# Deterministic Coupon Collection and Better Strong Dispersers*

**Raghu Meka[1], Omer Reingold[1], and Yuan Zhou[2]**

1 **Microsoft Research Silicon Valley**
{meka,omer.reingold}@microsoft.com

2 **Computer Science Department, Carnegie Mellon University**
yuanzhou@cs.cmu.edu

## Abstract

Hashing is one of the main techniques in data processing and algorithm design for very large data sets. While random hash functions satisfy most desirable properties, it is often too expensive to store a fully random hash function. Motivated by this, much attention has been given to designing *s*mall families of hash functions suitable for various applications. In this work, we study the question of designing space-efficient hash families $\mathcal{H} = \{h : [U] \to [N]\}$ with the natural property of *c*overing: $\mathcal{H}$ is said to be covering if any set of $\Omega(N \log N)$ distinct items from the universe (the *c*oupon-collector limit) are hashed to cover all $N$ bins by most hash functions in $\mathcal{H}$. We give an explicit family $\mathcal{H}$ of size $\mathrm{poly}(N)$ (which is optimal), so that hash functions in $\mathcal{H}$ can be specified efficiently by $O(\log N)$ bits.

We build covering hash functions by drawing a connection to *dispersers*, which are quite well studied and have a variety of applications themselves. We in fact need *strong dispersers* and we give new constructions of strong dispersers which may be of independent interest. Specifically, we construct strong dispersers with optimal entropy loss in the high min-entropy, but very small error ($\mathrm{poly}(n)/2^n$ for $n$ bit sources) regimes. We also provide a strong disperser construction with constant error but for any min-entropy. Our constructions achieve these by using part of the source to replace seed from previous non-strong constructions in surprising ways. In doing so, we take two of the few constructions of dispersers with parameters better than known extractors and make them strong.

**1998 ACM Subject Classification** F.2.2 Nonnumerical Algorithms and Problems

**Keywords and phrases** Coupon collection; dispersers, strong dispersers, hashing, pseudorandomness

**Digital Object Identifier** 10.4230/LIPIcs.APPROX-RANDOM.2014.872

## 1 Introduction

Hashing is one of the main techniques in data processing and algorithm design for handling large data sets. When processing data from a universe $U$, to avoid various computational bottlenecks such as storage and load distribution it is often helpful to hash down to a smaller universe (aka bins), via a hash family $\mathcal{H} = \{h : h : U \to [N]\}$. Different applications call for different requirements of the hash family $\mathcal{H}$ and there is a rich body of work on constructions of families with various properties starting with the seminal work of Carter and Wegman [6].

One such prominently studied property is *load-balancing* where one desires no bin to receive too many items. In this paper, we consider the related property of *covering*. Besides

[*] Work done while the third author was visiting Microsoft Research Silicon Valley.

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) /
18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 872–884
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

being a natural and desirable property by itself, the *covering* property is also useful in achieving load balancing. A classical question in probability theory is the *coupon collector problem*. Suppose that there are $N$ different coupons and at each trial you get one random coupon. How many trials do you need in expectation before you collect all $N$ coupons? The answer of course is $\Theta(N \log N)$. An implication of this is that if we randomly hash $\Omega(N \log N)$ distinct objects from a universe to $N$ bins, then with high probability, the objects cover all the bins. This motivates the following definition formulated by Alon et al. [2][1].

▶ **Definition 1.** A family of hash functions $\mathcal{H} = \{h : U \to [N]\}$ is $\epsilon$-covering if there exists a constant $C$ such that for all subsets $S \subseteq U$, $|S| \geq CN \log N$,

$$\Pr_{h \in_u \mathcal{H}} [h(S) = [N]] \geq 1 - \epsilon.$$

We say $\mathcal{H}$ is covering if it is $1/2$-covering.

The coupon collector argument shows that fully random functions satisfy $\epsilon$-covering property with $\epsilon = N^{-\Omega(1)}$. However, fully random hash functions are inefficient in practice as we need space $|U|$ to describe them and space is critical in many scenarios where hashing is helpful. We address the question of designing efficient hash families with covering property as above which are small or equivalently can be sampled with, and hence described by, few bits. As the covering property of fully random hash functions follows from the coupon collection problem, one can intuitively view our end goal as a *derandomization* of the classical coupon collection process.

Standard families like $O(\log N)$-wise independent hash functions (see preliminaries for formal definition) which are known to have strong load-balancing properties are also $N^{-\Omega(1)}$-covering. However, such hash families have size $N^{O(\log N)}$. Similar parameters were achieved by Alon et al. [2] by using random linear hash functions. The work of Celis et al. [7] gives efficient covering hash families of size $N^{O(\log \log N)}$. Here, we solve the problem by giving the first polynomial size, efficient (logarithmic evaluation time[2]) hash family:

▶ **Theorem 2.** *Let $N > 0$ and $c > 0$. Then, there exists an $N^{-c}$-covering family of hash functions $\mathcal{H} = \{h : U \to [N]\}$ with $|\mathcal{H}| = ((\log |U|) \cdot N)^{O(1)}$. Moreover, each $h \in \mathcal{H}$ can be evaluated in time $O(\log N)$ from a description of length $O(\log N)$.*

Our construction of such hash families relies on a simple connection between covering hash families and the well studied concept of *randomness dispersers*. While the connection itself is not surprising, the strong dispersers we need were not known before and we give new explicit constructions which are interesting by themselves. In the following subsection, we first briefly recall the basic notions of randomness extractors and dispersers, and introduce our new constructions of dispersers. We then describe the relation to covering hash families.

