CAS CS 599 B: Mathematical Methods for TCS

Lecturer: Mark Bun

Spring 2022

Lecture Notes 12:

More on extractors, Nisan's PRG

Reading.

- Vadhan, Sections 6.1, 6.2.1
- Avishay Tal's notes:
  `https://drive.google.com/file/d/1DXdmCAf6rqARVFwFx4h6kpI1Mk2a4iJN/view`

Last time, we defined an extractor as a function that takes a single sample from a weak random source, and outputs a near-uniform sample.

Definition 1. A deterministic extractor for a class $\mathcal{C}$ of sources over $\{0,1\}^n$ is a function $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}^m$ if $TV(\mathrm{Ext}(X), \mathcal{U}_m) \leq \varepsilon$ for every $X \in \mathcal{C}$.

We also saw that a necessary condition for an extractor to produce m (exactly) uniform bits is that the source has min-entropy at least m.

Definition 2. The min-entropy of a source X is

$$H_\infty(X) = \min_{x \in \mathrm{supp}(X)} \log \frac{1}{\Pr[X = x]}.$$

One way to think about $\log(1/\Pr[X = x])$ is as a measure of how surprising it is to observe the outcome $X = x$. The lower the probability of an outcome, the more surprising it is. If a source has high min-entropy, that means every outcome is somewhat surprising. Shannon entropy, which is the same thing as min-entropy but where the $\min_{x \sin \mathrm{supp}(X)}$ is replaced by $\mathbb{E}_{x \sim X}$, by contrast measures the *average* surprisal of an outcome $x \sim X$. The two measures coincide when X is the uniform distribution, but differ on non-uniform distributions. Min-entropy is a lower bound on Shannon entropy and is thus a more conservative estimate for how much randomness is present in a source X.

Definition 3. An (n,k) source is a random variable X on $\{0,1\}^n$ such that $H_\infty(X) \geq k$. Equivalently, $\Pr[X = x] \leq 2^{-k}$ for all $x \in \{0,1\}^n$.

Some examples of (n,k) sources include:

- Von Neumann sources. n i.i.d. bits each with bias $\delta < 1/2$ comprise an (n,k) source for $k = n\log(1/(1-\delta))$.
- Bit-fixing sources. An oblivious bit-fixing source is one where k bits are chosen uniformly at random, at the rest are fixed to constants. A non-oblivious bit-fixing source is where k bits are chosen at random, and the rest are chosen depending on the outcomes of those k bits.

- Flat sources. These are distributions which are uniform over a subset $S \subseteq \{0,1\}^n$ of size $|S| = 2^k$. You can think of flat sources as the building blocks of general (n,k) sources, in the sense that every (n,k) source is a convex combination (mixture) of flat sources.

We'd like to be able to construct extractors for the class of all (n,k) sources. Unfortunately, this still turns out to be impossible using our definition of deterministic extractors:

Claim 4. *For any* $\mathrm{Ext} : \{0,1\}^n \to \{0,1\}$*, there exists an* $(n-1)$*-source* X *such that* $\mathrm{Ext}(X)$ *is constant.*

Proof. Since $|\mathrm{Ext}^{-1}(0)| + |\mathrm{Ext}^{-1}(1)| = 2^n$, there exists a $b \in \{0,1\}$ such that $|\mathrm{Ext}^{-1(b)}| \geq 2^{n-1}$. Let X be the uniform distribution on $\mathrm{Ext}^{-1}(b)$. □

Faced with this impossibility result, there are a few workarounds. One, of course, is to restrict our attention to structured classes of (n,k) sources, such as bit-fixing sources or sources exhibiting algebraic structure (e.g., dimension-k affine subspaces of $\mathbb{F}_2^n$). Another is to consider extractors for two or several *independent* (n,k) sources. We will focus on *seeded* extractors, which take as input an (n,k) source as well as an independent (and ideally short) d-bit seed of uniform randomness, and output a long string of $\approx k+d$ bits of uniform randomness.

1 Seeded Extractors

Definition 5. A (k,ε)-seeded extractor is a function $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ such that $TV(\mathrm{Ext}(X,\mathcal{U}_d),\mathcal{U}_m) \leq \varepsilon$ for every (n,k) source X.

