Lecture 1: Introduction and reduction from weak OWF to strong OWF

instructor: Jin-Yi Cai Scribe: Bess Berg, Michael Kowalczyk

1 Overview

Handouts for this lecture include Luby's notes, which contain some background definitions and other basic notions, and a set of notes by Martin Tompa discussing random walks and Nisan's Theorem. We give an outline of material planned for this course and then start the topic of today's lecture, a reduction from weak to strong one-way functions.

2 Course Outline

Below is the planned material for approximately the first third of the course.

1. One-way functions, permutations, reductions
2. Expanders, random walks, Nisan's generator
3. Space Bounded Computation
   - Focus on issue between NL, SL, L, and random walks
   - $RL \subseteq SPACE[\log^{1.5} n]$ (Saks-Zhou)
   - Space-time tradeoff (J.-Y. Cai, V. Chakaravarthy, and D. van Melkebeek)
4. $USTCON \in L$ so $SL = L$ (Reingold)
5. Lower bounds
6. Hardcore bit
7. Graph isomorphisms

3 $w(n)$-weak $S(n)$-secure one-way functions

We need to define the notion of a $w(n)$-weak $S(n)$-secure one-way function, $f : \{0,1\}^n \to \{0,1\}^{\ell(n)}$. The intuitive idea is that we want this function to be easily computable given an input, but generally difficult to invert. Without loss of generality, we will use strings of the form $x \in \{0,1\}^n$ for inputs[1]. In this particular case we call n the security parameter; we think of n as indicating the level of secrecy. We consider the image $f(x) \in \{0,1\}^{l(n)}$ and want to capture the idea that f is in general hard to invert.

[1] We will sometimes use the notation 2^n in place of $\{0,1\}^n$.

Consider an adversary A that wants to invert the image of f to recover the original string. We allow A to be an algorithm or circuit family (sometimes we will use the terms uniform and nonuniform, respectively). For every such adversary A running within time $S(n)$, the probability of inverting the function on a random input should not be too high. Note that if f is not one-to-one, then it might be unreasonable to require A to try to guess the correct preimage, and with this in mind, we define the success probability of A to be:

$$\Pr_{x \in_R 2^n}[f(A(f(x))) = f(x)].$$

Note that if f is a permutation (and therefore one-to-one), we can express this in a slightly cleaner form:

$$\Pr_{x \in_R 2^n}[A(f(x)) = x].$$

Definition 3.1 ($w(n)$-weak $S(n)$-secure one-way function) *A function f is a $w(n)$-weak $S(n)$-secure one-way function if for all adversaries A with runtime at most $S(n)$,*

$$\Pr_{x \in_R 2^n}[f(A(f(x))) = f(x)] \leq 1 - w(n).$$

Typically, the weakness $w(n)$ is a small quantity, say $w(n) = \frac{1}{n^2}$. The term "weak" is used because A is only guaranteed to fail for that small portion of the time. It stands to reason that allowing $w(n)$ to be small gives us a better chance of finding such functions, and therefore we make the allowance.

Example: Consider the problem of factoring composite numbers into two factors. This problem is difficult, but mainly due to a "core" set of integers with only two prime factors of about the same size. Indeed, we can efficiently factor an overwhelming percentage of the integers by simply trying out the first few primes. Considering the set $P = \{\text{primes } p : \frac{\sqrt{N}}{2} \leq p \leq \sqrt{N}\}$, the Prime Number Theorem tells us that $|P| = \pi\left(\sqrt{N}\right) - \pi\left(\frac{\sqrt{N}}{2}\right) \approx \frac{2\sqrt{N}}{\ln N} - \frac{\sqrt{N}}{\ln N} = \frac{\sqrt{N}}{\ln N}$. The number of integers with exactly 2 prime factors produced from this set is $\binom{|P|}{2} \approx \frac{N}{\ln^2 N}$. Thus there are a polynomial fraction (in the description length) of numbers which seem to be difficult. Assuming that they are indeed hard to factor, then this is a $\frac{1}{n^2}$-weak one-way function. ⊠

We use a fairly standard defintion for strong one-way functions, which is similar to its weak counterpart.

Definition 3.2 (Strong one-way function) *A function $f : 2^n \to 2^{l(n)}$ is a strong one-way function if for all polynomial (uniform or nonuniform) adversaries A,*

$$\delta_A(n) = \frac{1}{n^{\omega(1)}} \textit{ where } \delta_A(n) = \Pr_{x \in_R 2^n}[f(A(f(x))) = f(x)]$$

Luby goes further and defines a "time success ratio", the ratio of the adversary's runtime over its success rate. A strong one-way function is thus said to be $S(n)$-secure if $\frac{\text{runtime of } A}{\delta(n)} \geq S(n)$. This notion is not standard though it is occasionally useful, in particular in the case where $S(n)$ is super-polynomial.