## 1.1 Extractors and Dispersers

Extractors and dispersers are important combinatorial objects with many applications throughout computer science. Informally, an extractor is a function which takes a *biased* source of randomness with some entropy and outputs ("extracts") a distribution that is close to the uniform distribution. To achieve this, we are allowed to use a few additional random

[1] Throughout, $x \in_u X$ denotes a uniformly random element from a multi-set $X$.
[2] In the standard unit-cost RAM model.

bits (*seed*), which is necessary. Here, the entropy is quantified by the notion of *min-entropy*: for any random variable $X$ over a universe $U$, the min-entropy of $X$ is defined by

$$H_\infty(X) = \min_{a \in U} \log \left( \frac{1}{\Pr[X = a]} \right).$$

We quantify closeness between random variables via the standard statistical distance denoted $\Delta( \, , \, )$.

▶ **Definition 3** (Extractor, [20]). For $k, \epsilon > 0$, a $(k, \epsilon)$-extractor[3] is a function $\mathsf{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that for any random variable $X$ over $\{0,1\}^n$ with $H_\infty(X) \geq k$, we have $\Delta(X, U_n) \leq \epsilon$.[4]

We say the extractor is *strong* if the extractor output is close to uniform even given the seed:

$$\Delta((\mathsf{Ext}(X, U_d), U_d), (U_m, U_d)) \leq \epsilon.$$

We refer to the parameter $d$ as the seed-length of the extractor and say the extractor is explicit if the function $\mathsf{Ext}$ can be computed in time which is polynomial in $n$. We refer to $k + d - m$ as the *entropy loss* of the extractor as this corresponds intuitively to the number of random bits we lose in the extraction process; for strong extractors, the entropy loss is defined to be $k - m$.

Closely related to extractors are dispersers, which can be seen as a relaxation of extractors where instead of requiring the output distribution to be close to uniform, we only require the output distribution to have large support.

▶ **Definition 4** (Disperser). For $k, \epsilon > 0$, a $(k, \epsilon)$-disperser is a function $Dsp : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that for any random variable $X$ over $\{0,1\}^n$ with $H_\infty(X) \geq k$, we have $|\text{Supp}(Dsp(X, U_d))| \geq (1 - \epsilon)2^m$.

We say the disperser is *strong* if the output of the disperser has large support for most seeds:

$$|\text{Supp}((Dsp(X, U_d), U_d))| \geq (1 - \epsilon)2^{m+d}.$$

Similar to extractors, we refer to the parameter $d$ as the seed-length of the disperser and say the disperser is explicit if the function $Dsp$ can be computed in time which is polynomial in $n$. We refer to $k + d - m$ as the entropy loss of the disperser; for strong dispersers, the entropy loss is defined to be $k - m$.

Over the past few decades, extractors and dispersers have been the focus of intense study. In particular, because of their many pseudo-random properties, extractors and dispersers have by now become standard tools in complexity theory and algorithm design with numerous applications: e.g., error-correcting codes (e.g., [27]), cryptography (e.g., [9], [17]) and pseudorandom generators (e.g., [20], [5]). We refer to the recent survey of Vadhan [28], and the references therein for more details.

It can be shown by the probabilistic method that a) there exist strong extractors with seed-length $d = \log(n - k) + 2\log(1/\epsilon) + O(1)$ and entropy loss of $2\log(1/\epsilon) + O(1)$; b) there exist strong dispersers with seed-length $d = \log(n - k) + \log(1/\epsilon) + O(1)$ and entropy loss of $\log\log(1/\epsilon) + O(1)$. These bounds were also shown to be tight up to additive $O(1)$ terms by Radhakrishnan and Ta-Shma [21]. However, most applications of extractors and dispersers require explicit constructions and most effort in the area has been towards giving explicit

[3] We often omit $n, m$ as they will be clear from context.

[4] Throughout, $U_n$ denotes the uniform distribution on $\{0,1\}^n$.

constructions matching the above bounds. Indeed, we do have several strong and nearly optimal constructions in many important parameter regimes (see [14], [10] for some of the most recent constructions). However, we do not yet have the best constructions in many other regimes and this is especially so when dealing with very small errors, which is what we need. Here we address the construction of strong dispersers with very small error and show the following.

▶ **Theorem 5.** *For all $n, 0 \leq \epsilon \leq 1/2, c \geq 1$, there exists an explicit $(n - c, \epsilon)$-strong disperser $Dsp : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(2^c \log(1/\epsilon))$ and entropy loss $(n - c) - m = \log\log(1/\epsilon) + c + O(1)$.*

The main feature of the above construction is the optimal (up to an additive constant) bound on entropy loss for *any* error parameter $\epsilon$, in particular, even for error as small as $2^{-n}/\text{poly}(n)$). Previously, explicit dispersers with optimal entropy loss as above were known [13], but they were not strong. Being able to handle very small errors and having strong dispersers will both be important for the application to covering hash families.

In addition, using a different set of ideas, we also build a strong disperser with small entropy loss for all min-entropies but for constant error $\epsilon$.

▶ **Theorem 6.** *For all $n$, $k$, and constant $\epsilon$, there exists an explicit $(k, \epsilon)$-strong disperser $Dsp : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n)$ and entropy loss $k - m = 3\log n + O(1)$.*

While we do not use the above construction here, it follows a similar theme as in Theorem 5 and could be of independent interest.

## 2 Techniques

Let us first examine the connection between covering hash families and strong dispersers. Let $\mathcal{H} = \{h : U \to [N]\}$ be a $\epsilon$-covering family of hash functions. Let $U \equiv \{0,1\}^n$ and $[N] \equiv \{0,1\}^m$. Then, we can define a function $Dsp : \{0,1\}^n \times \mathcal{H} \to \{0,1\}^m$ by $Dsp(x, h) = h(x)$. Clearly, $Dsp$ can be viewed as a plausible disperser with seed-length $d = \log(|\mathcal{H}|)$ and conversely, any disperser with these parameters defines a corresponding hash family $\mathcal{H}$ from $\{0,1\}^n$ to $\{0,1\}^m$.

