Storing information with extractors

Amnon Ta-Shma

Computer Science Department, Tel-Aviv University, Tel-Aviv, Israel 69978

Abstract

We deal with the problem of storing a set of K elements that are taken from a large universe of size N, such that membership in the set can be determined with high probability by looking at just one bit of the representation. Buhrman et al. show an *explicit* construction with about $K^2 \log N$ storing bits. We show an explicit construction with about $K^{1+o(1)}$ storing bits, that gets closer to the optimal $K \log N$ bound.

Our technique is of independent interest. Buhrman et al. show a *non-explicit* optimal (up to constant factors) construction that is based on the existence of certain good unbalanced expanders. To make the construction explicit one needs to be able to explicitly 'encode' and 'decode' such expanding graphs. We generalize the notion of loss-less condensers of Ta-Shma et al. [Proc. 33rd Annual ACM Symposium on Theory of Computing, 2001, pp. 143–152] and build such graphs. We further show how to efficiently decode such graphs using an observation from Ta-Shma and Zuckerman [Manuscript, 2001] about Trevisan's extractor. © 2002 Elsevier Science B.V. All rights reserved.

Keywords: Data structures; Expanders; Extractors; Condensers; List decoding

1. Introduction

One immediate way of storing a set of K elements is writing down a sorted list of the K elements in the set. Thus a size K set is represented by $K \log N$ bits, which is close to optimal. Given an element $x \in [N]$, checking whether x belongs to the set requires reading $\log K \cdot \log N$ bits from the representation. Yao [15] asked whether we can do better?

The problem attracted a lot of attention. Later research proceeded in two paths. One path studied the *cell probe* model suggested by Yao in [15], where one can only store and read elements from the universe $[N]$. Up until recently most of the work done was

in that model. Fredman, Komlos and Szemeredi [3] showed how to solve the problem in this model using only a constant number of queries and linear in the optimal space. Later the space complexity was further improved [4,1,7]. Yet, obviously any algorithm in this model has to query at least one element of the universe, hence queries at least $\log N$ bits.

Minsky and Papert [6] first studied the *bit probe* model where information can be stored in an arbitrary way. Recently, Buhrman et al. showed [2] that non-explicitly there are schemes which store sets of size K using $O((K \log N)/\varepsilon)$ bits, and determine membership with at most ε error while reading only a **single** bit of the representation. We first explain the error model. An $\binom{N}{K} \to_\varepsilon p([M])$ storing scheme consists of:

E-mail address: amnon@post.tau.ac.il (A. Ta-Shma).

Table 1
$\binom{N}{K}$ storing schemes

No. of bits probed	Space complexity	Reference
O($\log N$)	O($K \log N$)	[3], explicit
1	O($\frac{K^2 \log N}{\varepsilon^2}$)	[2], explicit
1	$K \cdot 2^{\mathrm{O}((\log \frac{\log N}{\varepsilon})^3)}$	This paper, explicit
1	About $K \frac{\log N}{\varepsilon}$	Lower bound and non-explicit [2]

- A deterministic storing function $f : \binom{[N]}{K} \to p([M])$ that maps a set $\mathcal{A} \subseteq [N]$ with K elements to a subset of $[M]$. f is efficient if it can be computed in time polynomial in K and $\log N$.
- A *probabilistic* membership function $g : p([M]) \times [N] \to \{0, 1\}$ that gets the output of the storing scheme, and an input $x \in [N]$, and returns a Boolean value. g is explicit if it can be computed in time polynomial in $\log N$. (Notice that its input length is much bigger, $M + \log N$.)

It should hold that for every $\mathcal{A} \in \binom{[N]}{K}$ the encoding $f(\mathcal{A})$ is such that:

$$\forall x \in \mathcal{A} \quad \Pr\big[g(f(\mathcal{A}), x) = 1\big] \geqslant 1 - \varepsilon, \quad \text{and}$$

$$\forall x \notin \mathcal{A} \quad \Pr\big[g(f(\mathcal{A}), x) = 1\big] \leqslant \varepsilon.$$

The storing space of the scheme is M. The storing scheme is one probe if for every $\mathcal{A}$ and x the probabilistic membership function g probes a single bit of $f(\mathcal{A})$.

The Buhrman et al. result shows that with optimal storing space (only a constant factor times the best possible) one can determine membership by looking at just one bit of the representation. This nice result is, however, mainly a proof of feasibility rather than a construction as the storing and membership functions are non-explicit and might take space exponential in N to be expressed. Constructively, using designs, Buhrman et al. [2] show to build efficient one-probe storing schemes with $M = \mathrm{O}((K^2 \log N)/\varepsilon^2)$.

Our main result is the construction of an explicit one-probe storing scheme with $K \cdot 2^{\mathrm{O}((\log(\log N)/\varepsilon)^3)}$ space complexity. We summarize these results in Table 1.

