CS 880: Pseudorandomness and Derandomization 4/15/2013

Lecture 22: Pseudorandomness from Hardness

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

DRAFT

In the next few lectures, the focus will be on the connection between Pseudorandom Generators and hardness; the class of Boolean circuits of linear size will be studied here. Let start with a definition for this class.

Definition 1 (Class of Circuits). *The class of Boolean circuits defines as following:*

$$\mathcal{C}_r = \{\textit{Boolean circuits of maximum size } r \textit{ with maximum input size } r\}$$

where the size of circuit is the number of gates in the circuit, it actually measures the amount of work done in the circuit. Note that the circuits do not necessarily use all the inputs that they receive.

At the early lectures of this class, it was briefly mentioned that if there exist Pseudorandom Generators for this class, then there exists Pseudorandom Generators for all randomized procedures which run in polynomial time. Because any BPP can be simulated by a Boolean circuit, then there exists Pseudorandom Generators for BBP family. In this class the focus will be in completeness, but the drawn connection are very general and apply to many other classes of algorithms. A few relatively weak closure properties of the class are the only requirements for all the steps.
Also it was mentioned earlier that the existence of a Pseudorandom Generator for a class induces a problem that is hard for that class. Here hardness means that this problem can not be decided in that class; namely, this class can not decide if a certain sequence is in the range of the Pseudorandom Generator. Besides, the range of Pseudorandom Generator is small, and a random sequence has a slight chance to be in this range.
In this lecture, the focus is to show how to make Pseudorandom Generators out of a hard problem for a class, in a generic way. To do so some limitations of that class will be used. This is actually the same method that has been used to construct all the Pseudorandom Generators in the past lectures without talking about any specific weaknesses of the studied classes.
A hard problem for the class of circuits will be defined as below:

$$f : \{0,1\}^m \to \{0,1\} \text{ such that } f \in \mathcal{C}_r \text{ and can not be computed in class } \mathcal{C}_r$$

f is a Boolean function that can not be computed in class $\mathcal{C}_r$ which means there is no small circuit that computes f.

Worst-Case Hardness No matter which circuit has been picked from class $\mathcal{C}_r$, there will be at least one input, on it the circuit computes wrong output for function f. In next lecture, it will be discussed that Worst-Case hardness suffices to construct a Pseudorandom Generator.

Average-Case Hardness No matter which circuit has been picked from class $\mathcal{C}_r$, there will be many inputs, on them the circuit computes wrong output for function f. Average-Case hardness which is weaker than the previous case will be studied in this lecture.

Definition 2 (Average-Case Hardness of a function). *Average-Case hardness of function f is defined as below, where m is the length of input of circuit C.*
$H_f(m) \doteq$ Smallest integer s such that there exist a circuit C of size at most s for which

$$Pr_{x \leftarrow U_m}[C(x) = f(x)] \geq \frac{1}{2} + \frac{1}{s}$$

There is two parameters for the average-case hardness: the size of the circuit and the fraction of the inputs in which the circuit's output differs from function f. The larger the parameter s the easier the above condition will be because first, when the upper-bound for the circuit size grows, the collection of candidate circuits grows as well. Also when s increases, $\frac{1}{s}$ decreases, so the agreement requirement becomes weaker. It means the smallest the s, the hardest the function f will be which is suitable for the development of Pseudorandom Generator. As shown above the two parameters can be combined.

Goal: If there exist a time linear exponential function $f \in E$ such that $H_f(m)$ is high, where $E = Dtime(2^{O(m)})$, then there exist a PRG $G : \{0,1\}^{l(r)} \rightarrow \{0,1\}^r$ for C_r such that:

- its seed length $l(r)$ is small
- it is computable in time $2^{O(l(r))}$

Because of the full derandomization used in the algorithm, the above running time is good enough. How small the seed length can be depends on the hardness. If the hardness is slightly high, then the seed length will be logarithmic. Such case leads to polynomial time derandomization because $2^{log(r)}$ is polynomial.