By setting up the definitions appropriately, it is not hard to show that $\epsilon$-covering hash families imply $(k, \epsilon)$-strong dispersers for $k = \log_2(N \log N) + O(1) = m + \log_2 m + O(1)$ and $\epsilon = N^{-O(1)}$. Similarly, $(k, \epsilon)$-strong dispersers as above imply $\epsilon$-covering hash families. However, note that the entropy loss of the dispersers we want is

$$k - m = \log_2 m + O(1) = \log\log_2(1/\epsilon) \pm O(1).$$

Therefore, $\epsilon$-covering hash functions necessitate strong dispersers with optimal entropy loss (up to $O(1)$ additive term) and very small error. We achieve such hash functions by appealing to Theorem 5. The actual construction proceeds in two steps.

We first hash the universe $U$ to $[CN \log N]$ bins for a sufficiently large constant $C$ so that the number of distinct bins hit is $\Omega(N \log N)$. We do so by using almost $O(1)$-wise independent hash functions. In the terminology of extractors and dispersers, this step can be seen as *condensing* the source so as to boost the relative min-entropy of the source. In particular, to obtain covering hash families from strong dispersers as outlined in the above argument, we now only need a disperser which works for entropy deficiency at most

$$\log(CN \log N) - \log(\Omega(N \log N)) = O(1),$$

as exactly achieved in Theorem 5. Thus, to get our final covering hash family, we hash from $[CN \log N]$ to $[N]$ bins using the strong disperser from Theorem 5.

We next discuss our constructions of dispersers.

## 2.1 Strong Dispersers

As remarked earlier, the main problem with using existing constructions for the dispersers we want is that the known constructions are not strong. This difference is not crucial for extractors as most known constructions are either strong or can be made strong via the reduction of Raz, Reingold and Vadhan [23]. No such generic reductions are known for dispersers.

The main insight in our constructions is to use the source to replace part of the seed from the previous non-strong constructions. We will shortly discuss why this is useful. This usually does not work with some notable exceptions being the works of Gabizon, Raz, Shalitel [11] and Barak et al. [3], Barak et al. [4]. Each of our constructions achieve this in a different way and a different analysis.

For the high entropy construction (Theorem 5), we use the techniques of Gradwohl et al. [13] which in turn rely on the classical *expander walk theme*. Roughly speaking, the disperser of Gradwohl et al. is obtained as follows. They first associate the source strings with the vertices of an expander and then compute the output of the disperser by taking a certain walk as specified by the seed on the expander graph. However, their construction implicitly involves a *guess* for how many steps to take in the random walk and this makes their constructions non-strong. We in turn use a part of the source to determine how many steps to take. This causes two problems. Firstly the part of the source we use is not fully random. Secondly, and perhaps more seriously, this also induces probabilistic dependencies between the starting point of the walk and the *edge labels* for the random walk. However, we show that the expander walk we take is robust enough to handle these issues.

For the general entropy disperser, our construction relies on the basic idea of splitting the source into a *block-wise source* which has been used in many constructions of extractors and dispersers. However, most previous constructions *guessed* a good splitting point for the source and this is the reason why they seem inherently non-strong as most of the guesses are usually wrong. Our approach is to first extract $\log n$ bits from the $n$-bit source and use them to determine the splitting point. We then argue that, despite the subtle dependencies introduced by this step, known constructions of extractors and dispersers for block-wise sources are robust enough to still work.

We give more details of our constructions along with the previous ones we build on in the corresponding sections.

# 3 Preliminaries

We start with some basic notations and definitions from probability.

- We use $U_n$ to denote the uniform distribution over $\{0,1\}^n$.
- Given a discrete random variable $X$, we use $\mathrm{Supp}(X)$ to denote the support of $X$, i.e. the set of elements $a$ such that $\Pr[X = a] > 0$. We call a distribution $X$ *flat* if $X$ is the uniform distribution over $\mathrm{Supp}(X)$.
- The statistical or variational distance between random variables $X, Y$ over a universe $U$ is defined as
$$\Delta(X,Y) = \max_{A \subseteq U} \left|\Pr[X \in A] - \Pr[Y \in A]\right|.$$

We say that $X, Y$ are $\epsilon$-close (or $X$ is $\epsilon$-close to $Y$) if $\Delta(X, Y) \leq \epsilon$.

We shall use the following easy fact.

▶ **Fact 7.** *If $X$ is $\epsilon$-close to $U_n$, then $|\mathrm{Supp}(X)| \geq (1 - \epsilon)2^n$.*

We shall also use the following basic tools from pseudorandomness.

## 3.1 Expander Graphs

▶ **Definition 8** (Expander Graphs). Let $G = (V, E)$ be a regular graph with normalized adjacency matrix $A$. We call $G$ a $(D, \lambda)$-expander if $G$ is $D$-regular and the second largest eigenvalue (in absolute value) of $A$ is at most $\lambda$. We say $G$ is explicit if there exists an algorithm that, given any vertex $v \in V$, and an index $i \in [D]$, computes the $i$-th neighbor of $v$ in time $\mathrm{poly}\log|V|$.

Explicit expanders with almost optimal trade-off between degree $D$ and expansion $\lambda$ are constructed (see, e.g. the work by Lubotzky, Philips and Sarnak [18]). In this paper, we only use the fact that for every constant $\lambda$, there exists a constant $D$ and explicit $(D, \lambda)$-expanders for every $V$. Explicit constructions of such expanders where the evaluation can be done with $O(1)$ word operations are known (for example Margulis's expander, see [16]).

