Lecture Notes
Randomness extraction

S. Wehner

October 22, 2012

1 Introduction

Many sources of weak randomness exist, but even a single perfect random coin is hard to come by. Today we will learn how to obtain such (almost) perfect randomness from weak randomness!

Consider again the "key making machine" - upon the push of a button it gave us a random string k chosen with some probability $P_K(k)$. When talking about randomness, we commonly refer to such a "machine" as a *source* of randomness, where randomness refers to a string $k\{0,1\}^n$ chosen according to some distribution $P_K(k)$. A *perfect* source of randomness has the property the strings k are chosen uniformly and independently of any other processes. That is, $P_K(k) = 1/2^n$. Most sources, however, do not have this property. Indeed, true randomness is an extremely rare resource in nature and very hard to come by. The goal of randomness extraction is to find a function $Ext : \{0,1\}^n \to \{0,1\}^m$ such that $\vec{\rho}_{Ext(K)} \approx_\varepsilon \text{unif}(\{0,1\}^m)$, possibly investing some perfect randomness to start with. The function Ext is also called a *randomness extractor* or simply *extractor*.

Extractors play an important role for obtaining randomness for use in e.g. algorithms, but also cryptography. Indeed, we will use them as a tool in many cryptographic protocols later on. To gain some intuition about randomness extraction, let us first consider two very simple sources of randomness.

1.1 IID Sources

The simplest case of a randomness source is the so-called i.i.d. or von Neumann source. The term i.i.d. stands for independently and identically distributed. Imagine a source such that every time we use the source it will emit a k with the same probability $P_K(k)$ independently of what happened in the past. As an example, imagine a source that emits a single bit $k \in \{0,1\}$ every time we use it. Clearly, if we use this source n times, we effectively get a new source - now for an entire string $x = x_1, \ldots, x_n\{0,1\}^n$ where

$$P_X(x) = P_{X_1}(x_1) \cdot P_{X_2}(x_2) \ldots P_{X_n}(x_n) P_K(x_1) \cdot P_K(x_2) \ldots P_K(x_n) \ . \tag{1}$$

Such a source in which each bit is chosen independently and with the identitical distribution is called an *IID source*. Such sources are sometimes also called *von Neumann* sources, since they were already considered by von Neumann. If you are curious about the history of randomness extraction, go look up the von Neumann extractor online!

As a warmup, let's consider how we could obtain a nearly random bit from the following source: $P_K(0) = 1/4$ and $P_K(1) = 3/4$, used n times to give a string x as described above. Do you have any ideas what we might do?

Let's suppose we let $B = Ext(X) := X_1 \oplus X_2 \oplus \ldots X_n$ be the XOR of all the n bits in the string. Our goal is to show that $\vec{\rho}_B \approx_\varepsilon \text{unif}(\{0,1\})$ for reasonably small ε.

- Let's suppose that $n = 2$. How well does our strategy work? Note that we have $P_B(0) = P_K(0)^2 + P_K(1)^2 = 1/16 + 9/16 = 10/16 = 0.625$ and $P_K(1) = 0.375$. Not quite uniform but closer to 50-50 then we started with!

- It turns out that for large n, ε does indeed become exponentially small in n. It's a good exercise to convince yourself of this fact!

1.2 Independent bit sources

A slightly refined version of the von Neumann source is the so-called *independent bit* source. As the name suggest it has the property that each bit is chosen independently of what happened before. However, the distribution no longer has to be identical for each bit. In other words, the distribution over strings $x \in \{0,1\}^n$ is of the form

$$P_X(x) = P_{X_1}(x_1)P_{X_2}(x_2)\ldots P_{X_n}(x_n) \ , \tag{2}$$

however, it's possible that $P_{X_1}(0) \neq P_{X_2}(0)$ and so forth. Clearly, any IID source is also an independent bit source, but the converse does not hold.

It turns out that taking the XOR of all the bits in the string still works, in the sense that $\vec{\rho}_B$ will be close to uniform as long as $P_{X_j}(0) = 1/2 \pm \delta$ for some $1/2 > \delta > 0$, i.e. the individual bits are at least somewhat unpredictable. Again, if you are curious about randomness extraction, it's a good exercise to convince yourself of this fact.

1.3 Deterministic randomness extraction

In the example above, we applied a fixed function Ext to the string X - namely the XOR of all its individual bits. Such a function is also known as a deterministic extractor, meaning that is it just one fixed function that itself does not depend on a random choice.

Definition 1.1. *A (k,ε)-deterministic extractor is a function $Ext : \{0,1\}^n \to \{0,1\}^m$ such that $\vec{\rho}_{Ext(X)} \approx_\varepsilon \text{unif}(\{0,1\}^m)$ whenever $H_{\min}(X) \geq k$.*

2 General sources

Our simple example of randomness sources above all had one thing in common: they gave us a string in which each bit was chosen *independently*. What if we relax this condition? For example, what if we only know that $H_{\min}(X) \geq k$ for the string X that we get from the source? Note that this is equivalent to asking that for all x we have $P_X(x) \leq 1/2^k$.

