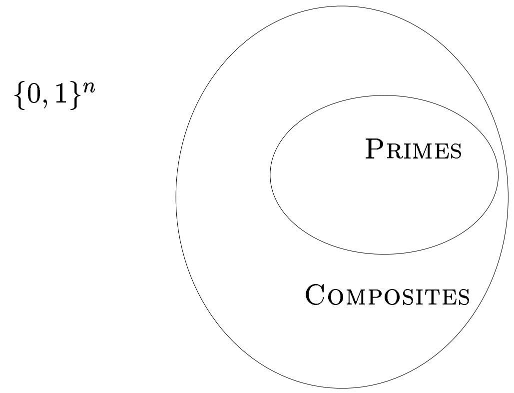
CS395T Pseudorandomness and Combinatorial Constructions
Lecture 15, March 6, 2001
Lecturer: Prof. David Zuckerman, Scribe: Jean-Philippe Martin

1 Extracting Randomness

This lecture focuses on pseudorandomness and algorithms, and partially follows the slides in the presentation titled "Extractors for Weak Random Sources and their Applications". Illustrations from the presentation have been copied into this document to make it easier to understand. We will start with a motivating example.

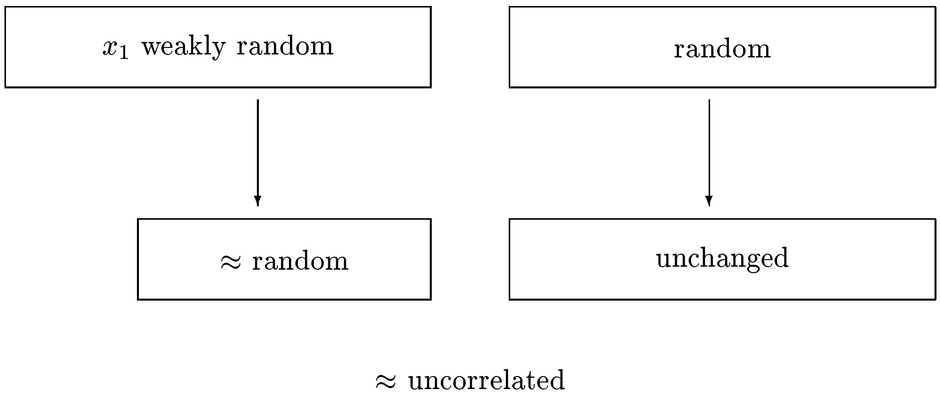
1.1 PRIMES

The goal is to generate a random n-bit prime. We assume that we have a deterministic primality test. Since $\Omega(1/n)$ of n-bit numbers are prime, the naive approach consists in picking random numbers until one of them is prime. This in average requires $O(n)$ tries, so $O(n^2)$ random bits.



A way to improve the situation is that if our first try x_1 is a composite, we can exploit the fact that x_1 is a *random* composite. So for our next try, instead of creating a new random number from scratch we extract the randomness left in x_1.

Method: Add random catalyst.



The randomness extracted from x_1 and the random number c used as a catalyst must be uncorrelated. Otherwise, for example, the extracted randomness could simply be a copy of c. The lack of correlation will be useful when we use c repeatedly.

Note that the randomness extracted from x_1 is smaller than x_1: we gained in randomness at the cost of some length.

To extract randomness, we will use a hash function.

2 Hash Functions

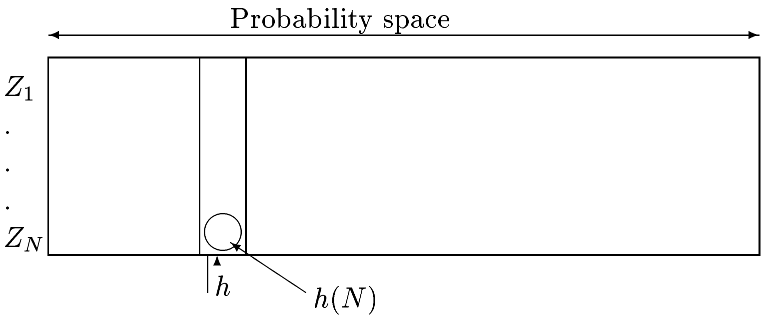
Let $N = |\{0,1\}^n|$ and $M = |\{0,1\}^m|$, $m \leq n$.