We will use the following sampling lemma essentially from [12] (the version we state with better constants and different sets $S_1, \ldots, S_t$ follows easily from [15]).

▶ **Theorem 9.** *Let $G = (V, E)$ be a $(D, \lambda)$-expander on $V$. Consider a random walk $X_0, X_1, \ldots, X_t$ on $G$, where $X_0$ is a random start vertex in $V$. Then, for all $S_1, S_2, \ldots, S_t \subseteq V$ with $\mu = \left(\sum_{i=1}^{t} |S_i|/|V|\right)/t$,*

$$\Pr[\forall i,\ X_i \notin S_i] \leq \exp\left(-\mu^2(1 - \lambda)t/4\right).$$

## 3.2 Hash Functions with Limited Independence

▶ **Definition 10** ($k$-wise independent Hash Functions). A hash family $\mathcal{H} = \{h\ :\ h : U \to [N]\}$ is $\delta$-almost $k$-wise independent if for all distinct $u_1, \ldots, u_k \in U$, and $h \in_u \mathcal{H}$, $(h(u_1), \ldots, h(u_k))$ is $\delta$-close to the uniform distribution on $[N]^k$. We say the hash family is explicit if the elements of $\mathcal{H}$ can be sampled efficiently.

Using $\epsilon$-biased distributions ([19]) one can design efficient hash families as above of small size.

▶ **Theorem 11** ([19]). *For all $U, N$, there exists an explicit $\delta$-almost $k$-wise independent hash family $\mathcal{H} = \{h : [U] \to [N]\}$ with $\log(|\mathcal{H}|) = O((\log\log U) + k\log N + \log(1/\delta))$. Further, for a given input, the output of any function in the family can be evaluated in $O(k\log N + \log(1/\delta))$ word operations in the unit cost RAM model.*

## 3.3 Known Extractors

Our constructions rely on some previous constructions of extractors which we review next.

The following constructions of Ta-Shma, Umans and Zuckerman [26] and Srinivasan and Zuckerman [25] give extractors with nearly optimal entropy losses but sub-optimal seed-lengths.

▶ **Theorem 12** ([26]). *For every $n, k$, and constant $\epsilon$, there exist explicit $(k, \epsilon)$-extractors $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log^2 k(\log\log k)^2)$ and $m = k + d - O(1)$.*

▶ **Theorem 13** ([25]). *For every $n, k, \epsilon$, there exists explicit $(k, \epsilon)$-strong extractors $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(k + \log n + \log(1/\epsilon))$ and $m = k - O(\log(1/\epsilon))$.*

The following theorem gives a way to convert explicit extractors to explicit strong extractors.

▶ **Theorem 14** ([24]). *Any explicit $(k, \epsilon)$-extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ can be transformed into an explicit $(k, O(\sqrt{\epsilon}))$-strong extractor $E' : \{0,1\}^n \times \{0,1\}^{d+d'} \to \{0,1\}^{m-(d+L+1)}$ where $d' = \operatorname{poly}\log(d/\epsilon)$ and $L = 2\log(1/\epsilon) + O(1)$.*

Applying Theorem 14 to Theorem 12, we get the following corollary.

▶ **Corollary 15.** *For every $n, k$ and constant $\epsilon$, there exist explicit $(k, \epsilon)$-strong extractors $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log^2 k(\log\log k)^2)$ and $m = k - O(1)$.*

We shall also need the well-studied notion of block-wise sources [8].

▶ **Definition 16** (block-wise source). Two (possibly correlated) distributions $X_1, X_2$ form a $(k_1, k_2)$ block-wise source if $H_\infty(X_1) \geq k_1$ and for every $x_1$, $H_\infty(X_2 | X_1 = x_1) \geq k_2$. $X$ is called a $(k_1, k_2)$ block-wise source if $X = X_1 \circ X_2$ where $X_1, X_2$ form a $(k_1, k_2)$ block-wise source.

▶ **Definition 17** (subsource). A distribution $X'$ over domain $\{0,1\}^n$ is a subsource of a distribution $X$ (over the same domain $\{0,1\}^n$) if there exists an event $A \subset \{0,1\}^n$ such that $X'$ is the probability distribution obtained by conditioning on $X$ being in $A$.

The following simple lemma says that any source has a subsource which is a block-wise source with high min-entropy.

▶ **Lemma 18.** *Let $X$ be a flat distribution over $\{0,1\}^n$ with $H_\infty(X) \geq k$. For every $k_0 < k$, there exists a subsource $X' = X_1 \circ X_2$ of $X$ which is a $(k - k_0 - \log n - 3, k_0)$ block-wise source.*

**Proof.** Let $X$ be uniformly random over $A \subseteq \{0,1\}^n$ where $|A| \geq 2^k$. For each $0 \leq i \leq n$ and each $u \in \{0,1\}^i$, let $A_u$ be the set of elements in $A$ whose $i$-prefix is $u$. Let

$$S = \{(i, u) : 1 \leq i \leq n, u \in \{0,1\}^i, \text{ and } 2^{k_0} \leq |A_u| < 2^{k_0+1} \leq |A_{u_{1,\ldots,(i-1)}}|\},$$

where $u_{1,\ldots,(i-1)}$ is the $(i-1)$-prefix of $u$. Also let

$$S_i = \{u : (i, u) \in S\}$$

for each $1 \leq i \leq n$. One can show that $\sum_{(i,u) \in S} |A_u| \geq |A|/2 \geq 2^{k-1}$. Therefore $|S| \geq 2^{k-k_0-2}$. Thus there exists an $i$ such that $|S_i| \geq 2^{k-k_0-\log n-2}$, and by letting $A = \cup_{u \in S_i} A_u$, $X' = (X|A)$ is the desired block-wise subsource. ◀

## 4 Strong Disperser for High Min-Entropy Sources

We now prove Theorem 5. Our construction builds on the work of Gradwohl et al. [13] and amplifies their construction by using a part of the source as a seed. As remarked in the introduction such approaches often do not work or at the very least are quite subtle. We first briefly review the main constructions of Gradwohl et al.