The parameters that tell us how good a seeded extractor are the seed length d and the output length m. We want to minimize d and maximize m. Note that achieving $m = d$ is easy by just outputting the seed. For a given d, we want to be able to take m as close as possible to $d+k$.

One can use the probabilistic method to show that a randomly chosen function is a (k,ε)-extractor when

$$d = \log(n-k) + 2\log(1/\varepsilon) + O(1), \qquad m = d + k - 2\log(1/\varepsilon) - O(1).$$

These parameters are excellent if our goal is to simulate randomized algorithms using a weak random source. Since the seed length is logarithmic, we can perform a partial derandomization by enumerating overall possible seeds and taking a majority vote. The challenge is to construct explicit, efficiently computable extractors matching this bound.

There are several explicit constructions of extractors nearly matching these bounds based on expanders, error-correcting codes, and PRGs. We don't yet have the tools to construct these extractors, but we can present a simple and important one based only on pairwise independence.

Definition 6. A family of "hash" functions $\mathcal{H} = \{h_r : \{0,1\}^n \to \{0,1\}^t \mid r \in \{0,1\}^d\}$ is *pairwise independent* if for every $x \neq x' \in \{0,1\}^n$ and $y, y' \in \{0,1\}^t$,

$$\Pr_{r \sim \{0,1\}^d}[h_r(x) = y \wedge h_r(x') = y'] = 2^{-2t}.$$

In other words, the random variables $(h_r(x))_{x \in \{0,1\}^n}$, for $r \leftarrow \{0,1\}^d$, are pairwise independent. Recall that we can sample such hash functions using seed length $d = 2\max\{n,t\}$.

Theorem 7 (Leftover Hash Lemma)**.** *Let $\mathcal{H}$ be a pairwise independent hash family where $t = k - 2\log(1/\varepsilon)$. Then*

$$\mathrm{Ext}(x, r) = (r, h_r(x))$$

is a (k, ε)-seeded extractor.

Note that this extractor has very poor seed length $d = O(n)$, where as the probablistic method tells us we can achieve seed length $O(\log(n/\varepsilon))$. However, the rate of extraction $m = d + t = d + k - 2\log(1/\varepsilon)$ is optimal.

Proof. We will actually show something stronger, that Ext is an extractor with respect to collision probability. That is, the probability that two independent samples from Ext collide is roughly the same as the probability that two uniform samples collide. More precisely:

Definition 8. The collision probability of a distribution $\mathcal{D}$ is $C(\mathcal{D}) = \Pr_{Z,Z' \sim \mathcal{D}}[Z = Z'] = \sum_{z \in \mathrm{supp}(\mathcal{D})} \mathcal{D}[z]^2$.

Note that the collision probability of the uniform distribution on m bits is $C(\mathcal{U}_m) = 2^{-m}$. The collision probability of any (n, k) source X is at most $\sum_{x \in \mathrm{supp}(X)} \Pr[X = x]^2 \leq (\max_x \Pr[X = x]) \sum_x \Pr[X = x] = 2^{-k}$.

Putting the following two lemmas together proves the claim.

Lemma 9. *For any distribution $\mathcal{D}$ on $\{0,1\}^m$, we have $TV(\mathcal{D}, \mathcal{U}_m) \leq 2^{-m/2-1}\sqrt{C(\mathcal{D}) - 2^{-m}}$.*

Lemma 10. *If X is an (n, k) source, then $C(\mathrm{Ext}(X, \mathcal{U}_d)) \leq \frac{1+\varepsilon^2}{2^m}$.*

□

Proof of Lemma 9. Let p be the 2^m-dimensional vector representing the probability mass function of $\mathcal{D}$, and let $u = (2^{-m}, \ldots, 2^{-m})$ be the vector representing the uniform distribution. Then

$$\begin{aligned}
TV(\mathcal{D}, \mathcal{U}_m) &= \frac{1}{2}\|p - u\|_1 \\
&\leq \frac{\sqrt{2^m}}{2}\|p - u\|_2 && \text{by Cauchy-Schwarz} \\
&\leq 2^{-m/2-1}\sqrt{\sum_{z \in \{0,1\}^m} (p_z - 2^{-m})^2} \\
&\leq 2^{-m/2-1}\sqrt{\sum_z p_z^2 - 2\sum_z 2^{-m} p_z + \sum_z 2^{-2m}} \\
&\leq 2^{-m/2-1}\sqrt{\left(\sum_z p_z^2\right) - 2^{-m}}.
\end{aligned}$$

