YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

CPSC 461b: Foundations of Cryptography *Notes 5 (rev. 1)*
Professor M. J. Fischer *January 27, 2009*

Lecture Notes 5

15 Other Classes of One-Way Functions

Our definitions of strongly and weakly one-way functions are liberal in the functions that may be considered one-way, but they are restrictive in requiring that one-way functions be hard to invert on almost all lengths n. We explore variations in the definitions, both to achieve properties that are desirable in practice and to gain practice in working with the definitions.

15.1 One-way functions on certain lengths

Natural candidate one-way functions might be defined only on strings of certain lengths. For example, a graph-theoretic function that might be conjectured to be one-way might take as input the adjacency matrix of an N-node graph, which is naturally encoded as a binary string of length N^2. Such a function would only be defined for strings whose length is a perfect square, and we would not care what it did or whether it was hard to invert for inputs that were not perfect squares. We are therefore led to consider functions that are defined only for certain lengths.

Let $I \subseteq \mathbb{N} - \{0\}$ be an infinite set of lengths. Define $s_I(n)$ to be the smallest integer $n' > n$ such that $n' \in I$. We say that I is *polynomial-time enumerable* if there is a polynomial time algorithm that on input n writes $s_I(n)$ 1's on its output tape and halts. It follows that $s_I(n)$ is polynomially bounded since a polynomial-time computation cannot produce an output string that is longer than its running time.[1] Note also that by excluding 0 from I, we have $I = \{s_I(n) \mid n \in \mathbb{N}\}$.

For example, suppose $I = \{n^2 \mid n \in \mathbb{N} - \{0\}\}$. Then $s_I(3) = 4$ and $s_I(4) = s_I(5) = s_I(6) = s_I(7) = s_I(8) = 9$. We can compute $s_I(n)$ as follows: Test $n + 1$, $n + 2$, $\ldots$, until a number m is reached that is a perfect square, and output m in unary. This algorithm is easily seen to be computable in polynomial time since testing if a number m is a perfect square can be done in polynomial time, and at most n numbers need to be tested before encountering a perfect square. (More precisely, the worst case comes when $n = k^2$ itself is a perfect square, in which case at most $(k + 1)^2 - k^2 = 2k + 1 = O(\sqrt{n})$ numbers must be tested.)

Definition: Let $I \subseteq \mathbb{N}$ be a polynomial-time enumerable set. The function f is *strongly one-way on lengths in I* if it satisfies the following conditions:

1. f is computable in polynomial time.

2. For all probabilistic polynomial-time algorithms A', all positive polynomials $p(\cdot)$, and all sufficiently large $n \in I$,

$$\Pr[A'(f(U_n), 1^n) \in f^{-1}(f(U_n))] < \frac{1}{p(n)}. \tag{1}$$

[1] Strictly speaking, this depends on ones input and output conventions. For example, a Turing machine that used the same tape for both input and output might be able to "compute" the identity function by simply halting immediately, but in any case, the length of the output string would be bounded by $n + p(n)$, where n is the length of the input string and $p(n)$ is a polynomial bound on the running time.

Definition: Let $I \subseteq \mathbb{N}$ be a polynomial-time enumerable set. The function f is *weakly one-way on lengths in I* if it satisfies the following conditions:

1. f is computable in polynomial time.
2. There exists a positive polynomial $p(\cdot)$ such that for all probabilistic polynomial-time algorithms A' and all sufficiently large $n \in I$,

$$\Pr[A'(f(U_n), 1^n) \notin f^{-1}(f(U_n))] > \frac{1}{p(n)}. \tag{2}$$

Note that these definitions become identical to the respective original definitions for strongly and weakly one-way functions if one takes $I = \mathbb{N}$, so they are strict generalizations of the former concepts.

One can easily transform a strongly (weakly) one-way function f on lengths in I into an ordinary strongly (weakly) one-way function g.

Theorem 1 *Let $I \subseteq \mathbb{N}$ be polynomial-time enumerable, and let f be strongly (weakly) one-way on lengths in I. Define $g(x) = f(x')$, where x' is the longest prefix of x such that $|x'| \in I$, and $g(x) = 0$ if x itself is shorter than any string in I. Then g is strongly (weakly) one-way.*

Proof: We first describe a polynomial-time algorithm to compute g. Note that $m \in I$ iff $s_I(m-1) = m$, so we can test membership in I in polynomial time. On input x, we test the lengths of successively shorter prefixes x' of x, beginning with x itself, until one is obtained that is in I or we determine that none are in I. We then output $f(x')$ or 0 depending whether such a prefix is found. This takes time $O(np(n) + q(|x'|))$, where $p(\cdot)$ bounds the time to compute s_I and $q(\cdot)$ the time to compute f. It follows that g can be computed in polynomial time.

We now show that g is hard to invert. Suppose to the contrary that there is a p.p.t. algorithm A' and a polynomial $p(\cdot)$ such that for infinitely many n,

$$\Pr[A'(g(U_n), 1^n) \in g^{-1}(g(U_n))] \geq \frac{1}{p(n)}. \tag{3}$$

That is, A' successfully inverts g a noticeable amount of the time on an infinite set of lengths I'.

