CSC 591 Cryptography 9/19/2018

Lecture Pseudorandom Generators

Lecturer: Alessandra Scafuro *Scribe: Peter Shore*

Lecture Topics

During this lecture, we covered three topics.

The **first topic** that we covered was a continuation of the previous lecture, where we had been given a PRG construction, and were asked to show that it was not a valid PRG. In class we were shown a proof for this.

The **second topic** that we discussed a technique to increase the expansion factor of any PRG. In other words, we learned that as long as we had a PRG that could take as random string, and output a Pseudo Random String of length at least n+1, then we could reuse this PRG to produce a Pseudo Random String of any length. That being said, we did not explicitly prove this at all, we merely discussed the idea behind chaining to extend a pseudo random number.

The **third topic** we covered was how to prove if a function was a PRG or not. We did not cover this as a new topic in class. This was more or less review and practice on how to make assumptions on if a newly constructed PRG is indeed a valid PRG or not. If the newly created PRG appeared to be a PRG, than we learned how to prove that it was a PRG through a proof by reduction. If instead we believed the scheme to not be a PRG, then we must provide an example case for which the scheme breaks.

Disproving a PRG

During the previous lecture, we were asked to prove that a given construction for a PRG G' was invalid. We proved it as follows:

Assume M is a PRG where
M: $\{0,1\}^n \to \{0,1\}^{2n}$,
$s : \{0,1\}^n$,
and s' is the string obtained by negating all bits of s.

$G'(s) = M(s)||M(s')$

D(y) →distinguishes between a uniform distribution and G'

1. First, we observe that G' should be secure for any M (as long as M is a PRG).

2. Second, we construct a PRG M*(s) (that uses a PRG G as building block) as follows:

   M*(S)

   - Parse S as $S_1 ... S_n$
   - If $s_1 = 0$, output G(s) otherwise output G(s').

   We note that M^* is a secure PRG, since G is a PRG.

3. Our next step is to instantiate G' with our PRG M^*. Indeed, recall that G' should work with any PRG. When instantiating G' with M^* we obtain the following behaviour:

   $G'(s) = M^*(s)||M^*(s')$
   If $s_1 = 0$, then $M^*(s)||M^*(s') = G(s)||G(s'') = G(s)||G(s)$
   if $s_1 = 1$, then $M^*(s)||M^*(s') = G(s')||G(s') = G(s')||G(s')$

4. The last step is to show when G' is instantiated with M^*, the output of G' is easily distinguishable from a truly random string. Indeed, there exists a PPT distinguisher D that works as follows:

   D(y)

   - Parse $y = y_L||y_R$
   - Decision: If $y_L = y_R$ then output 1, otherwise output 0.

5. Analysis:

   - if y is the output of G', then Pr[D(y)$\rightarrow$1] = 1
   - if y is the output of a truly uniform distribution, Pr[D(y) $\rightarrow$1] = $\frac{2^{2n}}{2^{4n}} = \frac{1}{2^{2n}}$
   - We conclude that our construction is not a valid PRG

Overview of PRG Expansion Through Chaining (Not a proof)

The intuition behind our second topic is to find out if a PRG can produce a Pseudo Random String of any length. We have shown that we can produce a PRG that takes S as an input string where S is a truly random string following the Uniform distribution and output a string that is of greater length than the input string. From this, we want to prove that if we can produce a PRG that can take a truly random string of length n and produce a Pseudo Random String of length n+1, then we are able to produce a PRG that can produce a Pseudo Random String of any length. We can prove this using Hybrid Arguments.

We did not give a formal proof in class, but we did cover the over all idea as to why any PRG is able to produce an output string of length n + p(n) where p(n) is any polynomial function. The general idea is, that as long as our PRG is valid and can an output a Pseudo Random String of length at least n+1, then we are able to reuse that new Pseudo Random String and by passing it through our PRG G, receive a new Pseudo Random String of length n+2. We can then continue this process until we have a Pseudo Random String

of length n+p(n).

