and Reingold [26] 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 [26], Patarin [29, 30, 31], and Maurer and Pietrzak [21]).

The 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 parameter δ one can use Theorem 3.8 and obtain a permutation with description length $O(kn \log(1/\delta)$ (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 $2^{n/2}$.

[*] A permutation group over the set $[N] = \{1, 2, \ldots, N\}$ is a subgroup of the symmetric group S_n. A permutation group G over $[n]$ is k-transitive if for every two k-tuples $\{x_1, \ldots, x_k\}$ and $\{y_1, \ldots, y_k\}$ of distinct elements of $[n]$ there exist a permutation $\pi \in G$ such that $\forall 1 \leq i \leq k,\ \pi(x_i) = y_i$. A permutation group G over $[n]$ is sharply k-transitive if for every two such tuples there exists exactly one permutation $\pi \in G$ such that $\forall 1 \leq i \leq k,\ \pi(x_i) = y_i$. A sharply k-transitive permutation group is in particular k-wise independent. Indeed for $k = 2$, the linear permutations form a sharply 2-transitive permutation group. For $k = 3$, there are known constructions of sharply 3-transitive permutation groups.

4.2 Card Shuffling

Consider a process for shuffling cards. Each round (shuffle) in such a procedure selects a permutation on the locations of the N cards of a deck (selected from some collection of basic permutations). Starting at an arbitrary ordering of the cards, we are interested at the number of rounds it takes to get the deck into a (close to) random ordering. 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.

Oblivious Card Shuffling: Call a shuffle *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* [45]. 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 [24] provided the first $poly(n)$ bound on its mixing time. More formally

Definition 4.1 (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 $g : \{0, 1\}^{n-1} \mapsto \{0, 1\}$. 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 4.2 *[24] The mixing time for the Thorp shuffle is $O(n^{44})$.*

An "old" proposal by the second author [39, page 17], [26] for the construction of k-wise almost independent permutations was to utilize oblivious card shuffling procedure. The idea 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. Therefore, the obliviousness of card shuffles is useful when constructing k-wise almost independent permutation, in terms of both the description length and time complexity.

Implementing permutations via the riffle shuffle: Even though the riffle shuffle is *not* oblivious there is a way of using it to construct k-wise almost independent permutations. The idea is to generate the choices for each position in a *range-summable* manner: there should be an efficient way to determine the number of '1's in a given range (for a given $1 \leq x \leq N$ how many '1' where chosen for the cards in $[1 \ldots x]$). We need the choices and random variables of the range-sum to be k-wise independent. Once this property exists, then the result is indistinguishable from a random riffle for any process that examines the location of at most k cards.

There is a construction satisfying these properties based on a 'divide-and-conquer' tree. This is described in [11] (due to Naor and Reingold) and [12]. The advantage of this construction over the Thorp shuffle is the lower round complexity, $O(n^2)$ vs. $O(n^{44})$. Both are amenable to the random walk derandomization.

4.3 Simple 3-Bit Permutations

A very intriguing method for generating k-wise δ-dependent permutation was explored first by Gowers [13] and then (with some variation) by Hoory et al. [14] and Brodsky and Hoory [9]. The idea is to pick a few bit positions, three to be concrete, which are the only bits the permutation is going to change. The three bits that are changed define a small sub-cube (with eight elements). To completely define the permutation, select a random permutation on this small sub-cube.[†] This is reminiscent of a shuffle, but here we invest only a few bits in each round. Therefore, the shuffle cannot converge quickly to a random permutation. What this line of research shows is that a composition of not too many simple permutations still yields a k-wise almost independent permutation. This approach is treated more formally in the Section 5.4 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.

5 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 [9, 13, 14] to obtain k-wise δ-dependent permutations with description length $O(nk + \log(\frac{1}{\delta}))$.

[†] In the Hoory et al. variation the permutation is selected in a more restricted manner: Only a single bit is changed as a random function of the other bits.