Our bound is better than that of [2] as long as

$$\log K = k \geqslant \Omega\big(\log \log N + \log \tfrac{1}{\varepsilon}\big)^3.$$

Table 2
Explicit $\binom{N}{K}$ one-probe storing schemes, $K = 2^{\sqrt{n}}$, $\varepsilon = 1/\mathrm{poly}(n)$

Space complexity	Reference
$K^{2+o(1)}$	[2]
$K^{1+o(1)}$	This paper
About $K \frac{\log N}{\varepsilon}$	Lower bound

To demonstrate the improvement we compare in Table 2 the space complexity of $\binom{N}{K} \to_\varepsilon p([M])$ one probe storing schemes when $N = 2^n$, $\varepsilon = 1/\mathrm{poly}(n)$ and, say, $K = 2^{\sqrt{n}}$.

1.1. The Buhrman et al. technique

Buhrman et al. prove that one can get a one-probe storing scheme by using good, highly unbalanced and highly expanding bipartite graphs. We start with an intuitive explanation of their result. The construction uses a bipartite graph $G(V_1 = [N = 2^n], V_2 = [M = 2^m], E)$ with left degree $D = 2^d$. We identify elements of $\{0, 1\}^n$ with vertices of V_1. We also identify set representations (that are M bit long) with zero/one assignments to all the M vertices of V_2. The storing function takes a subset $\mathcal{A} \subset V_1$ and represents it as a zero/one assignment to the M vertices of V_2 (therefore taking M bits of space). The assignment should have the property that for every $x \in \mathcal{A}$ most of its graph neighbors have assignment '1', while for every $x \notin \mathcal{A}$ most of its graph neighbors have assignment '0'. The probabilistic membership function g then gets $x \in V_1$, picks $i \in [1..D]$ uniformly at random and checks whether the ith neighbor of x in the graph G is set to 0 or 1. Buhrman et al. show that if the graph is expanding such an assignment exists.

We now formalize these notions. We can view a function $F : \{0, 1\}^n \times \{0, 1\}^d \to \{0, 1\}^m$ as a bipartite graph $G_F(V_1 = [N = 2^n], V_2 = [M = 2^m], E)$ where

each v_1 in V_1 has degree $D = 2^d$, and v_2 is the ith neighbor of v_1 iff $F(v_1, i) = v_2$. For $v_1 \in V$ we denote by $\Gamma(v_1)$ the set of V_2 vertices that are connected to v_1. For a set $\mathcal{A} \subset V_2$, the density of $\mathcal{A}$ is $\rho(\mathcal{A}) = |\mathcal{A}|/|V_2|$.

We say F has explicit *encoding* if there is an algorithm that given $v_1 \in V_1$ and $i \in [D]$, computes $F(v_1, i)$ in time polynomial in the input length. A bipartite graph $G = (V_1, V_2, E)$ is (K, c) expanding if for every $\mathcal{A} \subseteq V_1$ of cardinality at most K, $|\Gamma(\mathcal{A})| \geqslant c|\mathcal{A}|$, where $\Gamma(\mathcal{A})$ is the set of neighbors of $\mathcal{A}$.

We say F (and G_F) has efficient δ *decoding* if there is a deterministic algorithm that given a set $\mathcal{A} \subset V_2$ finds all elements $x \in V_1$ for which $|\Gamma(x) \cap \mathcal{A}| \geqslant (\rho(\mathcal{A}) + \delta)D$ and whose running time is at most $\mathrm{poly}(D, |V_2|, 1/\delta)$.

Buhrman et al. show:

Theorem 1 [2]. *If for $K = K(N) < N$ and $\varepsilon = \varepsilon(N)$ there exists a family of graphs $\{G_N = (V_1 = [N], V_2 = [M], E)\}$ of some degree $D = D(N)$ that is $(2K, (1 - \varepsilon)D)$ expanding, then there exist $\binom{N}{K} \xrightarrow{}_{4\varepsilon} p([M])$ one-probe storing schemes. Furthermore,*

- *If the family of expanding graphs has explicit encoding then the storing scheme has an efficient probabilistic membership function.*
- *If $M = \mathrm{poly}(K)$ and the family of expanding graphs has efficient 2ε decoding then the storing scheme has an efficient storing function.*

Our contribution is the construction of good expanders that have explicit encoding and efficient decoding.

1.2. Our technique

We want to build a $(2K, (1 - \varepsilon)D)$ expanding graph $G = (V_1, V_2, E)$ of degree D. This means that if for every $v_1 \in V_1$ we color all the edges leaving v_1 with the colors $1..D$ (in an arbitrary way), then for every subset $A \subseteq V_1$ of cardinality at most K, for almost all colors i, the ith color maps A to V_2 almost one-to-one. This leads to our definition of an $(n, k) \xrightarrow{1:1}_{\varepsilon} m$ strong condenser (see below). A useful generalization of this is that of an $(n, k) \xrightarrow{P:1}_{\varepsilon} m$ strong condenser, where the mapping is $P : 1$ rather than $1 : 1$. Given an

appropriate strong condenser $F : \{0, 1\}^n \times \{0, 1\}^d \to \{0, 1\}^m$ the desired graph is then G_F as defined in Section 1.1.