Definition 1 $H = \{h : \{0,1\}^n \mapsto \{0,1\}^m\}$ *is a* universal family of hash functions *iff:*

$$(\forall x \neq y) \Pr_{h \in_u H}[h(x) = h(y)] \leq 1/M$$

Pairwise Independence

The probability space Ω of pairwise independent random variables $Z_1 \ldots Z_N$ in $\{0,1\}^m$ can be seen as a family of hash functions as follows. Each element $\omega \in \Omega$ corresponds to a hash function h_ω, and each argument x to the hash function corresponds to a random variable Z_x. Namely, $h_\omega(x)$ is defined to be the value of the random variable Z_x at ω.



We have

$$\Pr[h(x) = h(y)] = \Pr[Z_x = Z_y]$$

Because Z is pairwise independent, $\Pr[h(x) = h(y)] \leq 1/M$ (M is the range of the random variable). Going through with the computation actually shows that $\Pr[h(x) = h(y)] = 1/M$.

One way to compute a pairwise independent string (as you recall from a previous lecture) is to use the family of functions $h(x) = ax + b$ with all operations in $\mathbb{F}_{2^n}$. Because of what we have seen above, these functions also constitute a universal hash family. In fact, the function family h' which takes only the last m bits of h also defines a universal family of hash functions because $\Pr[h'(x) = h'(y)] =$ (number of pairs of n-bit string whose last m bits are identical)/(number of pairs of n-bit strings) $= \frac{2^m 2^{2(n-m)}}{2^{2n}} = 2^{-m} = 1/M$. A final optimization comes from the realization that the value b is added to both sides of the equality and therefore does not contribute.

Example Family of Hash Functions

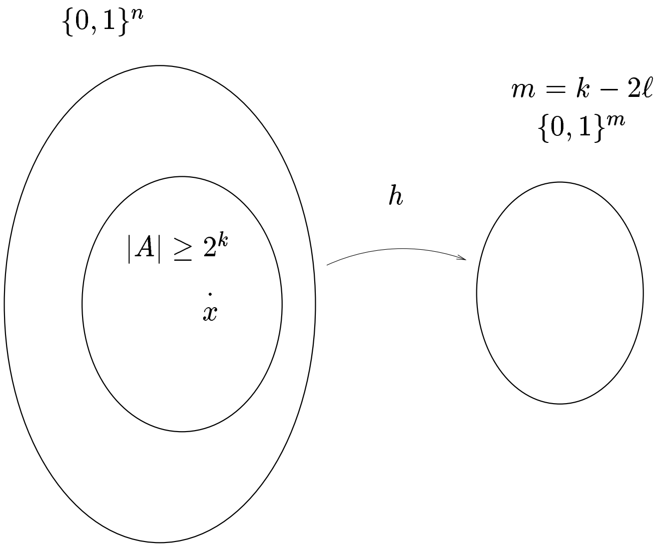
We can now define the following family of hash functions, in $\mathbb{F}_{2^N}$.

$$h_a(x) = \text{ last } m \text{ bits of } (a \cdot x)$$

$$|H| = 2^n$$

3 Leftover Hash Lemma

For $x \in_u A$, $h \in_u H$ the distribution of $h \circ h(x)$ is $2^{-\ell}$-uniform ($\circ$ denotes concatenation).



Definition 2 *The collision probability of a distribution D is*

$$cp(D) \doteq \Pr_{d_1, d_2 \in D}[d_1 = d_2]$$

We slightly abuse the $d_1 \in D$ notation to mean "pick d_1 randomly according to the D distribution".