Definition 2.1. *A random variable X is a k-source if $H_{\min}(X) \geq k$.*

It turns out that there is no deterministic extractor of any kind that can work without further guarantees of the source. I.e., without depending in some way of our knowledge of the distribution P_X. In fact, we cannot obtain even a *single* bit of randomness even if we had nearly full min-entropy to start with! Let's try and prove this crucial fact.

Lemma 2.2. *For any randomness extractor* $Ext : \{0,1\}^n \to \{0,1\}$ *there exists an* $n-1$*-source such that* Ext *is constant.*

Proof. Define the bit $b \in \{0,1\}$ such that $|S| \geq 2^n/2 = 2^{n-1}$ with $S = \{x \mid Ext(x) = b\}$. Note that there must exist such a b. Choose a subset $S' \subseteq S$ such that $|S'| = 2^{n-1}$. Consider now the source defined by the following probability distribution

$$P_X(x) = \begin{cases} 1/2^{n-1} & \text{if } x \in S' \ , \\ 0 & \text{otherwise} \ . \end{cases} \tag{3}$$

Clearly, $H_{\min}(X) = n-1$. However, note that $Ext(X) = b$ is a constant! □

At first glance, we might think that this ends our discussion right here. It seems we cannot hope to obtain any randomness except from very specialized sources, where we happen to know the distribution P_X and can tailor our extractor to this distribution. In general, however, we would like to extract randomness without knowing such details, and merely rely on a guarantee that the source has high min-entropy.

Luckily, however, it turns out that there is a way out, when we allow ourselves a small amount of randomness to start with. That is, we have some initial amount of perfect randomness $r \in 0,1^\ell$ also known as the *seed*. Given such a seed, we can extend our definition of extractors. Extractors that take a seed are also called *seeded extractors*.

Definition 2.3. *A* (k,ε)*-randomness extractor is a function* $Ext : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^m$ *such that* $\vec{\rho}_{Ext(X,R)} \approx_\varepsilon \text{unif}(\{0,1\}^m)$ *whenever* $H_{\min}(X) \geq k$.

In cryptography, as well as many other applications, we want to extend this definition even further and demand that the output is uniform even for an adversary holding from side information E about X. For example, E could simply be some information about what the source has done in the past, or some information that an adversary has gathered during the course of a protocol that is producing randomness. To see this, let's think back to our example last week! We thus more generally define

Definition 2.4. *A* (k,ε)*-randomness extractor is a function* $Ext : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^m$ *such that* $\vec{\rho}_{Ext(X,R)E} \approx_\varepsilon \text{unif}(\{0,1\}^m) \otimes \vec{\rho}_E$ *whenever* $H_{\min}(X|E) \geq k$.

Note that our previous definition is a special case of this one for the case that E is trivial, i.e. not there. Recall from lecture 1 that our definition says that any adversary E does not know anything about $Ext(X,R)$.

2.1 Strong extractors

When investing a seed R to obtain randomness from the source, you may rightfully object: why should we be surprised that randomness can be obtained this way? After all, we are feeding randomness in, and one way to get random bits out would be to simply output the seed R! Luckily,

it can be shown that there do indeed exist extractors such that the length of the seed is $|R| = \log(n-k) + 2\log(1/\varepsilon) + O(1)$ and output side $m = k + |R| - 2\log(1/\varepsilon) + O(1)$ (see my upload to IVLE ot the survey by Shaltiel). This is reassuring as the output can thus be longer than the seed that we invested.

Yet, especially when it comes to cryptographic applications this is still not satisfying, but we want something even stronger.

Definition 2.5. *A* (k, ε)-strong *randomness extractor is a function* $Ext : \{0,1\}^n \times \{0,1\}^\ell \to \{0,1\}^m$ *such that* $\vec{\rho}_{Ext(X,R)RE} \approx_\varepsilon \mathrm{unif}(\{0,1\}^m) \otimes \mathrm{unif}(\{0,1\}^\ell) \otimes \vec{\rho}_E$ *whenever* $H_{\min}(X|E) \geq k$.

In other words, we want a *strong* extractor which has the property that the output $Ext(X,R)$ *and* the seed are jointly uniform and uncorrelated from E. Why is this interesting? First of all, note that this means that the output is indeed (almost) independent from the seed - i.e. fresh randomness has been produced.

Most importantly, however, it means that even if an attacker later learns the seed R he still does not know $Ext(X,R)$ since the definition implies that

$$\vec{\rho}_{Ext(X,R)RE} = \mathrm{unif}(\{0,1\}^m) \otimes \vec{\rho}_{RE} \ . \tag{4}$$

I.e. even an attacker holding E *and* R is ignorant about $Ext(X,R)$.