Theorem → *If there exists a pseudorandom generator G with expansion factor n+1, then for any polynomial* poly*, there exists a pseudorandom generator with expansion factor* poly(n)*.* $|s_1| = n$ *and* $|G(s_1)| = n + 1$

G(s)
$t_1 \rightarrow G(s)$(where $|t_1| = n + 1$)
$t_1 = t_0 + b_1$(where b_1 is 1 pseudo random bit)
$t_2 \rightarrow G(t_1)$(where $|t_2|$ = n+2)
.
.
.
$t_n = t_{n-1} + b_n$
$t_{n+1} \rightarrow G(t_n)$

} $G(G(G(s)))$...poly(n)times

Proving and disproving the validity of a PRG

The intuition for our third topic is to explain how to prove or disprove whether or not a newly constructed PRG is indeed a valid PRG.

We have two options:

1. If the newly created PRG appeared to be a PRG, then we must prove that it was a PRG through a proof by reduction.
2. If instead we believed the scheme to not be a PRG, then we must provide an example case such that a distinguisher can decide with non negligible probability between our PRG and a truly random number.

Assumption

Assume G is a PRG $G : \{0,1\}^n \rightarrow \{0,1\}^{2n}$

Two Examples

$H_1(S)$

- x ←G(s)
- y ←G(0^n)

Output $y \oplus$x

$H_2\,(S)$

- x $\leftarrow$G(s)
- y $\leftarrow$G(0^n)

Output $x||$y

Proving the validity or invalidity of H_1 and H_2

So, now our job is to decide whether or not these schemes are vaid PRGs or not. As mentioned above, we have two options when we approach the question as to whether or not a newly constructed PRG is valid.

1. If the newly created PRG appears to be a PRG, then we must prove that it was a PRG through a proof by reduction.
2. If instead we believed the scheme to not be a PRG, then we must provide an example case for which the scheme breaks.

Let us first consider H_2 since it seems to be a bit easier to prove. We notice that since we are always concatenating G(0^n) onto the end of G(n), we have a pattern we can utilize in proving H_2 is an invalid PRG.

Proof $\rightarrow$Assume there exists a PPT distinguisher D that can distinguish between H_2 and a truly random number with non-negligible probability:
D(y) where y=4n bits

1. Parse y as $y_L||y_R$
2. if $y_R = G(0) \rightarrow$ *output* 1
3. else $\rightarrow$ *output* 0

Analysis $\rightarrow$

1. case y is the output of H_2 $\rightarrow$Pr[D(y) $\rightarrow$1] = 1
2. case y is truly random $\rightarrow$Pr[D(y) $\rightarrow$1] = $1/2^n$
3. Since $1 - \frac{1}{2^n}$ = non-negligable we can conclude that H_2 is an invalid PRG.

Now lets consider H_1. Since we cannot come up with an example that disproves H_1 being a PRG, we assume that H_1 is indeed a PRG and attempt to prove it through reduction.

Theorum: if G is a PRG, then H_1 is also a PRG

Proof:

1. Towards a contradiction, assume there exists a distinguisher D_1 that distinguishes the output of H_1 from random with non-negligible probability

2. we construct a polynomial time distinguisher that distinguishes the output

Reduction →D(x)

1. compute y ←G(0)

2. send z=y ⊕ x to D_1

3. when D_1 outputs a bit b_1, also output b

Analysis →

- if x is the output of G, then z is distributed exactly as the output of H_1 →Pr[$D_1(x)$ →1] = Pr[D_1(z) →1] where x ←G(s) and z ←H_1

- if x is truly random, z is distributed exactly as random →Pr[D_1(x) →1] = Pr[D_1(z) →1] where x ←$\{0,1\}^{2n}$ and z ←$\{0,1\}^{2n}$

Conclusion: Since a distinguisher D_1 can distinguish H_1 with the same probabilty that it can distinguish G_1, and G_1 is a true PRG that is indistinguishable from, or distinguishable with negligible probability from a truly random number, H_1 is a valid PRG.