5.1 Permutation Families and Random Walks on Graphs

Recall from Section 3.3 that we associate with a family $\mathcal{F}$ of permutations a *companion graph* (Def. 3.9) by connecting a k-tuples to $\bar{x}$ to $\sigma(\bar{x})$ for $\sigma \in \mathcal{F}$. All of the families of permutations of Section 4 are closed under taking an inverse of a permutation and always include the identity permutation. We summarize the properties of the companion graph that we need in the following proposition:

Proposition 5.1 *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, with self-loops. Furthermore, the companion graph is consistently labelled graph, in the sense that for every vertex v, every two incoming edges into w have distinct labels.*

Assume that $\mathcal{F}$ is such that $\mathcal{F}^t$ is a family of k-wise δ-dependent permutations. We claim 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 labelled π from the vertex $\bar{x}$ is the same as applying the permutation π on $\bar{x}$ (i.e., it reaches vertex $\pi(\bar{x})$). Taking t random edges is the same as applying the composition of t randomly chosen permutations. If there is any starting point $\bar{x}$ that does not yield an end-point that is δ-close to uniform, then this $\bar{x}$ is a witness to the non k-wise δ-dependency of $\mathcal{F}^t$.

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. The seed of the pseudorandom generator will be required to be sufficiently small and the number of labels the generator outputs will not be too large. Such a generator was given by Reingold, Trevisan and Vadhan [35, 36].

5.2 Pseudorandom Walk Generators

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

Definition 5.2 (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 5.3 (Pseudorandom Walk) *Let $G = (V, E)$ be a d-regular graph where for each node its d outgoing edges take distinct labels 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 a previously known construction of pseudorandom walk generators.

Theorem 5.4 *[35, 36][Pseudorandom Walk Generator] For every $m, d \in \mathbb{N}$, $\delta, \epsilon > 0$, there is a pseudorandom walk generator $PRG = PRG_{m,d,\delta,\epsilon} : \{0,1\}^r \to [d]^\ell$, 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 5.5 The generator of Reingold, Trevisan and Vadhan [36] is more general as it also applies to regular *directed* graphs (where the in-degree and out-degree of each vertex equals some fixed d). Here, only undirected regular graphs are relevant. Furthermore, the time-complexity of the generator is only implicit in [36].

5.3 Derandomizing Compositions of Permutation Families

We now describe our main construction which consists of applying the pseudorandom walk generators for the companion graph of a family of permutations $\mathcal{F}$. Our starting point is any family of permutations $\mathcal{F}$ where $\mathcal{F}^t$ (for t not too large) is k-wise almost independent. By Proposition 5.1, the companion graph $G_{\mathcal{F},k}$, is regular and consistently labelled. As argued following Proposition 5.1, if $\mathcal{F}^t$ is k-wise almost independent then the random walk on $G_{\mathcal{F},k}$ has small mixing time. By Theorem 2.3, this implies a bound on the eigenvalue gap ε of $G_{\mathcal{F},k}$. Therefore, Theorem 5.4 gives us a pseudorandom walk generator for $G_{\mathcal{F},k}$ ($PRG = PRG_{m,d,\delta,\epsilon}$ with $m = |[N]_k|$, $d = |\mathcal{F}|$, ε comes from the analysis of $\mathcal{F}$ and δ from how close to uniform we want the result to be). We now 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 the permutations from $\mathcal{F}$ that $PRG(s)$ generates. The set of all possible seeds defines our new family $\mathcal{F}'$. Theorem 5.6 formalizes this approach:

An advantage we have, which affects the parameters of our results (especially the description length), is that the efficiency of the generator of [36] 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 5.6 *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 the 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 5.4 on the companion graph of $\mathcal{F}$. Following Proposition 5.1 we know that $G_{\mathcal{F},k}$ fits the requirements of Theorem 5.4. 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 5.4. 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 permutation 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 5.4, 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 $poly(1/\epsilon, n, k, \log(\frac{d}{\delta})) \cdot \xi(n,k)$. □

For simplicity, we assumed in the above theorem that the bound ε on the eigenvalue gap is given, rather than deducing it by Theorem 2.3 (as in the discussion before the theorem). But in principal what this theorem tells us is that instead of taking truly independent choices in $\mathcal{F}^t$ it *always* makes sense (from description length point of view) to use PRG to define the permutations that are composed.

