CS225: Pseudorandomness Prof. Salil Vadhan

Lecture 12: Constructing Extractors

March 20, 2007

Based on scribe notes by Adam Kirsch.

In the previous lectures, we have seen that very good extractors exist — extracting almost all of the min-entropy from a source with only a logarithmic seed length. But the explicit constructions we have seen (via pairwise-independent hashing and spectral expanders) are still quite far from optimal in seed length, and in particular cannot be used to give a polynomial-time simulation of **BPP** with a weak random source.

Fortunately, much better extractor constructions are known — ones that extract any constant fraction of the min-entropy using a logarithmic seed length, or extract all of the min-entropy using a polylogarithmic seed length. In this lecture, we will see how to construct such extractors (assuming a certain *condenser* construction that we will see after Spring Break).

1 Block Sources

We introduce a useful model of source that has more structure than an arbitrary k-source:

Definition 1 *$X = (X_1, X_2, \ldots, X_t)$ is a $(k_1, k_2, \ldots, k_t)$* block source *if for every $x_1, \ldots, x_{i-1}$, $X_i|_{X_1=x_1,\ldots,X_{i-1}=x_{i-1}}$ is a k_i-source. If $k_1 = k_2 = \cdots = k_t = k$, then we call X a $t \times k$* block source.

Note that a $(k_1, k_2, \ldots, k_t)$ block source is also a $(k_1 + \cdots + k_t)$-source, but it comes with additional structure — each block is guaranteed to contribute some min-entropy. Thus, extracting randomness from block sources is easier task than extracting from general sources.

The study of block sources has a couple of motivations.

- They are a natural and plausible model of sources in their own right. Indeed, they are more general than Santha–Vazirani (SV) sources: if an SV source of parameter δ is broken into t blocks of length $\ell = n/t$, then the result is a $t \times \delta'\ell$ block source, where $\delta' = \log(1/(1-\delta))$. (The definition of SV sources is given in Lecture 10.)
- We can construct extractors for general weak sources by converting a general weak source into a block source. We will see how to do this later in the lecture.

We now illustrate how extracting from block sources is easier than from general sources.

Lemma 2 *Let $\mathrm{Ext}_1 : \{0,1\}^{n_1} \times \{0,1\}^{d_1} \to \{0,1\}^{m_1}$ be a (k_1, ε_1)-extractor, and $\mathrm{Ext}_2 : \{0,1\}^{n_2} \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}$ be a (k_2, ε_2)-extractor with $m_2 \geq d_1$. Define $\mathrm{Ext}'((x_1, x_2), y_2) = (\mathrm{Ext}_1(x_1, y_1), z_2)$, where $(y_1, z_2) = \mathrm{Ext}_2(x_2, y_2)$.*

Then for every (k_1, k_2) block source $X = (X_1, X_2)$ taking values in $\{0,1\}^{n_1} \times \{0,1\}^{n_2}$, it holds that $\mathrm{Ext}'(X, U_{d_2})$ is $(\varepsilon_1 + \varepsilon_2)$-close to $U_{m_1} \times U_{m_2 - d_1}$.

Proof: $(X_1, Y_1, Z_2) = (X_1, \mathrm{Ext}_2(X_2, U_{d_2}))$ is ε_2-close to $(X_1, U_{m_2}) = (X_1, U_{d_1}, U_{m_2-d_1})$.
Thus, $(\mathrm{Ext}_1(X_1, Y_1), Z_2)$ is ε_2-close to $(\mathrm{Ext}_1(X_1, U_{d_1}), U_{m_2-d_1})$, which is ε_1-close to $(U_{m_1}, U_{m_2-d_1})$.
By the triangle inequality, $\mathrm{Ext}'(X, U_{d_2}) = (\mathrm{Ext}_1(X_1, Y_1), Z_2)$ is $(\varepsilon_1+\varepsilon_2)$-close to $(U_{m_1}, U_{m_2-d_1})$. ■

The benefit of this compositoin is that the seed length of Ext' depends only on that of Ext_2. Thus, we get to extract from multiple blocks at the 'price of one'. It is even better than this because typically $m_2 > d_2$, so the seed length of Ext' can even be smaller than that of Ext_1. The lemma extends naturally to extracting from many blocks (as opposed to just two.)