Distinguishability vs. Predictability

Definition 3 (Distinguisher). *A circuit C_r is an $\epsilon-$distinguisher for a distribution D on $\{0,1\}^r$ if the probability that the circuit outputs 1 on a uniform input differs from the probability that the circuit outputs 1 on a chosen input from the distribution D by at least ϵ as shows below.*

$$|Pr[C(D) = 1] \ - \ Pr[C(U_r) = 1]| \geq \epsilon$$

The distinguisher is a test: whether a distribution is uniform or not.

Note: When D is an $\epsilon-$pseudorandom distribution for a class of linear size circuits, it means that there exists no $\epsilon-$distinguisher in that class C_r.

Definition 4 (Predictor). *P is an $\epsilon-$ predictor for a distribution D if:*

$$(\exists\ i \in [r]) \text{ such that } Pr_{\rho \leftarrow D}[P(\rho[1 \ldots i-1]) = \rho_i] \geq \frac{1}{2} + \epsilon$$

An $\epsilon-$predictor is a special case of $\epsilon-$distinguisher in which if given some prefix of a sample from distribution D, the predictor P can predict the next bit with the advantage $\frac{1}{2}+\epsilon$. In case that the distribution D is the perfect uniform distribution the best possible ϵ is 0, because the prefix does not tell anything about next bit. But if the pseudorandom distribution D has a much smaller support, then predicting the next bit becomes easier. In this course, the predictor P is assumed to be a small circuit, but the definition of a predictor does not require it to be a circuit.

Fact - If there exist an $\epsilon-$predictor of size s, then there is an $\epsilon-$distinguisher of size $s+O(1)$. The construction of the distinguisher based on the predictor follows:

$$C(s) = \begin{cases} 1 & \text{when } P(\rho[1\dots i-1]) = \rho_i \\ 0 & \text{o.w.} \end{cases}$$

where $Pr[P(\rho[1\dots i-1]) = \rho_i] = Pr[C(D)=1]$. It will be shown that up to some lost in the parameter ϵ, predictors as special case of distinguishers are enough for the purpose of this lecture.

Lemma 1. *The existence of an $\epsilon-$distinguisher of size s implies the existence of an $\frac{\epsilon}{r}-$predictor of size s.*

Proof: *Suppose an $\epsilon-$distinguisher C is given. By definition of distinguisher:*

$$|Pr[C(D)=1] \ - \ Pr[C(U_r)=1]| \geq \epsilon$$

The following applies either to C or $\overline{C}$(the negation of C), and the rest of argument will be correct for the circuit which holds this property. Without hurting the argument, let assume that it holds for C.

$$Pr[C(D)=1] \ - \ Pr[C(U_r)=1] \geq \epsilon$$

which means that when the inputs are from D, it is more likely for C to get a 1, than when the outputs are from uniform distribution. This can be used to form a predictor. Another required elements here are hybrid distributions D_i which is a hybrid between D and uniform distribution. The first i bits of D_i are from a sample from D and the rest of the r bits are from a sample of uniform distribution(U_r).

$$D_i = D[1\dots i]U_r[i+1\dots r]$$

Note that $D_1 = U_r$ and $D_r = D$. Now using the telescopic sum, and noticing that all the terms will cancel out except the first term and the last term, the above difference can be written as following:

$$\begin{aligned} Pr[C(D)=1] \ - \ Pr[C(U_r)=1] &= \sum_{i=1}^{r} (Pr[C(D_i)=1] \ - \ Pr[C(D_{i-1})=1]) \\ &\geq \epsilon \end{aligned}$$

If sum of the r terms is at least ϵ, at least one of those terms has the lower bound of $\frac{\epsilon}{r}$.

$$\exists i \in [r] \text{ such that } Pr[C(D_i)=1] \ - \ Pr[C(D_{i-1})=1] \geq \frac{\epsilon}{r}$$

