CS 880: Pseudorandomness and Derandomization 3/20/2013

Lecture 17: Randomness Extractors

Instructors: Holger Dell and Dieter van Melkebeek Scribe: Mahnaz Akbari

DRAFT

This lecture is about randomness extractors. Extractors are functions that map samples from a non-uniform distribution to samples that are close to being uniformly distributed. The length of the output will in general be smaller than the length of the input of the extractor.

The input distribution of the extractor is called the *source*. A source is a random variable which maps text values to bit strings. The goal is to extract randomness from such sources.

Definition 1. *Let (Ω, p) be probability space, that is, Ω is a finite set and $p \in [0,1]^\Omega$ is a vector with $\|p\|_1 = 1$. A* source *X over $\{0,1\}^n$ is a $\{0,1\}^n$-valued random variable over some space (Ω, p), that is, a function*

$$X \ : \ \Omega \ \to \ \{0,1\}^n \, .$$

We recall the definition of an extractor from Lecture 2, where we gave a general overview.

Definition 2 (Extractor). *Let C be a class of sources over $\{0,1\}^n$. A function $E : \{0,1\}^n \to \{0,1\}^r$ is a (deterministic) ϵ-extractor for C if, for all sources $X \in C$, we have*

$$d_{\text{stat}}\Big(U_r \ , \ E(X)\Big) \leq \epsilon \, .$$

We give an example of a source an how to extract randomness from it.

Example 1. *Let $X = X_1 X_2 \ldots X_n$ be a random variable on $\{0,1\}^n$ so that the bits X_i are identically and independently distributed with $P(X_i = 1) = \sigma$. So the source is not uniform if $\sigma \neq \frac{1}{2}$ To extract a uniform distribution from this source, one can partition X into strings of length two. Let us consider the first pair of bits, namely X_1 and X_2. Then we observe that the events $X_1X_2 = 01$ and $X_1X_2 = 10$ have the same probability:*

X_1	X_2	Pr	*E outputs*
0	*1*	$\sigma(1-\sigma)$	*1*
1	*0*	$\sigma(1-\sigma)$	*0*
0	*0*	σ^2	*discard*
1	*1*	$(1-\sigma)^2$	*discard*

Since the first two events happen with the same probability, they can be used to produce a single unbiased bit. In case the bits happen to be 00 *or* 11*, we can't extract unbiased bits directly, and we simply move on to the next pair of bits. On average, the number of bits that this procedure produces is the number of pairs for which one of these two events occurs. Thus, the expected length of the output of the extractor is $\mathbb{E}(r) = 2\sigma(1-\sigma)\frac{n}{2}$.*

The example above leads us to the question of how much randomness is contained in a source. In the extreme case where X is a constant, there is no hope to extract any randomness. On the other hand, if X is a source on $\{0,1\}^n$, we cannot expect to be able to extract more than n uniform bits. In the example above, the amount of randomness that we could extract was a constant fraction of n.

1 Entropy measures

Entropy measures the amount of randomness contained in a random source. There are various notions of entropy, two of which we will discuss now.

Definition 3 (Shannon Entropy of X). *Let X be a random variable. The* Shannon entropy *of X is the number*

$$H(X) \doteq \mathbb{E}_{x\sim X} \underbrace{\left(\log \frac{1}{\Pr[X=x]}\right)}_{\text{“amount of information contained in } x\text{”}} = \sum_x \Pr[X=x] \cdot \left(\log \frac{1}{\Pr[X=x]}\right).$$

One way to look at the part that shows the amount of information in the above formula is that it is, intuitively, the number of bits required to store the information in x, when the distribution X, from which x was sampled, is know to the receiver. Then the expectation is the average number of bits needed to store a sample from X. In other words, given a sample from X, how far it can be compressed. This intuition is made precise in Shannon's noisy-channel coding theorem.

If X is uniformly distributed, the entropy of X is equal to the dimension of X and the identity map is a perfect extractor. Furthermore, the closer the Shannon entropy is to the dimension of X, the closer X must be to the uniform distribution, so it is reasonable to suspect that a high Shannon entropy allows us to extract randomness that is close to uniform. The following example refutes this intuition for the Shannon entropy.

Example 2. *Let X be a random variable on $\{0,1\}^n$ with*

$$\Pr[X=x] = \begin{cases} \frac{1}{2} & \text{if } x = 0^n, \\ \frac{1}{2}\cdot\frac{1}{2^{n-1}} & \text{otherwise.} \end{cases}$$

This random variable is equal to 0^n with probability half, and it is uniformly distributed on the rest of the space. Then the Shannon entropy is with $H(X) \approx \frac{n}{2}$ fairly close to maximal, but we also have the following property for all function E:

$$\Big|\underbrace{\Pr\left(E(X)=E(0^n)\right)}_{\geq\frac{1}{2}} - \underbrace{\Pr\left(U_r=E(0^n)\right)}_{2^{-r}}\Big| \approx \frac{1}{2}$$

This implies that the statistical distance between $E(X)$ is close to 0.5 and therefore far away from 0 unless r is trivially small. This means that there does not exist an extractor for this random variable.