The starting point for the work of Gradwohl et al. is the classical expander walk theme for constructing pseudorandom objects as used for example in [1], [12]. However, they introduced a twist of the standard expander walk – the *reversed walk*. Let $Dsp : \{0,1\}^n \times \{0,1\}^d \times \{0,1\}^{\log t} \to \{0,1\}^m$ be the disperser we are aiming for. Let $G$ be an expander graph with degree $D$ on $\{0,1\}^n$ and constant spectral gap. Let us also associate the *seeds* $\{0,1\}^d \times \{0,1\}^{\log t}$ with tuples $([D]^t, \ell)$ for $\ell \in [t]$ and suitable parameters $D, t$. The disperser of [13] takes a $\ell$-step walk on the expander with edge-labels as given by the seed, but in the *reverse order*: for $x \in \{0,1\}^n$, $y = (e_1, \ldots, e_t) \in [D]^t$, $\ell \in [t]$, $Dsp(x, y) =$ vertex reached by taking the walk $(e_\ell, e_{\ell-1}, \ldots, e_1)$ in $G$ starting at $x$.

The idea of taking the walk in reverse order in comparison to the works of [1], [12], which may seem to be of little significance at first look is in fact important for the (simple) analysis of [13]. The final analysis is based on an application of Theorem 9. Unfortunately, the disperser obtained by this approach is not strong and we need some new ideas to make it strong.

Observe that in the above construction, by looking at the $\ell$-step random walk for all $\ell \in [t]$, we indirectly allow ourselves to look at a *random* intermediate point of a $t$-step random walk. This enables [13] to decrease the error probability even further (compared to standard expander walk constructions) to the desired bound. However, doing so costs at least $\log t$ additional random bits which is too much of a loss for us. We instead use a part of the source string itself, say the last $\log t$ bits, to get these additional $\log t$ bits. Note that this introduces problematic (probabilistic) dependencies between the starting point of the walk and the number of steps of the walk. We argue that the expander walk construction is robust enough to handle such dependencies to get our final construction for sources with $O(1)$ entropy deficiency ($(n-k)$).

**Proof of Theorem 5.** Our simple construction and analysis follow the above sketch closely. We first set up some parameters. Let $t = 2^{2c+3}\log(1/\epsilon)$, $m = n - \log t = n - \log\log(1/\epsilon) - 2c - 3$. Let $D$ be a sufficiently large constant to be chosen later and $G = (\{0,1\}^m, E)$ be an explicit $(D, 1/2)$-expander. Finally, let $d = t \log D$.

Given input $(X, Y) \in \{0,1\}^n \times \{0,1\}^d$, we split $X$ as $(X_0 \circ i)$ where $X_0 \in \{0,1\}^m$, and $i \in \{0,1\}^{\log t}$ which we will view as an integer in $[t]$. We also view the seed $Y$ as $Y \equiv (e_1, \ldots, e_t) \in [D]^t$ in lieu of performing a random walk on $G$.

Define $Dsp(X, Y)$ as follows. For $\ell \in [t]$, let $X_\ell \in \{0,1\}^m$ be the vertex in $G$ reached by traversing the edges $(e_\ell, e_{\ell-1}, \ldots, e_1)$ (in that order) from the vertex $X_0$. Then,

$$Dsp((X_0, i), (e_1, \ldots, e_t)) = X_i.$$

We next argue that $Dsp$ as defined above is a $(n-c, \epsilon)$-strong disperser. To see this, fix a set $S \subseteq \{0,1\}^n$ with $|S| \geq 2^{n-c}$. For $y \in \{0,1\}^d$ and $z \in \{0,1\}^m$, call $(y, z)$ *bad* if $z \notin Dsp(S, y)$, i.e., there is no $x \in S$ such that $Dsp(x, y) = z$. We will show that the fraction of bad pairs $(y, z) \in \{0,1\}^d \times \{0,1\}^m$ is at most $\epsilon$.

▶ **Claim 19.** *We have* $|\{(y, z) \in \{0,1\}^d \times \{0,1\}^m : (y, z) \textit{ bad}| \leq \epsilon \cdot 2^d \cdot 2^m$.

**Proof.** Recall that $n = m + \log t$. For each $i \in [t]$, let $S_i \subseteq \{0,1\}^m$ be the first $m$ bits of the strings in $S$ whose last $\log t$ bits equal $i$:

$$S_i = \{v \in \{0,1\}^m : v \circ i \in S\}.$$

Note that, $\mu := (\sum_i |S_i|)/t2^m \geq |S|/t2^m \geq 2^{-c}$.