Distributions D_i and D_{i-1} differ only in one position(position i), so the inequality can be rewritten as below:

$$Pr[C(D[1\dots i-1]D[i]U_r[i+1\dots r])] \geq Pr[C(D[1\dots i-1]U_r[i]U_r[i+1\dots r])] + \frac{\epsilon}{r}$$

where $U_r = \rho_1\rho_2 \ldots \rho_r$ by definition. Now, the $\rho_{i+1} \ldots \rho_r$ needs to be fixed the way that the inequality maintain. It actually can be fixed with the choice that results the average probability, namely $\widetilde{\rho_{i+1}} \ldots \widetilde{\rho_r}$. The result will be one more step closer to the predictor: there is a prefix of length i from D and the rest of it is fixed. To define the predictor, $D[1 \ldots i]$ can be used as the input(prefix). In order to guess the $(i+1)$'th bit, let take a random value ρ_i as the initial guess and plug it in the hybrid distribution $D[1 \ldots i-1]\rho_i\widetilde{\rho_{i+1}} \ldots \widetilde{\rho_r}$ into the given distinguisher. By definition, the distinguisher is more likely to output 1 if the input is the correct answer instead of a random value. Nevertheless, the predictor will return the initial guess if the distinguisher outputs 1, otherwise it will return the flipped value of the initial guess. Actually, this is not a predictor yet because it needs a random bit. Let call it P':

$$P'_{\rho_i \in_U \{0,1\}}(D[1 \ldots i-1]) = \begin{cases} \rho_i & when\ C(D[1 \ldots i-1]\rho_i\widetilde{\rho_{i+1}} \ldots \widetilde{\rho_r}) = 1 \\ \overline{\rho_i} & o.w. \end{cases}$$

For now suppose that:

$$Pr_{over\ \rho_i\ and\ D}[P'(D[1 \ldots i-1]) = D[i]] \geq \frac{1}{2} + \frac{\epsilon}{r}$$

The prove of this claim will remain as an exercise. The just mentioned probability is over choices of ρ_i and D. So there exists at least one choice of ρ_i that this advantage maintains; let call it $\widetilde{\rho_i}$ and replace the random bit in P' with $\widetilde{\rho_i}$. This way the construction of predictor is complete.

$$P(D[1 \ldots i-1]) = \begin{cases} \widetilde{\rho_i} & when\ C(D[1 \ldots i-1]\widetilde{\rho_i}\widetilde{\rho_{i+1}} \ldots \widetilde{\rho_r}) = 1 \\ \overline{\widetilde{\rho_i}} & o.w. \end{cases}$$

P is an $\frac{\epsilon}{r}-$predictor. If the original circuit C(from the given distinguisher) is of size s, then P is of size at most s.

This lemma simplifies the task of constructing a pseudorandom generator because with using this connection, constructing a pseudorandom generator reduces to constructing a generator which does not have any $\epsilon-$distinguisher. It is important to notice that in order to construct a PRG, it is enough to construct a generator that fools all distinguishers that are of special type, mainly of the predictor type.
Now, it is the time to construct a PRG out of hardness. Let start with a boolean function on n bits(f) which has high average-case complexity, and based on that construct a PRG for this family of small circuits. As usual, the goal is that this PRG stretches as much as possible because that corresponds to have seed lengths as short as possible.

Construction of Pseudorandom Generators out of Hard Functions Now, it is the time to construct a PRG out of hardness. Let start with a boolean function on n bits(f) which has high average-case complexity, and based on that function construct a PRG for this family of small circuits. As usual, the goal is that this PRG stretches as much as possible because that corresponds to have seed lengths as short as possible. The goal is to construct a PRG for C_r based on $f : \{0,1\}^m \to \{0,1\}$ where this function has a high average-case hardness($H_f(x)$).

Step 1: In this step, let aim to use f to get only one bit stretch. It means that f takes a seed of length m, namely σ, as an input, and returns one pseudorandom bit $f(\sigma)$. Then G defines as below:

$$G : \{0,1\}^m \to \{o,1\}^{m+1} \text{ OR } \sigma \to \sigma f(\sigma)$$

One can claim that this is a PRG because of the connection between unpredictability and pseudorandomness. Basically, because the seed string is chosen from a uniform distribution, there is no predictor for G. So the prefix can not completely be chosen from σ; it needs to have some part out of this range.
Computing $f(\sigma)$ has high average-case complexity, so it is hard to predict it. More precisely, it means no matter how small the size of the circuit is, it is not possible to compute this function correctly on more than $\frac{1}{2} + \epsilon$ fraction of points which means f is unpredictable. But it is not done yet, because extending only one bit is not very impressive. In next step, the process will be modified to stretch reasonably.