The example above shows that one can define a random variable X which has a high Shannon entropy, but from which no uniform randomness can be extracted with an extractor. Thus, Shannon entropy is insufficient to measure the extractable randomness contained in a random variable, for which reason we will use a different notion of entropy. The problem with the above example was that there was an outcome that individually occurs with large probability. The Min-entropy defined below yields an upper bound on the largest probability for individual outcomes.

Definition 4 (Min-entropy of X). *Let X be a random variable. The* Min-entropy *of X is the number*

$$H_\infty(X) \doteq \min_x \log \frac{1}{\Pr[X = x]} = \log \frac{1}{\max_x \Pr[X = x]} .$$

Equivalently, $H_\infty(X)$ is the largest number k such that all outcomes have probability at most 2^{-k}.

In the example above, we had $\max_x\{\Pr[X = x]\} = \frac{1}{2}$ and thus $H_\infty(X) = 1$, which is very small compared to the maximum possible value of n. Thus, the Min-entropy may be small even though the Shannon entropy is large. The following lemma shows the relationship between two entropies in general.

Lemma 3 (Relationship Between Entropies.). *Let X be a random variable and let* $\mathrm{supp}(X) \doteq \{x | \Pr[X = x] > 0\}$ *be the* support *of X. Then we have*

$$0 \le H_\infty(X) \le H(X) \le \log \left|\mathrm{supp}(X)\right| . \tag{1}$$

The last inequality in (1) means that a sample of X can always be stored using at most $\log |\mathrm{supp}(X)|$ bits. The Shannon-entropy may be smaller than that, which means that, on average, fewer bits are needed to store a sample from X. The Min-entropy can not be larger than that, and it is the number of uniform random that can be extracted from samples of X.

Lemma 4. *Let X and Y be independent random variables. Then both entropies are additive:*

$$H(X, Y) = H(X) + H(Y) ,$$
$$H_\infty(X, Y) = H_\infty(X) + H_\infty(Y) .$$

The additivity of the Shannon entropy says that, in order to store a sample from X and an independent sample from Y, we need to store both samples individually. For the min-entropy, the intuition is that the amount of randomness extractable from X and Y is equal to the sum of the amount of randomness extractable from these random variables individually.

We use random variables X as a source of (possibly non-uniform) randomness.

Definition 5. *A k-source is a random variable X with $H_\infty(X) \ge k$.*

As we will see later, one can extract roughly k uniform bits from a k-source. Let us first look at some examples of k-sources.

Example 5 (Bit-fixing sources). *A random variable X on $\{0,1\}^n$ is a* bit-fixing source *if*

- *k bits of X are uniformly distributed*
- *$n - k$ bits of X are fixed; these bits are deterministic function of the uniform k bits.*

A special case of bit-fixing sources occurs when the fixed bits are all fixed to 0.

Example 6 (Flat k-sources). *A random variable X on $\{0,1\}^n$ is a flat k-source if there is a set $S \subseteq \{0,1\}^n$ with $|S| = 2^k$ such that*

$$\Pr[X = x] = \begin{cases} \frac{1}{2^k} & \text{if } x \in S, \\ 0 & \text{otherwise.} \end{cases}$$

We also write X_S for the flat source.

In both of the cases above, the min-entropy is $H_\infty(X) = k$. Actually, *every* k-source is a convex combination of flat k-sources.

Lemma 7. *Every k-source X is a convex combination of flat k-sources, that is,*

$$\Pr[X = x] = \sum_S c_S \cdot \Pr[X_S = x]$$

holds for some coefficients c_S with $\sum_S c_S = 1$ and $c_S \geq 0$.

This lemma will enable us to reduce the extraction from an arbitrary k-source to the extraction from a flat k-source.

Exercise 1. *Prove Lemma 7.*

2 Deterministic extractors

The following lemma shows that every deterministic extractor as defined in Definition 2 is bad for some source.

Lemma 8. *For all functions $E : \{0,1\}^n \to \{0,1\}$, there is a flat $(n-1)$-source so that $E(X)$ is a constant function.*

In particular, every extractor has a source of very high entropy for which not even a single bit can be extracted.

Proof. Let $b \in \{0,1\}$ so that the number of $x \in \{0,1\}^n$ such that $E(x) = b$ holds is at least $2^n/2 = 2^{n-1}$. Such b exists by the pigeon hole principle. Now let X be the flat source whose support consists of those x for which $E(x) = b$ holds. Then X is a flat $(n-1)$-source and $E(X)$ is always equal to b. □

On the other hand, the following lemma shows that, for every k-source, *most* deterministic extractors work.

Lemma 9. *For all k-sources X on $\{0,1\}^n$ and $\epsilon > 0$ we have*

$$\Pr_E[\textit{random function } E \textit{ is an } \epsilon\textit{-extractor for } X] \geq 1 - 2^{-\Omega(2^k \epsilon^2)}$$

where $r = k - 2\log\frac{1}{\epsilon} - O(1)$ and $E : \{0,1\}^n \to \{0,1\}^r$.