The strong condenser we want to build should have the following properties:

- it should have explicit encoding,
- it should have efficient decoding, and
- V_2 should be small (or more precisely m should be $k + o(k)$).

We show that one can build such an entity by concatenating many little strong condensers used as building blocks. We show that if each has explicit encoding so does the concatenation, and if each has efficient decoding so does the concatenation (well, here the precise statement is a bit more careful).

We observe that strong universal extractors (defined below) give good $P : 1$ strong condensers, where P depends on the entropy loss. This builds on ideas from [9,11]. Furthermore, by repeatedly concatenating extractors we can make the entropy loss, and P, order of magnitudes smaller then the original problem. Repeating that enough times we make P so small that by using very simple objects we can make the whole construction $1 : 1$. This is similar in spirit to ideas in [14, 8]. Doing this carefully we get m small enough, and we also get explicit encoding. For efficient decoding we choose the strong extractor to be Trevisan's extractor [10]. Raz et al. [9] showed that Trevisan's extractor is strong and universal, and Ta-Shma and Zuckerman [12] showed that Trevisan's extractor has efficient decoding. We then get:

Theorem 2. *For every $K = K(N) < N$ and $\varepsilon = \varepsilon(N)$ there exists a family of graphs $\{G_N = (V_1 = [N = 2^n], V_2 = [M], E)\}$ of degree $D = D(N)$, that is,*

- $(2K, (1 - \varepsilon)D)$ *expanding with*

  $M = K \cdot 2^{\mathrm{O}((\log(\log N/\varepsilon))^3)},$

- *has explicit encoding, and*
- *has efficient ε decoding.*

We note that a variant of Trevisan's extractor was shown to be a good expander in [11]. That construction, however, is not good enough for us as it has poor dependence on ε, specifically, M is at

least $2^{1/\varepsilon}$. We also note that the construction here is somewhat atypical in that it does not care how large the degree D is (as long as M is relatively small).

By Theorem 1 this translates to an explicit one-probe scheme:

Theorem 3. *There exists an* $\binom{N}{K} \to_{4\varepsilon} p([M])$ *explicit one-probe storing schemes, with ε error and $M = K \cdot 2^{O((\log(\log N/\varepsilon))^3)}$ space.*

2. Strong condensers

Definition 1. A function $F : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is an $(n,k) \xrightarrow{P:1}_{\varepsilon} m$ strong condenser if in the graph $G_F = (V_1 = [N = 2^n], V_2 = [M = 2^m], E)$

- there are no parallel edges (i.e., for every $x \in V_1$ there are $D = 2^d$ distinct edges leaving x), and
- for every $A \subseteq V_1$ of cardinality at most $K = 2^k$, the induced graph $G_F(A) = (A, \Gamma(A), E)$ has a subgraph $E' \subseteq E$ with at least $(1 - \varepsilon)D|A|$ edges and right degree at most P.

In words, the second condition says that for every $A \subset V_1$ of cardinality at most 2^k we can ignore an ε fraction of the $D|A|$ edges leaving A and the remaining edges map at most P elements of A to the same element of V_2.

Definition 2 (Concatenation). Let $F_1 : \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_1}$ and $F_2 : \{0,1\}^n \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}$. Define $F_1 \circ F_2 : \{0,1\}^n \times \{0,1\}^{d_1+d_2} \to \{0,1\}^{m_1+m_2}$ by $F_1 \circ F_2(x; y_1, y_2) = F(x, y_1) \circ F(x, y_2)$.

Similarly we define concatenation $F_1 \circ \cdots \circ F_r$ for every $r \geqslant 1$. We first state the obvious fact that concatenating two strong condensers gives an even better strong condenser:

Lemma 1. *Suppose that F_1 is $(n,k) \xrightarrow{P_1:1}_{\varepsilon_1} m_1$ strong condenser and F_2 is $(n, \log P_1) \xrightarrow{P_2:1}_{\varepsilon_2} m_2$ strong condenser. Then $F_1 \circ F_2$ is $(n,k) \xrightarrow{P_2:1}_{\varepsilon_1+\varepsilon_2} m_1 + m_2$ strong condenser.*

Proof. Let $A \subset \{0,1\}^n$ be a subset of cardinality at most 2^k. Think of the graph $G_{F_1 \circ F_2}$. We can ignore ε_1 fraction of the $D_1|A|$ edges leaving A in G_{F_1} (where D_1 is the degree of G_{F_1}) and be left with a $P_1 : 1$ mapping, that is, every $w_1 \in \{0,1\}^{m_1}$ is reached by at most P_1 elements of A. Now, if we ignore ε_2 fraction of the edges of G_{F_2} leaving this set we are left with a $P_2 : 1$ mapping. $\square$

Now we want to show that given that the little blocks have explicit decoding then so does the concatenation.