Another remark is that this composition preserves 'strongness'. If Ext_1 and Ext_2 correspond to strong extractors in the sense that their seeds are part of their outputs, then Ext' will also correspond to a strong extractor. If in addition $d_1 = d_2$, then this construction can be seen as simply using the same seed to extract from both blocks.

Already with this simple composition, we can simulate **BPP** with a Santha-Vazirani source. As noted above, by breaking a SV source X with parameter δ into blocks of length ℓ, we obtain a $t \times k$ block source for $t = n/\ell$, $k = \delta'\ell$, and $\delta' = \log(1/(1-\delta))$.

Suppose that δ is a constant. Set $\ell = (10/\delta')\log n$, so that X is a $t \times k$ block source for $k = 10 \log n$, and define $\varepsilon = n^{-2}$. Letting $\mathrm{Ext} : \{0,1\}^\ell \times \{0,1\}^d \to \{0,1\}^{d+m}$ be a (k, ε) extractor created using pairwise independent hash functions, we have

$$\begin{aligned} d &= O(\ell) = O(\log n) \qquad \text{and} \\ m &= k - 2\log\frac{1}{\varepsilon} - O(1) > k/2 \end{aligned}$$

Composing Ext with itself t times as in Lemma 2, we obtain $\mathrm{Ext}' : \{0,1\}^{\frac{n}{\ell}\cdot\ell} \times \{0,1\}^d \to \{0,1\}^{d+\frac{n}{\ell}\cdot m}$, we have that $\mathrm{Ext}'(X, U_d)$ is ε'-close to uniform, for $\varepsilon' = 1/n$. This tells us that Ext' essentially extracts half of the min-entropy from X, given a random seed, and gives us the following result.

Theorem 3 *For any constant δ, we may simulate any* **BPP** *algorithm with an SV source with parameter δ in polynomial time.*

Proof: Use the above construction of Ext' to supply random bits to the algorithm. Enumerate over all possible seeds, and take a majority vote. ■

As one might expect, the above procedure performs very badly when δ is not a constant. The reason is that the big-O notation in the analysis hides a $(1/\delta')$ in the exponent of the running time. This can be overcome using extractors based on almost-pairwise independence, as on Problem Set 4.

2 Reducing General Sources to Block Sources

Given the results of the previous section, a common approach to constructing extractors for general k-sources is to reduce the case of general k-sources to that of block sources.

One approach to doing this is as follows. Given a k-source X of length n, where $k = \delta n$, pick a (pseudo)random subset S of the bits of X, and let $W = X|_S$ be the bits of X in those positions.

If the set S is of size ℓ, then we expect that W will have at roughly $\delta\ell$ bits of min-entropy (with high probability over the choice of S). Moreover, W can have at most ℓ bits of min-entropy, so if $\ell < \delta n$, there must still should be at least min-entropy left in X. Thus, the pair (W, X) should be a block source. This approach can be shown to work for appropriate ways of sampling the set S, and recursive applications of it was the original approach to constructing good extractors (and is still useful in various contexts today).

Another approach, which we will follow, is based on the observation that every source of high min-entropy rate (namely, greater than 1/2) is (close to) a block source, as shown by the lemma below. Thus, we will try to reduce the case of extracting from general sources to extracting from sources of high min-entropy rate.

Lemma 4 (exercise) *If (W, X) are two jointly distributed random variables, where (W, X) is a k-source and W has length at most ℓ, then for every $\varepsilon > 0$, it holds that with probability at least $1 - \varepsilon$ over $w \stackrel{R}{\leftarrow} W$, $X|_{W=w}$ is a $(k - \ell - \log(1/\varepsilon))$-source.*

Corollary 5 *If X is a $(n-\Delta)$-source of length n, and $X = (X_1, X_2)$ is a partition of X into blocks of lengths n_1 and n_2, then (X_1, X_2) is ε-close to some $(n_1 - \Delta, n_2 - \Delta - \log(1/\varepsilon))$ block source.*

Consider $\Delta = \alpha n$ for a constant $\alpha < 1/2$, and $n_1 = n_2 = n/2$. Then each block contributes min-entropy at least $(1/2 - \alpha)n$.