Proof:

The proof of the Leftover Hash Lemma has two steps. We'll first show that $cp(h \circ h(x))$ is small and then we will show that this is sufficient.

$$\begin{aligned}
cp(h \circ h(x)) &\leq \frac{1}{|H|}(\Pr[h_1(x_1) = h_2(x_2) | h_1 = h_2, x_1 = x_2] \Pr[x_1 = x_2] \\
&\quad + \Pr[h_1(x_1) = h_2(x_2) | h_1 = h_2, x_1 \neq x_2] \Pr[x_1 \neq x_2]) \\
&\leq \frac{1}{|H|}\left(\frac{1}{|A|} + \frac{1}{M}\left(1 - \frac{1}{|A|}\right)\right) \\
&\leq \frac{1}{|H|M}\left(1 + \frac{M}{|A|}\right)
\end{aligned}$$

We chose that formulation because $h_1(x)$ is a distribution on a set of size $|H|M$. This shows that $cp(h \circ h(x))$ is a little larger than the inverse of the set size. In the next step, we show that this implies that $h \circ h(x)$ is close to uniform.

Claim: Let D be a distribution on S. $cp(D) < \frac{1+4\epsilon^2}{|S|} \implies |D - U| < \epsilon$ (U is the uniform distribution).

Proof: (by contradiction). Suppose that this implication is false, i.e. it can be the case that $|D - U| \geq \epsilon$ and $cp(D) < \frac{1+4\epsilon^2}{|S|}$. The first equation implies that $\exists T \subseteq S, p \doteq D(T) \geq \frac{|T|}{|S|} + \epsilon$. We now compute $cp(D)$ and show that it contradicts the second equation.

$$\begin{aligned} cp(D) &\geq \Pr_{s_1,s_2 \in D}[s_1 \in T \wedge s_2 \in T]\frac{1}{|T|} + \Pr_{s_1,s_2 \in D}[s_1 \notin T \wedge s_2 \notin T]\frac{1}{|S|-|T|} \\ &\geq p^2\frac{1}{|T|} + (1-p)^2\frac{1}{|S|-|T|} \end{aligned}$$

this expression is minimized for $p = \frac{|T|}{|S|} + \epsilon$. Therefore

$$\begin{aligned} cp(D) &\geq \left(\frac{|T|}{|S|} + \epsilon\right)^2 \frac{1}{|T|} + \left(1 - \frac{T}{S} - \epsilon\right)^2 \frac{1}{|S|-|T|} \\ &\overset{\text{algebra}}{\geq} \frac{1+4\epsilon^2}{|S|} \end{aligned}$$

□

Therefore the distribution of $h \circ h(x)$ is $2^{-\ell}$-uniform.

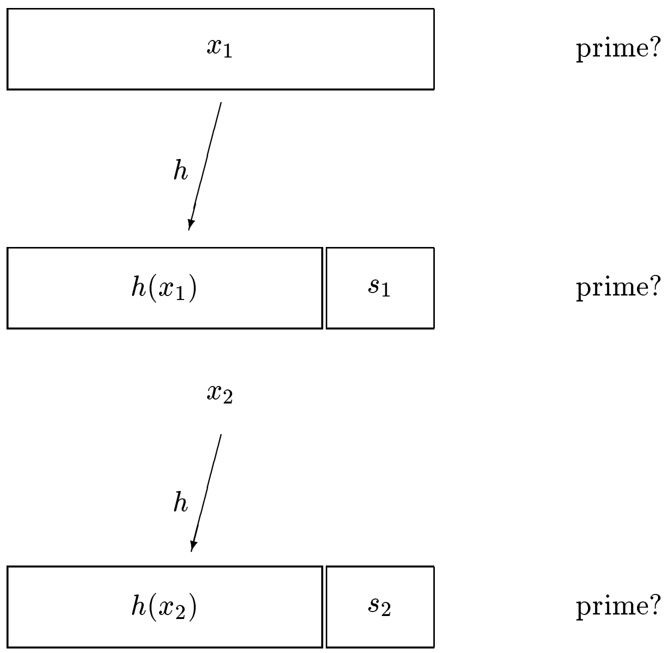
□

4 Example 1: Primes

Using the Leftover Hash Lemma, we can now improve on the PRIME example shown before. Remember that the goal is to generate a random n-bit prime and that we assume that we have a deterministic primality test.