Lemma 2. *Let $F_i : \{0,1\}^n \times \{0,1\}^{d_i} \to \{0,1\}^{m_i}$, and let the concatenation be $F = F_1 \circ \cdots \circ F_r$, $F : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, $d = d_1 + \cdots + d_r$, $m = m_1 + \cdots + m_r$. Assume $d \geqslant \log(n)$. Further assume:*

- *each F_i is an $(n, k_i) \xrightarrow{P_i:1}_{\varepsilon_i} m_i$ strong condenser,*
- *each F_i has explicit encoding, and*
- *each F_i has efficient δ decoding.*

Then F has efficient $r\delta$ decoding.

Proof. Given $\mathcal{A} \subset \{0,1\}^m$ our decoding algorithm is the following. For every $i = 1, \ldots, r$, for every possible value $z_j \in \{0,1\}^{m_j}$ for $j \neq i$, fix $i, z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_r$.

- Define $S_{i, z_j \ j \neq i} = \{s_i \mid (z_1, \ldots, z_{i-1}, s_i, z_{i+1}, \ldots, z_r) \in \mathcal{A}\}$.
- Let L be the output of the decoding algorithm for F_i with the set $S_{i, z_j \ j \neq i}$ and δ.
- Output $x \in L$ iff for at least $(\rho(\mathcal{A}) + \delta r)D$ values $y = y_1 \ldots y_d$, $F(x, y) = F_1(x, y_1) \circ \cdots \circ F_r(x, y_r) \in \mathcal{A}$.

We first bound the running time. We have at most $r \cdot 2^{m_1+\cdots+m_r} = \text{poly}(M)$ ways to fix i and z_j for $j \neq i$. For each possibility, $S_{i, z_j \ j \neq i} \subseteq \{0,1\}^{m_i}$ and decoding F_i takes time at most $\text{poly}(D2^{m_i}/\delta)$. In particular $|L| \leqslant \text{poly}(DM/\delta)$. Finally, for each element of L we compute its D neighbors in time $D \cdot \text{poly}(n) = \text{poly}(D)$. Altogether, the running time is bounded by $\text{poly}(MD/\delta)$.

We now prove correctness. Let $\mathcal{A} \subset \{0,1\}^m$ be a set of cardinality at most S, and let $x' \in \{0,1\}^n$ be an element that should be in the output, i.e.,

$$\left| \Gamma(x') \cap \mathcal{A} \right| \geqslant \left(\rho(\mathcal{A}) + \delta r \right)D.$$

We notice that the test T that given $w \in \{0, 1\}^m$ returns 1 iff $w \in \mathcal{A}$, accepts the uniform distribution with probability $\rho(\mathcal{A})$ while it accepts the distribution $F(x', [D])$ obtained by picking $y \in [D]$ and computing $F(x', y)$, with probability at least $\rho(\mathcal{A}) + r\delta$.

We now proceed with a standard hybrid argument. We define V_i the distribution obtained by picking y_j uniformly from $[D_j]$ for $j = 1, \ldots, i$, and z_j uniformly from $\{0, 1\}^{m_j}$ for $j = i + 1, \ldots, r$, and computing $F_1(x', y_1) \circ \cdots \circ F_i(x', y_i) \circ z_{i+1} \circ \cdots \circ z_r$. Notice that V_0 is the uniform distribution and hence $\Pr(T(V_0) = 1) = \rho(\mathcal{A})$ and V_r is the distribution $F(x', [D])$ and hence $\Pr(T(V_r) = 1) \geqslant \rho(\mathcal{A}) + \delta r$. It follows that there is an $i \in [1..r]$ for which

$$\Pr\big(T(V_i) = 1\big) - \Pr\big(T(V_{i-1}) = 1\big) \geqslant \delta.$$

Notice that in both distributions the $i + 1, \ldots, r$ blocks are chosen at random from $\{0, 1\}^{m_j}$ and in particular there is a way to fix them to values $z_{i+1}, \ldots, z_r$ that preserve the gap. Also, the ith block depends only on y_i and is independent of $y_1, \ldots, y_{i-1}$, hence there is a way to fix $y_1, \ldots, y_{i-1}$ and still preserve the gap. In particular there is a way to fix the first $i - 1$ blocks to $z_1, \ldots, z_{i-1}$ and still preserve the δ gap. Hence, when we try these values for $i, z_j, j \neq i$, and we apply the decoding algorithm of F_i we must have x among the outputs. Hence, it will also appear in the final output. Finally, everything that appears in the final output is checked, and hence correct. □