5.4 Particular Derandomization – 3-bit Permutations

We now provide a formal definition and analysis of simple 3-bit permutations, mentioned in Section 4.3.

Definition 5.7 (Simple Permutations) *[14] Let $w \leq n$. For $i \in [n]$, $J = \{j_1, \ldots, j_w\} \subseteq [n] \smallsetminus \{i\}$, and a function $f \in \{0,1\}^w \rightarrow \{0,1\}$, denote by $\sigma_{i,J,f}$ the permutation*

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

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

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

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

Theorem 5.8 *[9] 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 5.8 and 5.6 we obtain the main result of this paper:

Theorem 5.9 *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 $\epsilon = \Omega(\frac{1}{n^2 k})$. Applying Theorem 5.6 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 $poly(n, k, \log(\frac{1}{\delta}))$. □

6 Discussion and Further Work

6.1 Time Complexity of the Construction

The focus of this paper is the description length of k-wise almost independent permutations. Still our derandomization preserves the time-complexity of the permutations up to factors that are polynomial in the original time complexity and in the description length ($nk + \log(\frac{1}{\delta})$). One disadvantage of the approach of using a pseudorandom walk generator for derandomization is that we replace a permutation composed of ℓ simple permutations with another permutation composed of $\ell' \gg \ell$ simple permutations (this disadvantage is somewhat less extreme when using the more efficient pseudorandom walk generator recently given in [38]). In this respect it is better to derandomize using generators against general space-bounded computations (such as the Nisan generator [27]) as explained in the introduction. While this approach is slightly sub optimal in terms of description length (using currently known generators) it is quite efficient in terms of time complexity.

A more subtle concern in terms of time complexity is the following: Can we have k-wise almost independent permutations where the time complexity is independent of k (as the description length is larger than nk this only makes sense if we allow direct access to this description). Note that even for k-wise independent functions this issue is not completely resolved; the basic construction based on polynomials is expensive and more efficient constructions have longer descriptions (some lower and upper bounds are given by Siegel [42]). Assume now that we are starting with a construction of k-wise almost independent permutations that has this strong efficiency requirement. When derandomizing with a generator against space bounded computations, the only additional cost is the evaluation of the generator. In order for our derandomization to preserve such strong efficiency we need a pseudorandom 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. More specifically, it should be possible to compute each bit in time that is independent of k and only depends on n. Also note that since the only additional costs are in the evaluation of the pseudorandom generator, one can first "decompress" the succinct description of the derandomized permutations in order to speed up future computations (this may be useful in case storage is not expensive but randomness and communication are).

6.2 Permutations over Other Domains

An issue that we did not explore so far, is constructing k-wise independent permutations over domains that are not powers of 2. This problem was raised by Bar-Noy and S. Naor inspired by the needs of [5]. As was pointed out by Black and Rogaway [6], the credit card problem described in the introduction is in fact one on a domain size that is *not* a power of 2. Black and Rogaway [6] suggested several methods, for obtaining a pseudo-random permutation on domain size M, that is not a power of 2, from a pseudo-random permutation on domain size N, that is a power of 2 (say $N = 2^{\lceil logM \rceil}$). The most relevant method for our purposes is the 'cycle walking' one, where the idea is to construct a permutation on $[M]$ elements by iterating a permutation on $[N]$ until it lands in the first M values of $[N]$. In more details, let $\pi' : [N] \mapsto [N]$. Then $\pi : [M] \mapsto [M]$ is defined for $x \in [M]$ by $\pi(x) = \pi'^{(i)}(x)$ where $i \geq 1$ is the smallest value such that $\pi'^{(i)}(x) \in [M]$.