The lower bound for the probability is very close to one, which means that a random function is an extractor for the flat k-source X with high probability.

Exercise 2. *Prove Lemma 9 using the Chernoff bound.*

3 Seeded Extractors

We have seen that deterministic extractors cannot be used for all k-sources, and fully random extractors are good for all k-sources. With our goal of saving randomness in mind, we want to avoid using fully random extractors. *Seeded extractors* are a compromise between the two extremes. They assume to have access to a little bit of perfectly uniform randomness, but the amount is only logarithmic in the length of the source.

Definition 6. *A function $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^r$ is a (k,ϵ)-*extractor *if, for all k-sources X on $\{0,1\}^n$, we have $d_{\text{stat}}\big(E(X,U_d),U_r\big) \le \epsilon$.*

The following theorem shows that seeded extractors exist, and the proof uses the probabilistic method.

Theorem 10. *For all positive integers n and $k \le n$, and all $\epsilon > 0$, there exists a (k,ϵ)-extractor $E : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^r$ with $r = k - 2\log\frac{1}{\epsilon} - O(1)$ and $d = \log(n-k) + 2\log\frac{1}{\epsilon} + \Omega(1)$.*

Because the k-source X and the uniform distribution U_d are independent, their min-entropies add up, for which reason the overall entropy of the input of E is $k+d$. The smaller ϵ is, the less randomness r can be extracted.

Proof (Idea). Let E be sampled uniformly at random from the set of all functions of the specified type. We want to argue that E fails to be an extractor with probability strictly less than one. Let X be a k-source. By Lemma 7, we know that X can be written as a convex combination of flat k-sources. Thus, if X is an extractor for all flat k-sources, it is an extractor for X. For this, we use the union bound, and the probability of E not being an extractor for X is at most a factor of at most $\binom{2^n}{2^k}$ larger than the probability of E not being an extractor for a flat k-source, since that is the number of all flat k-sources on $\{0,1\}^n$. Formally, we have

$$\begin{aligned}\Pr_E\big[E \text{ is not a } (k,\epsilon)\text{-extractor}\big] &\le \binom{2^n}{2^k} \cdot \max_{\text{flat } k\text{-source } F} \Pr_E\big[E \text{ is not an } \epsilon\text{-extractor for } F\big] \\ &\le \Big(2^{n-k}e\Big)^{2^k} \cdot 2^{-\Omega\left(2^k 2^d \epsilon^2\right)} < 1\end{aligned}$$

The second inequality follows from $\binom{N}{K} \le (Ne/K)^K$ and Lemma 9 using the fact that (F,U_d) has min-entropy $k+d$, and the last inequality follows from the choice of parameters. □

4 Motivation: Derandomization

Extractors can be used to reduce the amount of randomness required by randomized algorithms. Let $A(x,\rho)$ be a randomized algorithm so that, for all inputs x, we have

$$\Pr\big(A(x,U_r) \text{ fails}\big) \le \delta\,,$$

and let E be a (k,ϵ)-extractor. Then we define a derandomized algorithm by generating the randomness using E and taking a majority vote over the extractor seed:

$$A'(x,\rho') \doteq \text{maj}_{s\in\{0,1\}^d} A\big(x,E(\rho',s)\big)\,.$$

The main advantage of this approach is that $A'(x, Y)$ does not need a uniform source Y anymore; a k-source suffices. Then we have, for all inputs x,

$$\Pr(A'(x, Y) \text{ fails}) \leq 2(\delta + \epsilon)\,.$$

This allows to simulate A when there is a little uniform randomness and access to large string of not necessarily uniform randomness like key strokes.

Lemma 11. *For all functions f so that $A(x, U_r)$ computes $f(x)$ with probability at least $1 - \delta$, we have that $A'(x, Y)$ computes $f(x)$ with probability at least $1 - 2(\epsilon + \delta)$ for all k-sources Y on $\{0,1\}^n$.*

Proof. We consider the statistical distance and use the triangle inequality:

$$\begin{aligned} d_{\text{stat}}\big(A(x, E(x, U_d)), f(x)\big) &\leq d\big(A(x, E(x, U_d)), A(x, Ur)\big) && (\leq \epsilon \text{ by extractor property}) \\ &\quad + d\big(A(x, U_r), f(x)\big) && (\leq \delta \text{ by failure probability}) \\ &\leq \delta + \epsilon \end{aligned}$$

Then we use Markov's inequality to compute the probability that the majority vote fails:

$$\Pr_{\rho' \sim Y}\Big[\geq \frac{1}{2} \text{ of all } s \text{ are bad}\Big] \leq 2(\epsilon + \delta)\,. \qquad \square$$

The algorithm A' runs in time 2^d times the running time of A plus the running time of E. Thus, while Theorem 10 guarantees that a seeded extractor with good parameters exists, we don't have it explicitly and we don't know how efficiently it can be computed.

Next time, we will see how to construct seeded extractors explicitly by using their strong relationship with bipartite expanders.