3. The building blocks

3.1. Universal extractors

We first give some standard definitions. Two distributions X_1, X_2 on the same universe Λ are ε-close if their ℓ_1 distance is small, i.e.,

$$\sum_{a \in \Lambda} \big| X_1(a) - X_2(a) \big| \leqslant \varepsilon.$$

The min-entropy of a distribution X on Λ is

$$H_\infty(X) = \min_{a \in \Lambda} \big\{ -\log_2 X(a) \big\}.$$

A function $E : \{0, 1\}^n \times \{0, 1\}^t \to \{0, 1\}^m$ is a strong (k, ε)-*extractor* if for every distribution X having k min-entropy, the distribution obtained by drawing x from X, y uniformly from $\{0, 1\}^t$ and evaluating $y \circ E(x, y)$, is within statistical distance ε from the uniform distribution on $\{0, 1\}^t \times \{0, 1\}^m$. The entropy loss of E is $k + t - m$, i.e., the difference between the entropy in the system, and the number of output bits.

Intuitively, extractors get an input x from a weak random source having at least k min-entropy, and with the help of a short truly random string y output $m(k)$ bits that are close to uniform. Notice, however, that the extractor is required to work only when the distribution X has enough min-entropy. We introduce new objects, that were first defined without a name in [9], to rectify this situation. We say $E' : \{0, 1\}^n \times \{0, 1\}^t \to \{0, 1\}^{m(k')}$ is a restriction of $E : \{0, 1\}^n \times \{0, 1\}^t \to \{0, 1\}^{m(k)}$ if there are $m(k')$ output bits such that for every $x \in \{0, 1\}^n$, $y \in \{0, 1\}^t$ the value of $E'(x; y)$ is the value $E(x; y)$ restricted to these bits. We say E is a strong (k, ε) **universal** extractor if for every $k' \leqslant k$ there is a restriction $E' : \{0, 1\}^n \times \{0, 1\}^t \to \{0, 1\}^{m(k')}$ of E which is a strong (k', ε) extractor.

Lemma 3. *Let* $E : \{0, 1\}^n \times \{0, 1\}^t \to \{0, 1\}^m$ *be a strong, universal* (k, ε) *extractor with* $k - m(k)$ *monotone in* k. *Then* $F : \{0, 1\}^n \times \{0, 1\}^t \to \{0, 1\}^{t+m}$ *defined by* $F(x, y) = y \circ E(x, y)$ *is a* $(n, k) \xrightarrow{2^{k-m}:1}_{2\varepsilon} t + m$ *strong condenser.*

Proof. As $F(x, y)$ has y as part of its output, we have no parallel edges in G_F. We are left to show property (2) of Definition 1. Let $\mathcal{A} \subseteq \{0, 1\}^n$ be any subset of cardinality at most 2^k. It is enough to prove that for $\mathcal{A}$ of size a power of two, we can throw away $\varepsilon D |\mathcal{A}|$ edges leaving $\mathcal{A}$ in G_F and get the desired $2^{k-m} : 1$ graph. So let us assume the cardinality of $\mathcal{A}$ is a power of two, say, $2^{k'}$.

E is a strong universal extractor, hence we can choose $m' = m'(k')$ output bits of E and get a function $E' : \{0, 1\}^n \times \{0, 1\}^t \to \{0, 1\}^{m'}$ that is a strong (k', ε) extractor. We show that we can remove $\varepsilon |\mathcal{A}| D$ edges leaving $\mathcal{A}$ in $G_{F'}$ (where $F'(x, y) = y \circ E'(x, y)$) such that at most $2^{k'-m'} \leqslant 2^{k-m}$ elements of $\mathcal{A}$ are mapped to the same vertex. As $F'(x, y)$ is a subset of the bits of $F(x, y)$ this clearly implies the same result for G_F.

Given a vertex in $V_2(G_{F'})$, namely, a fixed $y \in \{0, 1\}^t$ and a fixed $w' \in \{0, 1\}^{m'}$ let $\varepsilon_{w'}^y$ be

$$\left| \Pr_{a \in \mathcal{A}, u} \big(F'(a, u) = (y, w') \big) - \frac{1}{M'D} \right|,$$

where $D = 2^t$ and $M' = 2^{m'}$. As E' is a strong extractor we know that $\sum_{y,w'} \varepsilon_{w'}^y \leqslant \varepsilon$. Let us also denote $T_{w'}^y$ the number of elements $a \in A$ such that $E'(a, y) = w'$. Then

$$\varepsilon_{w'}^y = \left| \frac{|T_{w'}^y|}{|\mathcal{A}|D} - \frac{1}{M'D} \right|,$$

and therefore

$$|T_{w'}^y| \leqslant \frac{|\mathcal{A}|}{M'} + |\mathcal{A}|D\varepsilon_{w'}^y.$$

Both $|T_{w'}^y|$ and $\frac{|\mathcal{A}|}{M'} = 2^{k'-m'}$ are integers and therefore

$$|T_{w'}^y| \leqslant \frac{|\mathcal{A}|}{M'} + \lfloor |\mathcal{A}|D\varepsilon_{w'}^y \rfloor.$$