When one translates this construction to k-wise almost independent permutations, then the requirement on the underlying permutation π' is, that it should be k'-wise δ'-dependent for some $k' \geq k$ (we will see the requirement on δ' momentarily), since some of the evaluations of π require more than a single call to π'. Note also that this mapping requires that π' be immune to adaptive attacks. In general, consider the 'bad' case for a k tuple $x_1, x_2, \ldots x_k$ in $[M]$: the evaluation of π on $x_1, x_2, \ldots x_k$ requires more than k' calls to π'. If $M/N \geq 1/2$, then the probability that this bad case happens, is proportional to an exponential in $k' - 2k$, by a Chernoff bound. Conditioned on the event that the bad case did *not* happen, then the distribution of π on $x_1, x_2, \ldots x_k$ is δ'-far from uniform on $[M_k]$. Hence, the resulting set of permutations is k-wise δ-dependent for δ that is larger than δ' by an additive factor, which is exponential is $k' - 2k$.

This analysis means, that for large k it is relatively easy to get a small error, by taking k' to be, say, $2k$, without significantly increasing the family size. However, when k is small, the resulting error is too large. In this case, as before, the derandomized walk method is applicable for reducing the error, since Theorem 5.6 does not require the domain size to be a power of 2.

6.3 Further Questions

One interesting question is whether it is possible to 'scale down' a construction for k-wise independent permutations on n bits to one on $n' \leq n$ bits. When n' is very close to n then some of the techniques described in the previous section (such as cycle walking) are relevant, but they become inefficient when $n - n'$ is larger than logarithmic. This is most relevant in the computational pseudorandomness setting: is it possible to obtain from a block-cipher on large blocks (e.g. 128 bits) a block-cipher on small blocks (e.g. 40 bits), while maintaining the security of the former.

Finally, there is no strong reason to suppose that explicit small families (or distributions) of *exact* k-wise independent permutation do not exist and Theorem 3.5 hints to their existence. So how about finding them?

Acknowledgments

The authors are grateful to Ronen Shaltiel for his invaluable collaboration during the early stages of this work, and thank Danny Harnik, Asaf Nussboim and Adam Smith for useful comments.

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, 1992.

[4] Y. Azar, R. Motwani and J. Naor, *Approximating Probability Distributions Using Small Sample Spaces*, Combinatorica 18(2), 1998, pp. 151–171.

[5] A. Bar-Noy, J. Naor and B. Schieber, *Pushing Dependent Data in Clients-Providers-Servers Systems*, Wireless Networks 9(5), 2003, pp. 421–430.

[6] J. Black and P. Rogaway, *Ciphers with Arbitrary Finite Domains*. Topics in Cryptology - CT-RSA 2002, Lecture Notes in Computer Science, vol. 2271, Springer, 2002, 114–130.

[7] A. Z. Broder, M. Charikar, A. M. Frieze and M. Mitzenmacher, *Min-wise independent permutations*, Journal of Computer and System Sciences, 60(3), 2000, pp. 630–659 (preliminary version STOC 2000).

[8] A. Z. Broder S. C. Glassman, M. S. Manasse and Geoffrey Zweig, *Syntactic clustering of the Web*, Computer Networks 29, 1997, pp. 1157–1166.

[9] A. Brodsky and S. Hoory, *Simple Permutations Mix Even Better*, Arxiv math.CO/0411098.

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

[11] A. C. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan and M. Strauss, *Fast, small-space algorithms for approximate histogram maintenance*, Proc. of the 34th Annual ACM Symposium on Theory of Computing, 2002, pp. 389–398.

[12] O. Goldreich, S. Goldwasser and A. Nussboim, *On the Implementation of Huge Random Objects*, Proc. 44th Annual IEEE Symposium on Foundations of Computer Scienc, 2003, pp. 68–79.

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

[14] S. Hoory, A. Magen, S. Myers and C. Rackoff, *Simple permutations mix well*, The 31st International Colloquium on Automata, Languages and Programming (ICALP), Lecture Notes in Computer Science 3142, Springer, 2004, pp. 770–781.

[15] P. Indyk, *Stable Distributions, Pseudorandom Generators, Embeddings and Data Stream Computation*, Proc. 41st Annual IEEE Symposium on Foundations of Computer Scienc, 2000, pp. 189–197.

[16] T. Itoh, Y. Takei and J. Tarui, *On permutations with limited independence*, Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, 2000, pp. 137–146.