Let $(y, z) \in \{0,1\}^d \times \{0,1\}^m$ be bad, where $y = (e_1, \ldots, e_t)$. Then, by the definition of $Dsp$, it means that the $i$-step walk $(e_i, e_{i-1}, \ldots, e_1)$ starting from any element of $v \in S_i$ does not end at the vertex $z \in \{0,1\}^m$. This is equivalent to saying that the $i$-step walk

$(e_1,\ldots,e_i)$ starting at $z$ does not end at an element of $S_i$ for all $i$. As for $z \in_u \{0,1\}^m$, $y \in_u \{0,1\}^d$ the above corresponds to a $t$-step random walk in $G$, by applying Theorem 9 to the walk, we get

$$\begin{aligned}\Pr[(y,z)\text{ bad}] &= \Pr[\forall i,\ \text{Starting at } z,\ Walk(e_1,\ldots,e_i) \text{ does not land in } S_i] \\ &\le \exp\left(-\mu^2 t/8\right) \\ &\le \exp\left(-2^{-2c}t/8\right) = \exp(-\log(1/\epsilon)) \le \epsilon.\end{aligned}$$

◀

To prove the theorem, let $X \subseteq \{0,1\}^n$ be a $(n-c)$ min-entropy source. Then, $|\mathrm{Supp}(X)| \ge 2^{n-c}$. And, by applying the above arguments to $S = \mathrm{Supp}(X)$, we get

$$|\mathrm{Supp}((Dsp(X,U_d),U_d))| = \{(y,z) : z \in Dsp(S,y)\} \ge (1-\epsilon)2^{m+d}.$$

◀

## 5 Covering Hash Families

We now prove Theorem 2. Recall that the goal is to hash from a universe $U$ to $[N]$ so as to satisfy $\epsilon$-covering property. As described in the introduction, the construction proceeds in two steps. For the first step, we shall use the following simple property of almost $O(1)$-wise independent hash functions.

▶ **Lemma 20.** *For any integer $d > 1$, let $\mathcal{H} = \{h : U \to [N]\}$ be a family of $N^{-2d}$-almost $2d$-wise independent hash functions. Then for any $S \subseteq U$ such that $|S| \ge N$, we have,*

$$\Pr_{h\in\mathcal{H}}[|h(S)| \le N/4] \le O_d\left(\frac{1}{N^{d-1}}\right).$$

**Proof.** For any set $S \subseteq U$ of cardinality $N$, let the random variable $T$ be the number of distinct pair $(i,j)$'s such that $i,j \in S$ and $h(i) = h(j)$, where $h$ is a random element from $\mathcal{H}$. One can show that

$$\mathbf{E}[T^d] \le O_d(N).$$

Therefore

$$\Pr[T \ge N] \le \mathbf{E}\left[\frac{T^d}{N^d}\right] = O_d\left(\frac{1}{N^{d-1}}\right).$$

The lemma follows by the fact that when $|h(S)| < N/4$, the $T$ value for $h$ is at least $N$. ◀

We are now ready to construct the family of hash functions in Theorem 2, proving our main result.

**Proof of Theorem 2.** We start with the construction of the set of hash functions. We will only work with $N = 2^n$ for some integer $n > 0$. It is easy to extend the construction to general $N$'s. The construction proceeds in two stages.

Let $Dsp : \{0,1\}^{n+\log n+\alpha} \times \{0,1\}^d \to \{0,1\}^n$ be a $(n+\log n+\alpha-2, 1/2^{(\lceil c+2\rceil)(n+1)})$-strong disperser as constructed in Theorem 5. We know that $\alpha = O(1)$ and $d = O(n)$. Further, as the disperser takes an $O(n)$ walk on an expander, it can be computed with $O(n) = O(\log N)$ word operations.

Let $\mathcal{G} : \{g : U \to [2^\alpha N \log N]\}$ be a family of $(2^\alpha N \log N)^{-2\lceil c+2\rceil}$-almost $2\lceil c+2\rceil$-wise independent functions constructed in Theorem 11. A random hash function $h : U \to [N] \in \mathcal{H}$ is defined as follows. Pick a random $g \in \mathcal{G}$, and pick a random $r \in \{0,1\}^d$. Let

$$h(x) = Dsp(g(x), r).$$

By Theorem 11, any $g \in \mathcal{G}$ can be described with $O((\log\log U) + c\log N)$ bits and computed with $O(\log N)$ word operations. Therefore, we have $|\mathcal{H}| \le ((\log U) \cdot N)^{O(1)}$ and each function in $\mathcal{H}$ can be computed with $O(\log N)$ word operations.

Now we will prove that for any $S \subseteq U$ such that $|S| \ge 2^\alpha N \log N$, the probability (over a random $h \in \mathcal{H}$) that $h(S) \neq [N]$ is at most $N^{-c}$. By Lemma 20, the probability that $|g(S)| < 2^{\alpha-2} N \log N$ is at most $O(N^{-c-1}) < N^{-c}/2$ (for sufficiently large $N$). When $|g(S)| \ge 2^{\alpha-2} N \log N$, by the definition of $Dsp$, with probability at least $(1 - 1/2^{\lceil c\rceil(n+1)}) \ge 1 - N^{-c}/2$ over the random choice of $r$, we have $|Dsp(g(S), r)| \ge (1 - 1/2^{n+1})2^n$, i.e. $Dsp(g(S), r) = \{0,1\}^n$. By applying a union bound over these two events, we prove the desired statement. ◀

## 6 Strong Disperser for General Sources

In this section we will prove Theorem 6. We first construct a strong disperser for *block-wise* sources and reduce the general case to that of block-wise sources. The high-level idea is as follows: we first apply an extractor on the second source $X_2$ to get a short string, which we then use as a seed for applying a strong extractor to the first source $X_1$. This approach has been used in many other works and is in fact central to many constructions of extractors and dispersers, see [22] for a recent example. We present it here for completeness.