By erasing $\sum_{w',y} \lfloor |\mathcal{A}|D\varepsilon_{w'}^y \rfloor \leqslant |\mathcal{A}|D\varepsilon$ edges, every vertex (w', y) has at most $|\mathcal{A}|/M' = 2^{k'-m'} \leqslant 2^{k-m}$ edges going into it. $\quad\square$

3.2. Trevisan's extractor

Raz et al. [9] modified Trevisan's extractor and built a strong (k, ε) universal extractor

$$E_{\mathrm{TR}}^{k,\varepsilon} : \{0, 1\}^n \times \{0, 1\}^t \to \{0, 1\}^{m(k)}$$

with $t = \mathrm{O}(\log^2(n/\varepsilon))$ and $m(k) = k/2$. [1] We denote $F_{\mathrm{TR}}^{k,\varepsilon}(x, y) = y \circ E_{\mathrm{TR}}^{k,\varepsilon}(x, y)$. We thus get:

Lemma 4. *Let $k \leqslant n$, $m = k/2$ and $\varepsilon > 0$. Set $t = \mathrm{O}(\log^2(n/\varepsilon))$. There is an $(n, k) \overset{2^{k-m}:1}{\longrightarrow}_{2\varepsilon} m + t$ strong condenser $F_{\mathrm{TR}}^{k,\varepsilon}$.*

Furthermore, Ta-Shma and Zuckerman [13] proved: [2]

---

[1] An informal statement of this, with a brief explanation, appears in the conclusions section of [9]. We briefly sketch how to get this result from the Raz et al. [9] paper. Raz et al. [9] define weak designs, and also discuss a variant of it where $\sum_{j < i} 2^{|S_i \cap S_j|} \leqslant \rho(i - 1)$. We need the variant, and we name it prefix-weak design. Raz et al. [9] study Trevisan's extractor with prefix-weak design and show that for every $k' \leqslant k$ if we look at just the first $m(k')$ output bits we get a strong extractor. This gives a strong universal extractor with $t = \mathrm{O}(\log^2(n/\varepsilon))$ and $m(k') = k'/2$.

[2] Ta-Shma and Zuckerman's [13] is a revised (and generalized) version of [12]. In [12] Ta-Shma and Zuckerman prove a similar result but with a probabilistic decoder that has exponentially small (in n) probability of missing solutions that should appear in the output. Minor modifications to [12] give the deterministic decoder of [13] and Lemma 5.

Lemma 5 [12]. *For every $\varepsilon > 0$, $F_{\mathrm{TR}}^{k,\varepsilon}$ has efficient $\delta = \mathrm{O}(k\varepsilon^{1/4})$ decoding.*

3.3. Error correcting codes

$C \subseteq \{0, 1\}^{\bar{n}}$ is an $[\bar{n}, n, d]_2$ binary error correcting code if it is a subspace of $\{0, 1\}^{\bar{n}}$ of dimension n and the Hamming distance between any two elements in C is at least d. C is *explicit* if given k the kth codeword in C can be computed in time polynomial in $\bar{n}$. A binary $[\bar{n}, n, d]_2$ code has combinatorial list decoding property α if every Hamming ball of radius $(\frac{1}{2} - \alpha)\bar{n}$ has $\mathrm{O}(1/\alpha^2)$ codewords. Any $[\bar{n}, n, (\frac{1}{2} - \varepsilon)\bar{n}]_2$ code has combinatorial list decoding property $\sqrt{\varepsilon}$. We say a code has list decoding property α if for every $w \in \{0, 1\}^{\bar{n}}$ one can find in $\mathrm{poly}(\bar{n} \cdot 1/\alpha)$ time all the codewords that lie in a ball of radius $(\frac{1}{2} - \alpha)\bar{n}$ around w.

We will use an explicit $[\bar{n}, n, (\frac{1}{2} - \varepsilon)\bar{n}]_2$ binary error correcting code $C_{n,\varepsilon}$ with $\bar{n} = \mathrm{poly}(n/\varepsilon)$. Such a code can be obtained, e.g., by concatenating a Reed–Solomon code with a Hadamard code (see, e.g., [10]). Furthermore, in [5] it is shown that the above code has list decoding property $\beta = \sqrt{\varepsilon}$.