We construct an algorithm A and show that it successfully inverts f a noticeable amount of the time on an infinite subset of I. Here's how A works. $A(y, 1^m)$ runs $A'(y, 1^n)$ for each n such that $m \leq n < s_I(m)$. Suppose $A'(y, 1^n)$ produces output x'. A tests if $y = f(x')$, and if so, it outputs x' and halts. If it fails to invert f for all values of n, then it outputs failure and halts.

Let $J = \{m \in I \mid \{m, m+1, \ldots, s_I(m)-1\} \cap I' \neq \emptyset\}$, that is, J consists of those $m \in I$ such that the interval $[m, s_I(m)-1]$ includes a number $n \in I'$, the infinite set of lengths for which equation 3 holds. We argue that

$$\Pr[A(f(U_m), 1^m) \in f^{-1}(f(U_n))] \ \geq \ \Pr[A'(g(U_n), 1^n) \in g^{-1}(g(U_n))] \tag{4}$$

holds for all $m \in J$ and $n \in [m, s_I(m)]$.

First, we claim that $f(U_m)$ and $g(U_n)$ are identically distributed random variables. This is because if x has length n, then the longest prefix x' of x with $|x'| \in I$ is the prefix of length m, and the length-m prefixes of uniformly distributed strings of length n are themselves uniformly distributed. Moreover, by definition of g, $g(x) = f(x')$, so the claim follows.

Let $y = g(x) = f(x')$. $A(y, 1^m)$ succeeds in inverting f on y if $A'(y, 1^n)$ succeeds in inverting g on y for any $n \in [m, s_I(m)]$ since it tries them all. Hence, it's probability of success is at least as great as that of $A(y, 1^n)$.

Finally, combining inequalities 3 and 4 gives that

$$\Pr[A(f(U_m), 1^m) \in f^{-1}(f(U_n))] \;\geq\; \frac{1}{p(n)} \tag{5}$$

holds for all $m \in J \subseteq I$, contradicting the hypothesis that f is strongly one-way on lengths in I. ■

The proof for weakly one-way functions is similar and is omitted.

16 Regular and length-preserving functions

A function f on strings is *length regular* if $|x| = |y| \Rightarrow |f(x)| = |f(y)|$. Thus, the length of the argument determines the length of the result. f is *length preserving* if $|f(x)| = |x|$.

We first remark that if the function f of Theorem 1 above is length preserving, a slight modification of the construction of g can make it length preserving as well. Namely, if x' is the longest prefix of x with $|x'| \in I$, let x'' be the remainder of x, so $x = x'x''$, and define $g(x) = f(x')x''$. The proof would have to be adjusted accordingly and is left as an exercise.

We now show that if f is an arbitrary strongly (weakly) one-way function, we can construct a length-preserving strongly (weakly) one-way function f''. We do it in two stages:

1. Let $p(\cdot)$ be a polynomial such that $|f(x) \leq p(|x|)$. Such a $p(\cdot)$ exists since f is polynomial-time computable. Then
$$f'(x) \overset{\text{df}}{=} f(x)10^{p(|x|)-|f(x)|}. \tag{6}$$
Clearly f' is length regular since $|f'(x)| = p(|x|) + 1$ for all x.

2. Let $I = \{p(n) + 1 \mid n \in \mathbb{N}\}$. We define f'' on strings of lengths in I. Let $|x| = p(n) + 1 \in I$. Write $x = x'x''$, where $|x'| = n$ and $|x''| = p(n) + 1 - n$. Then
$$f''(x) \overset{\text{df}}{=} f'(x') = f(x')10^{p(n)-|f(x)|}. \tag{7}$$
$|f''(x)| = |f'(x')| = p(|x'|) + 1 = |x|$, so f'' is length regular.

The proofs that f' and f'' are strongly (weakly) one-way given that f is are similar to the proof of Theorem 1 and are left as exercises.

17 Non-uniform one-way functions

A function f is *non-uniformly strongly (weakly) one-way* if it satisfies the definition for being strongly (weakly) one-way, where the inverting algorithm is allowed to be a non-uniform family $\{M_0, M_1, M_2, \ldots\}$ of polynomial-time Turing machines with polynomial-size descriptions, or equivalently, a family of polynomial-size circuits $\{C_n\}_{n\in\mathbb{N}}$. One can show that anything computed by a Turing machine in polynomial time can also be computed by a non-uniform family of polynomial-time polynomial-size Turing machine or by a polynomial-size family of circuits. As a result, we have:

Theorem 2 *If f is non-uniformly strongly (weakly) one-way, then f is strongly (weakly) one-way.*

The converse is not known, and there remains the possibility that strongly (weakly) one-way functions exist but non-uniformly strongly (weakly) one-functions do not.

It is also conceivable that weakly one-way functions exist but strongly one-way functions do not. However, we prove that is not the case, which will be the next topic.