The algorithm starts out choosing a random number x_1 and testing its primality. If the number is not prime then instead of using a completely new random number for our second try, we hash x_1 and add fresh random bits, s_1, to obtain our new random number. The hash function is chosen randomly at the beginning of the algorithm; the same function is then used on each number.

Pick $h \in_U H$.



This new algorithm uses in average $O(n \log n)$ random bits.

5 Example 2: Amplification of BPP

Recall that a BPP algorithm uses r random bits and has a $1/3$ chance of returning the wrong answer. We want to run the algorithm multiple times to obtain a probability of error of only 2^{-s}.

Instead of running the algorithm with r new random bits each time, we will start with r random bits x_1, then concatenate $h(x_1)$ with a few fresh random bits to create x_2, the "random" bits for the BPP algorithm, and so on.

If we run the algorithm $t = O(s)$ times then the independence error is $2^{-(s+1)}$.

Lemma 1 *The error due to the leftover hash lemma is $t \cdot \epsilon$ ($\epsilon = 2^{-\ell}$).*

We will prove this lemma in the next section, but first let us discuss its implications.

We want both error sources to be of the same order, $t\epsilon \leq 2^{-(s+1)}$. Therefore $\epsilon \leq \Theta\left(\frac{2^{-s}}{s}\right)$. Take $l = s + 2log(s)$. We start out in $\{0,1\}^n$, the error set $\bar{W}$ contains less than $\frac{1}{3}2^n$ elements. We can choose $k = n-2$ because $2^k = \frac{1}{4}2^n < \frac{1}{3}2^n$. This means that at each new stage we need $2+2l$ fresh random bits $= \theta(s)$.

The total number of random bits used by this algorithm is $O(n+s^2)$ (because the hash function requires random bits, too). Note that random walks on expanders perform better.

5.1 Proof

$$
\begin{aligned}
b(x) &= \begin{cases} 1 \text{ if } x \in W \\ 0 \text{ otherwise} \end{cases} \\
D_i &= \text{distribution of } b(x_1) \circ b(x_2) \circ \cdots \circ b(x_i) \circ h \circ x_{i+1} \\
x_{i+1} &= h(x_i) \circ s_i, \quad s_i \text{ uniformly random} \\
p &= \Pr_{x \in_u \{0,1\}^n} [x \in W]
\end{aligned}
$$

Let $b_1, b_2, ...$ be chosen independently such that $\Pr[b_i = 1] = p$ and $\Pr[b_i = 0] = 1 - p$.

$$U_i = \text{ distribution of } b_1 \circ b_2 \circ \cdots \circ b_i \circ h \circ x_{i+1}$$

Claim: $|D_i - U_i| \leq i \cdot \epsilon$ (ϵ is the error in the leftover hash lemma, $2^{-\ell}$).
Proof: (by induction)
i=1

We want to show that $b(x_1) \circ h \circ x_2$ is almost uniform.

$$x_2 = h(x_1) \circ s_1$$

The base follows from the leftover hash lemma because if we condition on $b(x_1)$, then x_1 is chosen uniformly from a large set so $h \circ h(x_1)$ is within ϵ of uniform. Therefore, $b(x_1) \circ h \circ h(x_1)$ is with ϵ of uniform.

Induction

Let U_i' be the distribution of $b_1, \cdots, b_i, b(x_{i+1}), h, x_{i+2}$.

Facts

$$|D_1 - D_3| \leq |D_1 - D_2| + |D_2 - D_3| \tag{1}$$

$$\text{if } f \text{ is deterministic then } |f(D_1) - f(D_2)| \leq |D_1 - D_2| \tag{2}$$

$$\text{By the } i = 1 \text{ case, } |U_i' - U_{i+1}| \leq \epsilon$$

$$\text{By (2), } |D_{i+1} - U_i'| \leq |D_i - U_i| \overset{\text{ind.hyp.}}{\leq} i \cdot \epsilon$$

$$|D_{i+1} - U_{i+1}| \overset{(1)}{\leq} |D_{i+1} - U_i| + |U_i' - U_{i+1}| \leq i\epsilon + \epsilon = (i+1)\epsilon$$

$\square$