3.4. A simple condenser

Our second building block is the following strong condenser: let $x \in \{0, 1\}^n$, and $y_1, \ldots, y_\ell \in \{0, 1\}^{\log \bar{n}}$. Define,

$$C_{n,\varepsilon}^{(\ell)}(x; y_1, \ldots, y_\ell)$$
$$= y_1 \circ \cdots \circ y_\ell \circ C_{n,\varepsilon}(x)_{y_1} \circ \cdots \circ C_{n,\varepsilon}(x)_{y_\ell}.$$

Lemma 6. *Suppose $k \geqslant \log(1/\varepsilon) \geqslant 2$. $C_{n,\varepsilon}^{(9k)}$ is a $(n, k) \overset{1:1}{\longrightarrow}_\varepsilon \mathrm{O}(k\log(n/\varepsilon))$.*

Proof. For every pair $x_1 \neq x_2$, the probability over $y_1, \ldots, y_\ell$ they collide is smaller than $(\frac{3}{4})^\ell$ and for $\ell = 9k$ this is smaller than 2^{-3k}. Therefore the probability over $y_1, \ldots, y_\ell$ that there is any collision is at most $2^{2k}2^{-3k} \leqslant 2^{-k} \leqslant \varepsilon$. Hence, for all but ε fraction of the y, the mapping is $1 : 1$. $\quad\square$

Lemma 7. *Let $\delta > 0$ and set $\varepsilon = \delta^2$. $C_{n,\varepsilon}^{(1)}$ has efficient δ decoding.*

Proof. Let $\mathcal{A} \subseteq \{0,1\}^{d+1}$, $d = \log(\bar{n})$.

For a fixed $b \in \{0,1\}$ our decoding algorithm finds a set $L_b \subseteq \{0,1\}^n$ as follows:

- For every $y \in \{0,1\}^d$ if $y \circ b \in \mathcal{A}$ set z_y to be b otherwise to $\neg b$.
- Look at $z \in \{0,1\}^{\bar{n}}$ that has in the yth place the value z_y. $C_{n,\varepsilon}$ has list decoding property $\sqrt{\varepsilon} = \delta$. Run the list decoding algorithm on $z \in \{0,1\}^{\bar{n}}$ and get a list $L_b \subset \{0,1\}^n$ of size $O(1/\varepsilon)$ containing all inputs that correspond to codewords given by the list decoding algorithm.

The decoding algorithm, then, checks every element in $L = L_0 \cup L_1$ to see if it is indeed a good solution. Altogether, the running time of the decoding algorithm is poly($\bar{n}, 1/\delta$).

Now suppose $x \in \{0,1\}^n$ is such that $|\Gamma(x) \cap \mathcal{A}| \geqslant (\rho(\mathcal{A}) + \delta)D$ where $D = 2^d$. Define a test T that returns 1 iff $x \in \mathcal{A}$. Then $\Pr(T(U_{d+1}) = 1) = \rho(\mathcal{A})$ where U_{d+1} is the uniform distribution on $\{0,1\}^{d+1}$, while $\Pr(T(C^{(1)}(x, U_d))) = 1 \geqslant \rho(\mathcal{A}) + \delta$ where $C^{(1)}(x, U_d)$ is obtained by picking u uniformly from $\{0,1\}^d$ and computing $C^{(1)}(x, u) = u \circ C_{n,\varepsilon}(x)_u$. The first d bits in both distributions are truly uniform and so there is no gap there. The gap must be at the last bit. The by-now standard argument due to Yao (see, e.g., [10]) shows then that

$$\Pr_{b,y}\big[z_y = C(x)_y\big] \geqslant \tfrac{1}{2} + \delta.$$

In particular, there is some $b \in \{0,1\}$ such that

$$\Pr_y\big[z_y = C(x)_y\big] \geqslant \tfrac{1}{2} + \delta$$

and x appears in $L_b \subseteq L$. $\square$

Lemma 7 can be viewed as an algorithm that given $\mathcal{A}_1, \ldots, \mathcal{A}_{\bar{n}}$ with each $\mathcal{A}_i \subseteq \{0,1\}$, finds all codewords $c \in \{0,1\}^{\bar{n}}$ that have $c_i \in \mathcal{A}_i$ too often. The case where $|\mathcal{A}_i| = 1$ is the standard list decoding problem. Thus, Lemma 7 solves a generalization of the list decoding problem.

3.5. Putting it together

We now want to prove Theorem 2. For every $K = K(N) < N$ and $\varepsilon = \varepsilon(N)$ we look for a graph $G_F = (V_1 = [N = 2^n], V_2 = [M = 2^m], E)$, that is, $(2K, (1 - \varepsilon)D)$ expanding and has explicit 2ε

decoding, with M being small. Alternatively, we look for a function $F : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ that is a good condenser, $M = 2^m$, $D = 2^d$. We set $\varepsilon' = \varepsilon/n$, and let $k = \log(K)$. We define the function

$$F = F_{\mathrm{TR}}^{k,((\varepsilon'/n))^4} \circ F_{\mathrm{TR}}^{(k/2),((\varepsilon'/n))^4} \circ \cdots$$
$$\circ F_{\mathrm{TR}}^{k',(\varepsilon'/n)^4} \circ C_{n,(\varepsilon')^2}^{(9k')},$$

where k' is of order $O(\log^2(n/\varepsilon'))$.[3]

Lemma 8. $F : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ is a

$$(n, k) \xrightarrow{1:1}_\varepsilon k + O(\log^3 n/\varepsilon),$$

and therefore G_F is $(K, (1-\varepsilon)D)$ expanding, for $K = 2^k$, $D = 2^d$. Furthermore, F has efficient ε decoding.