Step 2: In this step, the goal is to cycle over the process mentioned in the previous step many times, in order to be able to get more random bits. First thought would be in order to get r bits then cycling r times independently. So the argument from before will still holds. Also the output distribution is pseudorandom. But each cycle requires a separate $m-$bit seed to generate only one pseudorandom bit, so basically it is shrinking instead of extending. Even using joint of the seed bits along with the output bit does not have a good stretch.
Because the output distribution is pseudorandom (not random), instead of picking all the seeds completely independently, they are allowed to be slightly dependent.[1] For now, let just hope that the previous argument still holds. To achieve that let start with a single seed σ of length l where l is longer than m but a lot smaller than r times m. In order to form different inputs for each cycle, pick m positions in σ. Any pair of position sets for different inputs can have a small overlap. Let define these sets as bellow where $i \in [1 \ldots r]$.

$$S_i = \{\text{positions in } \sigma \text{ that are picked for the i'th PRG bit}\}$$

$S_i \subset [l]$ such that exactly m positions are picked for each set, and the pairwise intersections of sets are small:

- $|S_i| = m \forall i \in [r]$
- $|S_i \cap S_j| \leq a \;\; (\forall i, j \in [r]) \; i \neq j$ where a is small.

This is called (m, a)design. If $l = O(m)$ the number of possibilities for different inputs will be exponential. As usual the desire is to have a small seed length l and a big output size r. Let require m to be at least logarithmic in r. So with the seed length of $\log^2 r$, the number of outputs will be r which is an exponential difference. See figure 1 for an illustration of this process.

Lemma 2. *For $m \geq \log r$, there exists an efficiently constructible $(m, \log r)$design with $l = O(m^2)$.*

[1]As seen before, allowing a little bit of dependency can make a huge difference in efficiency. Going from prefect k-wise uniform to almost k-wise uniform made an exponential factor of the difference in the possible seed length.

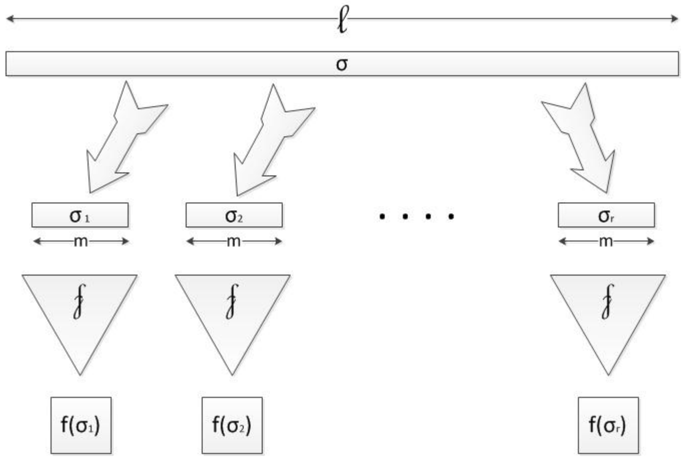

Figure 1: An illustration of a Design where r bits of output is desired, and m is the input size.

This lemma makes it possible to have a real construction by using polynomials of low degree, but the detail is beyond this lecture. However, this lemma will be used to argue that the result is PRG. Let define G_r as bellow:

$$G_r(\sigma) : \{0,1\}^l \rightarrow \{0,1\}^r \text{ and } \sigma \rightarrow f(\sigma|S_1)f(\sigma|S_2)\dots f(\sigma|S_r)$$

where σ is the seed, $l = m^2$, and $\sigma|S_i$ means bits from σ corresponding to S_i. The connection of G_r and the design will be through hardness.

Claim: If this f has high average-case complexity, then the r bits of the output will be pseudo-random.

Theorem 3. *If $H_F(m) \geq r^2$ then G_r is $\frac{1}{r}PR$ for C_r.*