2.2 Fundamental limitations

Before we consider a construction of extractors, let us consider one of its fundamental limitations: If $H_{\min}(X|E) \geq n - t$ for an n-bit string X and some $t \geq 0$, how much randomness can a strong extractor give us? It turns out that $H_{\min}(X|E)$ is indeed the maximum number of bits we can hope to obtain[1]

To see why this is the case, let us establish another useful fact about the min-entropy, which we will argue informally: Recall that $H_{\min}(X|E) = -\log P_{\mathrm{guess}}(X|E)$. Suppose now that we apply some function f to X. How hard is it to guess $f(X)$ given E, i.e., what's $P_{\mathrm{guess}}(f(X)|E)$? Clearly, since one way to guess $f(X)$ is to guess X, and then apply f to our guess, we have $P_{\mathrm{guess}}(f(X)|E) \geq P_{\mathrm{guess}}(X|E)$. However, this is equivalent to

$$H_{\min}(f(X)|E) \leq H_{\min}(X|E) \ . \tag{5}$$

However, note that this means that also the extractor, which for fixed R is just a function f yielding $f(X) = Ext(X,r)$ must have min-entropy less than $H_{\min}(X|E)$. However, this implies that the output $Ext(X,R)$ can be uniform on at most $H_{\min}(X|E)$ bits!

2.3 Constructing strong extractors

Much research has gone into constructing randomness extractors, and their study has beautiful relations to other areas of computer science and physics. Typically, the challenge is to obtain as much randomness as possible, while minimizing the size of the seed. Here, we will not survey all known constructions but focus on one particularly easy example. This example has a very large seed size, however, it is relatively easy to prove (indeed, you will do so in your homework!).

To this end, we will consider a very special class of functions defined by Carter and Wegman.

[1] Without outputting the seed, but note that this is not allowed in a strong extractor!

Definition 2.6. *A family of hash functions* $\mathcal{F} = \{f : \{0,1\}^n \to \{0,1\}^m\}$ *is* 2-universal *if for every two strings* $x, x' \in \{0,1\}^n$ *with* $x \neq x'$ *we have*

$$\Pr_{f \in \mathcal{F}}[f(x) = f(x')] \leq \frac{1}{2^m} \ . \tag{6}$$

It can be shown that the set of all possible functions $f : \{0,1\}^n \to \{0,1\}^m$ is 2-universal. However, much smaller sets exists and you can find some well known constructions on e.g. wikipedia. What does this have to do with our problem?

Note that we can now think of the seed as selecting a random function from the set of functions $\mathcal{F}$. That is, $Ext(X, r) = r(X)$ where we will use r to denote the choice of function $f = r$ from the set $\mathcal{F}$. A central result, first proven by Impagliazzo, Levin, and Luby is the so-called *Leftover hash lemma*[2]. It's great, because if tells us that with a sufficiently large seed, we can indeed obtain $H_{\min}(X|E)$ bits of fresh randomnesss! In your homework, we'll walk through its proof.

Theorem 2.7 (Leftover hash lemma)**.** *Let* $\mathcal{F} = \{f : \{0,1\}^n \to \{0,1\}^m\}$ *be a* 2*-universal set of hash functions, with* $m = k - 2\log(1/\varepsilon)$*. Define* $Ext : \{0,1\}^n \times \mathcal{F} \to \{0,1\}^m$ *as* $Ext(X, r) := r(x)$*. Then* Ext *is a* (k, ε)*-strong randomness extractor.*

2.4 Privacy amplification

Let's now return to our little key generation scenario from last week. Can you see how we can extend our protocol to allow Alice and Bob to generate a secure key?

Clearly, our protocol had a good start: Alice can generate a random string $x_1, \ldots, x_n$ and input it into the device. The device will output $x_1, \ldots, x_n$ to Bob, and (unknown to Alice and Bob) give each bit x_j with probability p to any eavesdropper. Let's compute the probability that the eavesdropper learns x: For each bit, Eve either gets it with probability p, or else she can make a random guess which is correct with probability $1/2$. This means that the probability that Eve gets the bit is $q = p + (1-p)/2$ and hence $P_{\text{guess}}(X|E) = q^n$. Hence $H_{\min}(X|E) = n(-\log q)$. Now, Alice can choose a hash function $r \in \mathcal{F}$ from a set of 2-universal functions and compute $k = r(X)$. Alice takes k as her key. Now, Alice sends r to Bob, who can also compute $r(X) = k$. The leftover hash lemma now implies that Eve knows little about the key *even* though she knows the hash function r.

[2]The lemma extends to so-called pairwise independent hash functions, but we won't consider them here.