▶ **Lemma 21.** *For all $n, i, k_1 \le i, k_2 \le n - i$, let $X_1 \circ X_2 \in \{0,1\}^i \times \{0,1\}^{n-i}$ be a $(k_1, k_2)$ block-wise source. Suppose that $\log^3 n \le k_2 \le \exp(\log^{1/3} n)$. Then, for all constant $\epsilon > 0$, there exists an explicit strong disperser $Dsp : \{0,1\}^i \times \{0,1\}^{n-i} \times \{0,1\}^d \to \{0,1\}^m$ with error $\epsilon$, where $d = O(\log n)$ and the entropy loss $k_1 + k_2 - m = O(1)$.*

**Proof.** Let $\delta = \epsilon/2$. Let $E_1 : \{0,1\}^{n-i} \times \{0,1\}^d \to \{0,1\}^{d'}$ be a $(k_2, \delta)$-strong extractor as in Corollary 15, where $d = O(\log n + \log^2 k_2 (\log\log k_2)^2) = O(\log n)$ and $d' = k_2 - O(1)$. Since $\log^3 n \le k_2 \le \exp(\log^{1/3} n)$, $d' = \Omega(\log^3 n)$. Let $E_2 : \{0,1\}^i \times \{0,1\}^{d'} \to \{0,1\}^{m'}$ be a $(k_1, \delta)$-strong extractor in Corollary 15, where $m' = k_1 - O(1)$. This is possible since $d' = \Omega(\log^3 n)$ has more bits than required for Corollary 15 ($O(\log n + \log^2 k_1 (\log\log k_2)^2)$).

Now, we construct our disperser $Dsp : \{0,1\}^i \times \{0,1\}^{n-i} \times \{0,1\}^d \to \{0,1\}^m$, where $m = m' + d' = k_1 + k_2 - O(1)$:

$$Dsp(x_1, x_2, r) = E_2\left(x_1, E_1(x_2, r)\right) \circ E_1(x_2, r).$$

Now we show that

$$|\mathrm{Supp}\left(Dsp(X_1, X_2, U_d) \circ U_d\right)| \ge (1 - 2\delta)2^{m+d},$$

and therefore $Dsp$ is a strong disperser with error at most $2\delta = \epsilon$.

Fix an $x_1$ from the distribution $X_1$. Let $Y = E_1(X_2|X_1 = x_1, r)$ be a distribution, where $r$ is uniformly chosen from $U_d$. By the definition of block-wise source, we know that $H_\infty(X_2|X_1 = x_1) \ge k_2$. Therefore, by the definition of strong extractors, $(Y, r)$ is $\delta$-close to $U_{d'+d}$. In particular,

$$\Delta[(E_2(x_1, Y), Y, r), (E_2(x_1, U_{d'}), U_{d'}, U_d)] \le \delta$$

holds for every $x_1$, where the two copies of $U_{d'}$ denote the same random variable. Therefore, by taking the (weighted) average over $x_1$, we have

$$\Delta[(E_2(X_1, Y), Y, r), (E_2(X_1, U_{d'}), U_{d'}, U_d)] \leq \delta,$$

where the two copies of $U_{d'}$ denote the same random variable. By the definition of $E_2$ we have

$$\Delta[(E_2(X_1, U_{d'}), U_{d'}, U_d), (U_{m'+d'}, U_d)] \leq \delta,$$

where the two copies of $U_{d'}$ denote the same random variable. Therefore,

$$\Delta[(E_2(X_1, Y), Y, r), U_{m'+d'+d}] \leq 2\delta.$$

Our claim is proved by observing that $Dsp(X_1, X_2, r) = (E_2(X_1, Y), Y)$, $m = m' + d'$, and Fact 7.

◀

## 6.1 Proof of Theorem 6

Now we are ready to prove Theorem 6.

**Proof of Theorem 6.** We will assume that the min-entropy $k$ is at least $\log^4 n$, otherwise the extractor defined in Corollary 15 is good enough to be our disperser.

Let $E : \{0,1\}^n \times \{0,1\}^{O(\log n)} \to [n]$ be an $(\Omega(\log n), \delta/(2n))$-strong extractor as in Theorem 13. For $i \leq n$, let $D_i : \{0,1\}^i \times \{0,1\}^{n-i} \times \{0,1\}^{d'} \to \{0,1\}^m$ be a disperser described in Lemma 21 with $k_1 = k - \log^3 n - 3\log n - 5, k_2 = \log^3 n$. Observe that $d' = O(\log n)$ and $m = k_1 + k_2 - O(1) = k - 3\log n - O(1)$.

Now we define the disperser $Dsp$ as follows. Given input $(x, r) \in \{0,1\}^n \times \{0,1\}^d$, we break $r$ into $r_1 \circ r_2$ where each of $r_1$ and $r_2$ has $\Omega(\log n)$ bits. Let $i = E(x, r_1)$. Let $Dsp(x, r) = D_i(x, r_2)$.

We now prove that $Dsp$ is the desired disperser. Without loss of generality, we can assume that the source is a flat distribution, i.e., assume that $X$ is uniform on $A \subseteq \{0,1\}^n$ where $|A| \geq 2^k$.

By the definition of $E$, with probability at least $(1 - \delta)$ over the random choice of $r_1$, the distribution $E(X, r_1)$ is $1/(2n)$-close to the uniform distribution over $[n]$. We fix such an $r_1$ in the following analysis.

By Lemma 18, there exists an $i_0$ such that there exists $X_1 \circ X_2$ being a $(k - \log^3 n - 2\log n - 4, \log^3 n + \log n + 1)$ subsource of $X$, where $X_1$ has the first $i_0$ bits of the string and $X_2$ has the remaining $(n - i_0)$ bits. Now, as $i = E(x, r_1)$ is $(1/2n)$-close to being uniform on $[n]$, $\Pr[i = i_0] \geq 1/(2n)$. Let $\tilde{X_1} \circ \tilde{X_2}$ be the random variable obtained by conditioning $X_1 \circ X_2$ on the event $i = i_0$. Then, each of $\tilde{X_1}$, $\tilde{X_2}$ have at most $\log n$ bits less entropy compared to $X_1$, $X_2$ (respectively). Therefore, from the above observations and our choice of parameters we get that $\tilde{X_1} \circ \tilde{X_2}$ is a $(k_1, k_2)$ block-wise source.