Proof. The number of output bits of

$$F_{\mathrm{TR}}^{k,(\varepsilon'/n)^4} \circ F_{\mathrm{TR}}^{k/2,(\varepsilon'/n)^4} \circ \cdots \circ F_{\mathrm{TR}}^{k',(\varepsilon'/n)^4}$$

is at most k. We also have $O(k') = O(\log^2(n/\varepsilon))$ invocations of $C^{(1)}$ with $O(\log(\bar{n})) = O(\log(n/\varepsilon))$ output bits each. This totals to $k + O(\log^3 n/\varepsilon)$. Lemma 1 says the graph is a $1 : 1$ condenser up to $O(\log(n)\varepsilon') + O(k'2^{-\Omega(k')}) \leqslant \varepsilon$ error. Finally, notice that $C^{(\ell)}$ is a concatenation of ℓ $C^{(1)}$ codes. Each concatenated condenser has ε' decoding. Lemma 2 then tells us F has $O(\log^2(n/\varepsilon))\varepsilon' < \varepsilon$ decoding.

Acknowledgements

---

[3] We note that Raz et al. [9] also show how to reduce the entropy loss by repeatedly taking the basic extractor until the entropy loss is small enough to allow using an extractor with an optimal entropy loss. One can see that the repeated extractor is also universal; for $k' \leqslant k$ one can take the first $m(k')$ output bits of the first extractor (with entropy loss $k'' = k' - m(k')$), and the first $m(k'')$ of the second extractor and so on, and then one gets a strong (k', ε) strong extractor. Thus, one can get a universal extractor with $t = O(\log^3(n/\varepsilon))$ and optimal entropy loss. Indeed, we could have used this universal extractor, as long as we also show that this special extractor has also efficient decoding.

References

[1] A. Brodnik, J.I. Munro, Membership in constant time and minimum space, in: J. van Leeuwen (Ed.), Algorithms—ESA'94, Lecture Notes in Comput. Sci., Vol. 855, Springer, Berlin, 1994, pp. 72–81.

[2] H. Buhrman, P.B. Miltersen, J. Radhakrishnan, S. Venkatesh, Are bitvectors optimal?, in: Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, 2000, pp. 449–458.

[3] M.L. Fredman, J. Komlos, E. Szemeredi, Sorting a sparse table with o(1) worst case access time, J. ACM 31 (3) (1984) 538–544.

[4] A. Fiat, M. Naor, J.P. Schmidt, A. Siegel, Non-oblivious hashing, J. ACM 39 (1992) 764–782.

[5] V. Guruswami, M. Sudan, List decoding algorithms for certain concatenated codes, in: Proceedings of the 32nd Annual ACM Symposium on Theory of Computing, 2000, pp. 181–190.

[6] M. Minsky, S. Papert, Perceptrons, MIT Press, Cambridge, MA, 1969.

[7] R. Pagh, Low redundancy in static dictionaries with O(1) worst case lookup time, in: J. Wiedermann, P. van Emde Boas, M. Nielsen (Eds.), ICALP'99, Lecture Notes in Comp. Sci., Vol. 1644, Springer, Berlin, 1999, pp. 595–604.

[8] R. Raz, O. Reingold, On recycling the randomness of states in space bounded computation, in: Proceedings of the 31st Annual ACM Symposium on Theory of Computing, 1999, pp. 159–168.

[9] R. Raz, O. Reingold, S. Vadhan, Extracting all the randomness and reducing the error in trevisan's extractors, in: Proceedings of the 31st Annual ACM Symposium on Theory of Computing, 1999, pp. 159–168.

[10] L. Trevisan, Construction of extractors using pseudo-random generators (extended abstract), in: Proceedings of the 31st Annual ACM Symposium on Theory of Computing, 1999, pp. 141–148.

[11] A. Ta-Shma, C. Umans, D. Zuckerman, Loss-less condensers, unbalanced expanders, and extractors, in: Proceedings of the 33rd Annual ACM Symposium on Theory of Computing, 2001, pp. 143–152.

[12] A. Ta-Shma, D. Zuckerman, Extractor codes (extended abstract), in: Proceedings of the 33rd Annual ACM Symposium on Theory of Computing, 2001, pp. 193–199.

[13] A. Ta-Shma, D. Zuckerman, Extractor codes (full version), Manuscript, 2001.

[14] A. Wigderson, D. Zuckerman, Expanders that beat the eigenvalue bound: Explicit construction and applications, in: Proceedings of the 25th Annual ACM Symposium on Theory of Computing, 1993, pp. 245–251.

[15] A. Yao, Should tables be sorted?, J. ACM 28 (3) (1981) 615–628.