4 Private versus public input

The idea here is that we may be able to come up with a function that is harder to invert due to the use of additional random bits that are available to all parties. For example, consider a $1 \times n$ matrix B consisting of n-bit integers and generated from public random bits. Then the function f given by $f(B, x) = B \cdot x$ where $x \in \{0,1\}^{n \times 1}$ is easy to compute. However, not only is this function hard to invert, but it is hard to invert even when we are given the randomly generated matrix B. Since the adversary can look at the output of the function as well as the matrix B, we write[2] the function $f : 2^{p(n)} \times 2^n \to 2^{l(n)}$. We can then write the success probability as follows[3].

$$\delta(n) = \Pr_{\substack{x \in_u 2^n \\ y \in_u 2^{p(n)}}} [f_y(A[f_y(x), y]) = f_y(x)]$$

The notion of $S(n)$-security is defined similarly here as before.

5 Reduction from weak to strong one-way functions

We aim to get a reduction from weak one-way functions to strong one-way functions. By this, we mean that given any weak-one way function f, we can construct a strong one-way function F. We will prove that F is strong by assuming that there exists an algorithm, A, that breaks F and show that there is an algorithm, S, that can break f by using A as an oracle. In other words, we will be doing a proof by contradiction.

$$\begin{array}{ccc} f & \rightsquigarrow & F \\ S^A & \overset{S}{\leftarrow} & A \end{array}$$

THEOREM 5.1 *Given a $w(n)$-weak one-way function f, we can construct a strong one-way function g.*

The idea is simply to encode public random bits multiple times with f. Thus we can define $g : \{0,1\}^{n \times N} \to 2^{l(n) \times N}$ with $(x_1, \ldots, x_N) \mapsto (f(x_1), \ldots, f(x_N))$. Since the success probability for f is at most $\delta(n) = 1 - w(n)$, we would hope that setting $N \simeq \frac{n}{w(n)}$ would yield a success probability for g of at most $(1 - w(n))^N = e^{-n}$ as $w(n)$ limits to zero. However, we can't say this because we have no guarantee this will work for many simultaneous applications of f. However, we can make two observations:

1. N can't be much smaller than $\frac{n}{w(n)}$ in this reduction. For example, $N \simeq \frac{1}{w(n)}$ may allow the adversary to succeed a constant percentage of the time.

2. Since we must have $N = n^{O(1)}$, it follows that $w(n) \geq \frac{1}{n^c}$. We will henceforth assume this, and such a parameter is called non-negligable.

We now give some important definitions and a technical lemma that we will use to prove the theorem.

[2] $p(n)$ is standard for public coin flips

[3] $\in_u$ indicates independently uniformly produced random bits

Definition 5.2 ((ϵ, δ)-forward expanding) *A bipartite graph (L, R, E) of left degree D_l and right degree D_r is (ϵ, δ)-forward expanding if $\forall S \subseteq L$ such that $\mu(S) \geq \epsilon$, we have $\mu(\Gamma(S)) \geq 1 - \delta$. Here μ is the density of a subset on the respective side of the bipartite graph and $\Gamma(S)$ is the neighbor set of S.*

Definition 5.3 ((ϵ, δ, γ)-reverse expanding) *A bipartite graph (L, R, E) of left degree D_l and right degree D_r is $(\epsilon, \delta, \gamma)$-reverse expanding if for all $T \subseteq R$ such that $\mu(T) \geq \delta + \gamma$, we have $\mu(S) > 1 - \epsilon$, where $S = \{x \in L \mid \Pr_{y(x)}[y(x) \in T] \geq \gamma / D_r\}$. (Here $y(x)$ is a random neighbor of x.)*

These two definitions are connected by the following lemma, which will be proved next lecture and around which the reduction will be constructed.

Lemma 5.4 *For all $\epsilon, \delta > 0$, if the bipartite graph (L, R, E) is (ϵ, δ)-forward expanding, then for all $\gamma > 0$ it is also $(\epsilon, \delta, \gamma)$-reverse expanding.*

Given a weak one-way function f, let $L = \{0,1\}^n$ be the set of all length n inputs to f, and let $R = \{0,1\}^{n \cdot N}$ be the set of all inputs to the function g (defined above). For any $x \in L$ there will be an edge between x and every N-tuple in R that contains x as one of its entries. Then the degree of vertices in L is $D_l = N \cdot (2^n)^{N-1}$, and the degree of vertices in R is $D_r = N$. Recall that our ultimate goal is to show that g is a strong one-way function. To do this, we assume otherwise. That is, we assume there exists some adversary A that can invert g with some reasonable probability of success. Then, given some $z = f(x)$, we will show that we can use A as a subroutine to invert f with a probability of success that contradicts that f is a weak one-way function. This contradiction implies that there can be no such adversary A, and therefore g is indeed a strong one-way function. To use A as a subroutine in order to invert f, assume we are given some $z = f(x)$. We randomly pick a spot $i \in_u [1, N]$ and pick x_j randomly for $1 \leq j \leq N, j \neq i$. We then sneak the z in, submitting $\langle f(x_1), \ldots, f(x_{i-1}), z, f(x_{i+1}), \ldots, f(x_N) \rangle$ to subroutine A to obtain some x' corresponding to z. Check if $f(x') = z$, and if so, return x'. In the next lecture we will use the graph defined above along with the lemma to obtain the desired contradiction.