[17] T. Itoh, Y. Takei and J. Tarui, *On the sample size of k-restricted min-wise independent permutations and other k-wise distributions*, Proc. of the 35th Annual ACM Symposium on Theory of Computing, 2003, pp. 710–719.

[18] E. Kaplan, M. Naor and O. Reingold, *Derandomized Constructions of k-Wise (Almost) Independent Permutations*, The 9th International Workshop on Randomization and Computation (RANDOM), Lecture Notes in Computer Science 3624, Springer, 2005, pp. 354–365.

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

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

[21] U. M. Maurer and K. Pietrzak, *The Security of Many-Round Luby-Rackoff Pseudo-Random Permutations*, Advances in Cryptology - EUROCRPYT '2003, Lecture Notes in Computer Science 2656, Springer, pp. 544–561.

[22] U. M. Maurer and K. Pietrzak, *Composition of Random Systems: When Two Weak Make One Strong*, First Theory of Cryptography Conference, TCC 2004, Lecture Notes in Computer Science 2951, Springer, pp. 410–427.

[23] S. Myers, *Black-Box Composition Does Not Imply Adaptive Security*, Advances in Cryptology - EUROCRYPT '2004, Lecture Notes in Computer Science, vol. 3027, Springer, pp. 189–203.

[24] B. Morris, *On the mixing time for the Thorp shuffle*, Proc. of the 37th Annual ACM Symposium on Theory of Computing, 2005, pp. 403–412.

[25] R. Motwani and P. Raghavan, **Randomized Algorithms**, Cambridge University Press, New York (NY), 1995.

[26] M. Naor, O. Reingold, *On the Construction of Pseudorandom Permutations: Luby-Rackoff Revisited*, J. of Cryptology, vol. 12(1), Springer-Verlag, 1999, pp. 29–66.

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

[28] N. Nisan and D. Zuckerman, *Randomness is Linear in Space*, J. Comput. Syst. Sci. vol. 52(1), 1996, pp. 43–52.

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

[30] J. Patarin, *Luby-Rackoff: 7 Rounds Are Enough for $2^{n(1-\epsilon)}$ Security*. Advances in Cryptology - CRYPTO 2003, Lecture Notes in Computer Science 2729, Springer, pp. 513–529.

[31] J. Patarin *Security of Random Feistel Schemes with 5 or More Rounds*, *Advances in Cryptology - CRYPTO'2004*, Lecture Notes in Computer Science 3152, Springer, pp. 106–122.

[32] K. Pietrzak, *Composition Does Not Imply Adaptive Security*, Advances in Cryptology - CRYPTO'2005, Lecture Notes in Computer Science 3621, Springer, pp. 55–65.

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

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

[35] O. Reingold, *Undirected ST-Connectibvity in Log-Space*, Proc. of the 37th Annual ACM Symposium on Theory of Computing, 2005, pp. 376–385.

[36] O. Reingold, L. Trevisan, S. Vadhan, *Pseudorandom Walks in Biregular Graphs and the RL vs. L Problem*, ECCC, TR05-022, 2005.

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

[38] E. Rozenman and S. Vadhan, *Derandomized Squaring of Graphs*, The 9th International Workshop on Randomization and Computation (RANDOM), Lecture Notes in Computer Science 3624, Springer, 2005, pp. 436–447.

[39] S. Rudich, *Limits on the provable consequences of one-way functions*, PhD Thesis, 1988, U. C. Berkeley.

[40] A. Russell, H. Wang, *How to fool an unbounded adversary with a short key*, Advances in Cryptology - EUROCRYPT'2002, Lecture Notes in Computer Science 2332, Springer, 2002, pp. 133-148.

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

[42] A. Siegel, *On Universal Classes of Extremely Random Constant-Time Hash Functions*, SIAM Journal on Computing 33(3), 2004, pp. 505–543.

[43] D. Sivakumar, *Algorithmic derandomization via complexity theory*, Proc. of the 34th Annual ACM Symposium on Theory of Computing, 2002, pp. 619–626

[44] 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.

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