

Computational Difficulty

One-Way Functions

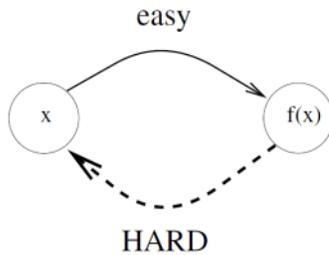


Figure 2.1: One-way functions: an illustration.

"Easy to compute but hard to invert on the average."

The existence of secure encryption schemes implies the existence of an efficient way (PPT algorithm) to generate instances with corresponding auxiliary input such that :

1. It is easy to solve these instances given the auxiliary input.
2. It is hard, on the average, to solve these instances when not given the auxiliary input.

$$\begin{array}{c}
 \begin{array}{ccc}
 r & \xrightarrow{f} & \begin{array}{c} \text{(message)} \\ x \end{array} & \xrightarrow{h} & z \\
 & & \begin{array}{c} y \\ \text{(key)} \end{array} & & \\
 f(r) = xy & & h(xy) = z & & \\
 f_1(r) = x & & & &
 \end{array}
 \end{array}$$

If reversing $f_1(r)$ was easy then encryption scheme would break.

"Processes that are easy in forward direction and are hard to reverse are abundance in nature e.g. the lightning of a match. We need a complexity-theoretic analogue of the daily experience !" - Oded Goldreich.

Definition of Strong one-way functions. Discuss auxiliary input 1^n , function $f_{\text{len}(\cdot)}$

Random guess algorithm A_1 :

1. For any one-way f , inverting probability can be made easily strictly positive.
2. If f is one-way then collision probability of $f(U_n)$ is negligible.

Fixed output algorithm A_2 :

3. If f is one-way the fraction of x in $\{0,1\}^n$, that are mapped to $f(0^n)$ are negligible. Same in fact for any string s in $\{0,1\}^n$.

Negligible probability stays negligible even if algorithm repeated polynomial number of times.

Noticeable functions. Functions may be neither negligible nor noticeable.

Definition (Weak one-way functions):

Functions defined only on some lengths : Polynomial-time enumerable set
I. Proposition 2.2.3. and proof.

"Reducibility argument" : Reduction that needs to preserve the success probabilities of algorithms.

Length regular functions, length preserving functions.

Show how to obtain length regular f' and length preserving f'' one-way functions from any one-way function f .

Point out that if f is 1-1 then so is f' but not f'' . Hence we can assume w.l.o.g. that a one-way function is length regular but not length preserving. In fact the assumption that 1-1 one-way functions exist seem stronger than existence of arbitrary one-way function.

Candidates for one-way functions :

1. Integer Factorization: $f_{\text{mult}}(x,y) = xy$
 - Assuming intractability of factoring e.g. that given the product of two uniformly chosen n -bit primes, it is infeasible to find the prime factors and using density of primes (which guarantees that at least $N/\log N$ of the integers less than N are primes), f_{mult} is at least weakly one-way.
 - Best known algorithms for factoring have sub-exponential running times.
2. Decoding of Random Linear codes: (n,k,d) -linear codes. Gilbert-Varshamov bound, let δ, κ, ϵ be constants such that $\kappa < 1 - H_2((1+\epsilon)\delta)$.

$$f_{\text{code}}(C,x,i) = (Cx + e(i))$$

where C is a kn -by- n binary matrix, x is a kn -dimensional binary vector, i is the index of an n -dimensional binary vector having at most $\delta n/2$ one-entries within a corresponding enumeration of such vectors (the vector itself is denoted $e(i)$) and the arithmetic is in the n -dimensional binary vector space.

- An efficient algorithm for inverting f_{code} will yield an efficient algorithm for decoding a non-negligible fraction of the constant-rate linear codes, which would constitute an earth-shaking result in coding theory.

3. The Subset-Sum problem:

$$f_{\text{subset-sum}}(x_1, x_2, \dots, x_n, I) \stackrel{\text{def}}{=} (x_1, \dots, x_n, \sum_{i \in I} x_i)$$

where $|x_1| = |x_2| = \dots = |x_n| = n$, $I \subseteq \{1, 2, \dots, n\}$

- The subset-sum problem is known to be NP-complete.
- The conjecture that $f_{\text{subset-sum}}$ is one-way is based on the failure of known

algorithms to handle random "high density" instances i.e. instances in which the length of the elements approximately equals the number as in the definition of f_{ssum} .

- Efficient algorithms are known for "low density" instances.

Non-Uniformly one-way functions : Definition 2.2.6, Proposition 2.2.7

- Non-uniform notions of security imply uniform notions of security.
- The converse is not necessarily true. It is possible that one-way functions exist yet there are no non-uniformly one-way functions.
- However this situation seems unlikely and it is widely believed that non-uniformly one-way functions exist. In fact, all candidates mentioned previously are believed to be non-uniformly one-way functions.

Weak one-way functions imply Strong one-way function.

First show that not every weak one-way function is strong one-way. Proposition 2.3.1 and proof.

Thm 2.3.2 and proof.

Definition: Collection of functions and collection of one-way functions.

- $I\{1^n\}$ need not be uniformly distributed over $I^{\text{bar}} \cap \{0,1\}^n$
- $D(i)$ need not be concentrated on a single string
- $D(i)$ need not be uniformly distributed over D_i .

A triplet of PPT algo. (I,D,F) constitutes a collection of one-way functions if there exists a collection of functions for which these algo. Satisfy the forgoing two conditions.

Relaxations:

1. $I\{1^n\}$ can be $p(n)$ bits long.
2. All algorithms can fail with negligible probability.

Additional properties:

1. Having an efficiently recognizable (deciding membership) set of Indices.
2. Efficiently recognizable collection of domains.

Def. 2.4.4 (Trapdoor permutations)

Hardcore Predicates:

Def 2.5.1

Example: Hard core due to information loss $b(cx) = c$, $f(cx) = 0x$

Exercise 25: If f is one-to-one, hard core for f exists iff f is one-way.

Define hard core for one-way collection (I,D,F) .

Thm 2.5.2

- This requires f to be strongly one-way and the conclusion is false if f is only weakly one-way.
- g maintains the properties of f , that is length-preserving, one-to-one
- Analogous statement holds for collection of one-way functions with/without trapdoor.

Emphasize about the proof that why :

1. We need polynomially many r 's.
2. Why Chebyshev is enough and don't need Chernoff and hence pairwise independence is enough.

Assignments:

1.	Chapter 1 : 1,2,3,4,5
2.	Chapter 2: 1,2,6,7
3.	Chapter 2: 8,9,10,12,18
4.	Chapter 2: 21, 25,27,28(1,2),30,31