But we are still left with the question of converting a general k-source into one of high min-entropy rate. We will do this via the following kind of object, which we will see how to construct after Spring Break.

Definition 6 *A function* $\mathrm{Con} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is a* $k \to_\varepsilon k'$ condenser *if for every k-source X on $\{0,1\}^n$, $\mathrm{Con}(X, U_d)$ is ε-close to some k'-source.* Con *is* lossless *if $k' = k + d$.*

If $k'/m > k/n$, then the condenser increases the min-entropy rate, thereby making extraction an easier task.

Theorem 7 *For every constant $\alpha > 0$, for all positive integers $n \geq k$ and all $\epsilon > 0$, there is an explicit*

$$k \to_\varepsilon k + d$$

lossless condenser $C : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ with $d = O(\log n + \log(1/\varepsilon))$ and $m = (1+\alpha)k + O(\log(n/\varepsilon))$.

Note that setting α to be a small constant, we obtain an output min-entropy rate arbitrarily close to 1. On the homework, you will already see how to construct good extractors for high min-entropy rate when the error parameter ε is constant. In the next section, we will see how to do so for arbitrary values of ε.

3 The Extractor

In this section, we will use the ideas outlined in the previous section — namely condensing and block-source extraction — to construct an extractor that is optimal up to constant factors (assuming the condenser of Theorem 7).

Theorem 8 *Assume Theorem 7. Then for all positive integers $n \geq k$ and all $\varepsilon > 0$, there is an explicit (k, ε) extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with $m = k/2$ and $d = O(\log(n/\varepsilon))$.*

We will use the following building block, which you will construct on the homework.

Lemma 9 *Assume Theorem 7. Then for every* constant $t > 0$ *and all positive integers $n \geq k$ and all $\varepsilon > 0$, there is an explicit (k, ε) extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with $m = k/2$ and $d = k/t + O(\log(n/\varepsilon))$.*

The point is that this extractor has a seed length that is an arbitrarily large constant factor (namely $t/2$) smaller than its output length. Thus, if we use it as Ext_2 in the block-source extraction of Lemma 2, the resulting seed length will be smaller than that of Ext_1 by an arbitrarily large constant factor.

We now proceed with the proof of Theorem 8. Our approach will be to reduce the task of extracting from sources of min-entropy k to that of extracting from sources of min-entropy at most $k/2$.

Given n and $\varepsilon > 0$, we construct the following by induction on i:[1] For every $k \leq 2^i \cdot 8d$, we will provide an explicit (k, ε_i) extractor $\mathrm{Ext}_k : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{k/2}$ where $d = c\log(n/\varepsilon_0)$ and $\varepsilon_0 = \varepsilon/n^c$ for a constant c to be determined in the proof and the ε_i's are an increasing sequence that we will bound later.

Note that the seed length d and the fraction of min-entropy extracted (namely 1/2) should be independent of i.

The base case ($i = 0$) follows from Lemma 9, setting $t = 9$ and taking c to be a sufficiently large constant.

For the induction step, suppose the above holds for $i = j - 1$ and we will prove it for $i = j$. Given a k-source X of length n for $k \leq 2^j \cdot (8d)$, we proceed as follows:

1. We apply the condenser of Theorem 7 to obtain X' that is ε_0-close to a k-source of length $(1+\alpha)k + O(\log(n/\varepsilon_0))$, where α is a sufficiently small constant. This costs us $O(\log(n/\varepsilon_0))$ random bits.
2. We divide X' into two equal-sized halves (X_1, X_2). By Corollary 5, (X_1, X_2) is $2\varepsilon_0$-close to a $2 \times k'$ block source for $k' = k/2 - \alpha k/2 - O(\log(n/\varepsilon_0)$.
3. Since we may assume that $k > 8d$ ($k \leq 8d$ is handled by the base case), we have that the min-entropy in the second block is

$$k' = k/2 - \alpha k/2 - O(\log(n/\varepsilon_0)) \geq 2d,$$

[1] Since i is unbounded, we cannot actually prove the explicitness of the construction by induction. That is, a running time of $p_i(n)$ where p is a polynomial whose degree depends arbitrarily on i would not be acceptable. However, the construction described gives an explicit recursive algorithm that is easily seen to be polynomial time.

for a sufficiently small choice of the constant α and a sufficiently large choice of the constant c.