Noting that $C(\mathcal{D}) = \sum_z p_z^2$ completes the proof. □

Proof of Lemma 10. We estimate

$$\begin{aligned} C(\mathrm{Ext}(X,\mathcal{U}_d)) &= \Pr_{\substack{r,r'\sim\{0,1\}^d \\ x,x'\sim X}}[(r,h_r(x)) = (r',h_{r'}(x'))] \\ &= 2^{-d} \Pr_{r\sim\{0,1\}^d, x,x'\sim X}[h_r(x) = h_r(x')] \\ &\leq 2^{-d}\left(\Pr[x=x'] + \Pr[h_r(x) = h_r(x') \mid x \neq x']\right) \\ &\leq 2^{-d}(2^{-k} + 2^{-t}). \end{aligned}$$

The last inequality holds because $H_\infty(X) \geq k$ implies that the collision probability as at most 2^{-k}, and by pairwise independence. Taking $k = t + 2\log(1/\varepsilon)$ makes this $(1+\varepsilon^2)2^{-(d+t)}$ as we wanted. $\square$

2 Nisan's PRG

The leftover hash lemma illustrates how PRG technology can be used to construct extractors. Nisan's PRG is an important example of a construction that goes in the opposite direction. It uses a seeded extractor to construct a PRG that fools space-bounded computation. To describe the guarantees of the generator, let us first introduce a simple combinatorial model for space-bounded computation.

Definition 11. A read-once branching program (ROBP) is a directed acyclic graph where the vertices are organized into a grid of n layers, with w vertices in each layer, plus a single start vertex in layer 0. Every vertex in layers $0,\dots,n-1$ has exactly two outgoing edges to vertices in the next layer, labeled by 0 or 1. Every vertex in layer n is additionally labeled with an outcome, either "accept" (1) or "reject" (0).

On input $(x_1,\dots,x_n)$, the ROBP computes by successively taking each edge labeled x_i from layer $i-1$ to layer i. It outputs the decision labeling the vertex it reaches in layer n.

The parameter w is called the *width* of the ROBP, and n is called the *length*.

You should think of "small space" as corresponding to $w = \mathrm{poly}(n)$, i.e., the number of bits $\log w$ needed to describe the state of the branching program is only logarithmic in the length of the input.

Theorem 12 (Nisan)**.** *For all n, w, ε, there exists a log space computable PRG $G : \{0,1\}^\ell \to \{0,1\}^n$ that ε-fools every width-w, length-n ROBPs, using seed length $\ell = O(\log n \cdot \log(nw/\varepsilon))$.*

Consequences for derandomizing space-bounded computation:

- Consider a probabilistic Turing machine $M(y;x)$ (where y represents the "real" input and x represents the random coin tosses) with read-only one-way access to its input and random tape, and working space s. For every fixed input y, such a TM is a length-$|x|$, width-2^s ROBP as a function of x. One can derandomize M by enumerating over all 2^ℓ seeds of Nisan's PRG and taking the majority vote.

  Formally, this shows that the complexity class $\mathbf{BPL}$ of languages decidable in logarithmic space and polynomial time on a probabilistic TM is contained in $\mathbf{SPACE}(\log^2 n)$.

- Nisan's generator is a powerful tool in the design of streaming algorithms. A streaming algorithm gets one pass over a data stream of length n and aims to compute some property of the stream using space $\mathrm{poly}(\log n)$. Many low-space algorithms can be analyzed in the presence of a long string of random bits, but this is too much for a streaming algorithm to store. So one can in stead store the seed to Nisan's PRG and generate pseudorandom bits on-the-fly.