By Lemma 21, we know that

$$|\mathrm{Supp}(D_{i_0}(\tilde{X_1}, \tilde{X_2}, r_2), r_2)| \geq (1 - \delta)2^{m+|r_2|},$$

therefore

$$|\mathrm{Supp}(D_i(X_1, X_2, r_2), r_2)| \geq (1 - \delta)2^{m+|r_2|}.$$

Since $X_1 \circ X_2$ is a subsource of $X$, we have

$$|\mathrm{Supp}(D_i(X, r_2), r_2)| \geq (1 - \delta)2^{m+|r_2|}.$$

To summarize, we have proved that with probability at least $(1 - \delta)$ over the random choice of $r_1$,

$$|\mathrm{Supp}(D_i(X, r_2), r_2)| \geq (1 - \delta)2^{m+|r_2|},$$

which implies that

$$|\mathrm{Supp}(D_i(X, r_2), r_1, r_2)| \geq (1 - \delta)2^{m+|r_1|+|r_2|}.$$

Our claim is proved by observing that $Dsp(X, r_1, r_2) = D_i(X, r_2)$. ◀

## References

1 Miklós Ajtai, János Komlós, and Endre Szemerédi. Deterministic simulation in LOGSPACE. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 132–140. ACM, 1987.

2 Noga Alon, Martin Dietzfelbinger, Peter Bro Miltersen, Erez Petrank, and Gábor Tardos. Linear hash functions. *J. ACM*, 46(5):667–683, 1999.

3 Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating independence: new constructions of condensers, ramsey graphs, dispersers, and extractors. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing*, pages 1–10, 2005.

4 Boaz Barak, Anup Rao, Ronen Shaltiel, and Avi Wigderson. 2-source dispersers for sub-polynomial entropy and ramsey graphs beating the frankl-wilson construction. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 671–680, 2006.

5 Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. In *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science*, pages 40–47, 2010.

6 J Lawrence Carter and Mark N Wegman. Universal classes of hash functions. *Journal of computer and system sciences*, 18(2):143–154, 1979.

7 L Elisa Celis, Omer Reingold, Gil Segev, and Udi Wieder. Balls and bins: Smaller hash families and faster evaluation. In *Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science*, pages 599–608. IEEE, 2011.

8 Benny Chor and Oded Goldreich. On the power of two-point based sampling. *J. Complexity*, 5(1):96–106, 1989.

9 Yevgeniy Dodis, Amit Sahai, and Adam Smith. On perfect and adaptive security in exposure-resilient cryptography. In *EUROCRYPT*, pages 301–324, 2001.

10 Zeev Dvir, Swastik Kopparty, Shubhangi Saraf, and Madhu Sudan. Extensions to the method of multiplicities, with applications to kakeya sets and mergers. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 181–190, 2009.

11 Ariel Gabizon, Ran Raz, and Ronen Shaltiel. Deterministic extractors for bit-fixing sources by obtaining an independent seed. *SIAM J. Comput.*, 36(4):1072–1094, 2006.

12 Oded Goldreich, Russell Impagliazzo, Leonid A. Levin, Ramarathnam Venkatesan, and David Zuckerman. Security preserving amplification of hardness. In *Proceedings of the 31th Annual IEEE Symposium on Foundations of Computer Science*, pages 318–326. IEEE, 1990.

**13** Ronen Gradwohl, Guy Kindler, Omer Reingold, and Amnon Ta-Shma. On the error parameter of dispersers. *Approximation, Randomization and Combinatorial Optimization: Algorithms and Techniques*, pages 294–305, 2005.

**14** Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from parvaresh–vardy codes. *Journal of the ACM (JACM)*, 56(4):20, 2009.

**15** Alexander D Healy. Randomness-efficient sampling within NC1. *Computational Complexity*, 17(1):3–37, 2008.

**16** Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bull. of the American Mathematical Society*, 43(4):439–561, 2006.

**17** Jesse Kamp and David Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM J. Comput.*, 36(5):1231–1247, 2007.

**18** A Lubotzky, R Phillips, and P Sarnak. Explicit expanders and the ramanujan conjectures. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 240–246. ACM, 1986.

**19** Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

**20** Noam Nisan and David Zuckerman. Randomness is linear in space. *J. Comput. Syst. Sci.*, 52(1):43–52, 1996.

**21** Jaikumar Radhakrishnan and Amnon Ta-Shma. Tight bounds for depth-two superconcentrators. In *Proceedings of the 38th Annual IEEE Symposium on Foundations of Computer Science*, pages 585–594. IEEE, 1997.

**22** Anup Rao. Extractors for a constant number of polynomially small min-entropy independent sources. *SIAM Journal on Computing*, 39(1):168–194, 2009.

**23** Ran Raz, Omer Reingold, and Salil P. Vadhan. Error reduction for extractors. In *Proceedings of the 40th Annual IEEE Symposium on Foundations of Computer Science*, pages 191–201, 1999.

**24** Omer Reingold, Ronen Shaltiel, and Avi Wigderson. Extracting randomness via repeated condensing. *SIAM Journal on Computing*, 35(5):1185–1209, 2006.

**25** Aravind Srinivasan and David Zuckerman. Computing with very weak random sources. *SIAM Journal on Computing*, 28(4):1433–1459, 1999.

**26** Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Loss-less condensers, unbalanced expanders, and extractors. In *Proceedings of the 33th Annual ACM Symposium on Theory of Computing*, pages 143–152. ACM, 2001.

**27** Amnon Ta-Shma and David Zuckerman. Extractor codes. In *Proceedings of the 33th Annual ACM Symposium on Theory of Computing*, pages 193–199, 2001.

**28** Salil Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 2011.