4. Thus, using the extractor of Lemma 9 with $t = 16$, we can extract d bits from the second block using a seed of length $(2d)/16 + O(\log(n/\varepsilon_0))$ for a sufficiently large choice of the constant c.

5. By Lemma 2, we can use these d bits as a seed to $\mathrm{Ext}_{k'}$ to recursively extract $k'/2$ bits from the first block with error ε_{j-1}. (Note that $k' \leq k/2 \leq 2^{j-1} \cdot (8d)$.)

All in all, our new extractor has seed length at most $d/8 + O(\log(n/\varepsilon_0))$, error $\varepsilon_j \leq \varepsilon_{j-1} + O(\varepsilon_0)$, and output length $k'/2 \geq k/5$. This would be sufficient for our induction except that the output length is only $k/5$ rather than $k/2$. This can be remedied by applying the extractor several times.

Lemma 10 *Suppose* $\mathrm{Ext}_1 : \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^{m_1}$ *is a* (k_1, ε_1) *extractor and* $\mathrm{Ext}_2 : \{0,1\}^n \times \{0,1\}^{d_2} \to \{0,1\}^{m_2}$ *is a* (k_2, ε_2) *extractor for* $k_2 = k - m_1 - \log(1/\varepsilon_3)$. *Then* $\mathrm{Ext}' : \{0,1\}^n \times \{0,1\}^{d_1+d_2} \to \{0,1\}^{m_1+m_2}$ *is a* $(k_1, \varepsilon_1 + \varepsilon_2 + \varepsilon_3)$ *extractor.*

With one application of our extractor we extract $k/5$ bits of the source min-entropy. With the second application, we extract another $(4k/5 - \log(1/\varepsilon_0))/5$ bits, and so on. After four applications, we have extracted all but $(4/5)^4 \cdot k + O(\log(1/\varepsilon_0) \leq k/2$ bits of the min-entropy. Our seed length is then $4 \cdot (d/8 + O(\log(n/\varepsilon_0))) \leq d$ and the error is $\varepsilon_j = O(\varepsilon_{j-1}) = 2^{O(j)} \cdot \varepsilon_0 \leq \varepsilon$.

This completes the proof of Theorem 8, of course assuming Theorem 7 (which we will prove after Spring Break).

Applying Lemma 10 to Theorem 8, we can extract any constant fraction of the min-entropy using a logarithmic seed length, and all the min-entropy using a polylogarithmic seed length.

Corollary 11 *Assume Theorem 7. Then for every constant* $\alpha > 0$ *and all positive integers* $n \geq k$ *and all* $\varepsilon > 0$*, there is an explicit* (k, ε) *extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with* $m = (1-\alpha)k$ *and* $d = O(\log(n/\varepsilon))$.

Corollary 12 *Assume Theorem 7. Then for all positive integers* $n \geq k$ *and all* $\varepsilon > 0$*, there is an explicit* (k, ε) *extractor* $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with* $m = k + d - O(\log(1/\varepsilon))$ *and* $d = O(\log k \cdot \log(n/\varepsilon))$.

For the latter, an extractor that extracts $k + d - O(\log(1/\varepsilon))$ bits of min-entropy using a seed of length $O(k + \log(n/\varepsilon))$ should be used for the final extraction to ensure that we do not lose the randomness of the seed. Such an extractor can be obtained by combining Theorem 7 with pairwise-independent hashing or by an extractor based on almost pairwise-independent hashing.

A summary of the extractor parameters we have seen is in Table 1.

Method	Seed Length d	Output Length m
Optimal and Nonconstructive	$\log(n-k)+O(1)$	$k+d-O(1)$
Necessary for **BPP** Simulation	$O(\log n)$	$k^{\Omega(1)}$
Spectral Expanders	$O(n-k)$	n
Pairwise Independent Hashing	$O(n)$	$k+d-O(1)$
Corollary 11	$O(\log n)$	$(1-\gamma)k$, any constant $\gamma>0$
Corollary 12	$O(\log^2 n)$	$k+d-O(1)$

Table 1: Parameters for some constructions of (k,ε) extractors, where $\varepsilon>0$ is an arbitrarily small constant.