Here's the idea behind Nisan's PRG. Suppose we feed a length-n, width-w ROBP uniformly random bits $x_1, \ldots, x_{n/2}$ to get it from the start vertex v_0 to a vertex $v_{n/2}$ in the middle layer. Since the ROBP runs in small space, it can only "remember" $\log w$ bits of information about the random string $(x_1, \ldots, x_{n/2})$ via the vertex $v_{n/2}$. In other words, conditioned on $v_{n/2}$, the prefix $(x_1, \ldots, x_{n/2})$ still has min-entropy $n/2 - \log w$. One can then apply a seeded extractor to the prefix, say with seed length $O(\log w)$, to extract $n/2$ uniform bits for the rest of the computation. The total PRG seed length is now only $n/2 + O(\log w)$, which is progress. To get down to polylogarithmic seed length, the idea is then to apply this recycling procedure recursively.

Lemma 13 (Recycling lemma)**.** *Let $f : \{0,1\}^n \to [w]$ and let $\mathrm{Ext} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a (k, ε)-seeded extractor for $k = n - \log(w/\varepsilon)$. Then for $X \sim \mathcal{U}_n$ and $Z \sim \mathcal{U}_d$,*

$$TV(f(X) \circ \mathcal{U}_m, f(X) \circ \mathrm{Ext}(X, Z)) \leq 2\varepsilon.$$

Proof. For any vertex $v \in [w]$, let $X_v = (X \mid f(X) = v)$ be the uniform distribution on $f^{-1}(v)$. A short calculation shows

$$TV(f(X) \circ \mathcal{U}_m, f(X) \circ \mathrm{Ext}(X, Z)) = \sum_{v \in [w]} TV(\mathcal{U}_m, \mathrm{Ext}(X_v, Z)) \cdot \Pr[f(X) = v].$$

Define a "good" set G values v by $G = \{v \mid \Pr[f(X) = v] \geq \varepsilon/w\}$. By definition, if $v \in G$, then $H_\infty(X_v) \geq n - \log(w/\varepsilon)$. Meanwhile, most outcomes v are good in that $\Pr[f(X) \notin G] \leq w \cdot \frac{\varepsilon}{w} \leq \varepsilon$. Putting everything together,

$$\begin{aligned}\sum_{v \in [w]} TV(\mathcal{U}_m, \mathrm{Ext}(X_v, Z)) \cdot \Pr[f(X) = v] &\leq \sum_{v \in G} TV(\mathcal{U}_m, \mathrm{Ext}(X_v, Z)) + \Pr[f(X) \notin G] \\ &\leq 2\varepsilon.\end{aligned}$$

□

We now construct Nisan's PRG as follows. Let $\mathrm{Ext} : \{0,1\}^{jd} \times \{0,1\}^d \to \{0,1\}^{jd}$ be a family of $(jd - \log(w/\varepsilon), \varepsilon)$-seeded extractors, for $j = 1, \ldots, \log n$. Define a sequence of PRGs $G_j : \{0,1\}^{jd} \to \{0,1\}^{2^j}$ recursively by

$$\begin{aligned}&G_1(x, z) = z \\ &G_j(x, z) = G_{j-1}(x) \circ G_{j-1}(\mathrm{Ext}_{j-1}(x, z)) \text{ for } j > 1.\end{aligned}$$

We will show by induction on j that G_j ε_j-fools every length-2^j, width-w ROBP B for $\varepsilon_j \leq 4^j \cdot \varepsilon$. We do this by a hybrid argument. Consider the following distributions:

$$\begin{aligned}&D_0 = X \circ X' \text{ for } X, X' \sim \mathcal{U}_{2^{j-1}} \\ &D_1 = X \circ G_{j-1}(X') \text{ for } X \sim \mathcal{U}_{2^{j-1}}, X' \sim \mathcal{U}_{(j-1)d} \\ &D_2 = G_{j-1}(X) \circ G_{j-1}(X') \text{ for } X, X' \sim \mathcal{U}_{(j-1)d} \\ &D_3 = G_{j-1}(X) \circ G_{j-1}(\mathrm{Ext}(X, Z)) \text{ for } X \sim \mathcal{U}_{(j-1)d}, Z \sim \mathcal{U}_d\end{aligned}$$

Hybrids D_0 and D_1 are $4^{j-1}\varepsilon$-indistinguishable by the inductive hypothesis, as are hybrids D_1 and D_2.