Chapter 16

Amplification of Hardness

Many computational problems exhibit a weak form of hardness: though easy to solve on most inputs they seem hard on to solve on a nonnegligible fraction of inputs. As example, consider the following version of the factoring problem: given an n-bit integer N, find a factor of N. This problem is easy to solve for most integers, since every second integer is a multiple of 2, every third integer is a multiple of 3, and so forth. However, about $1/n^2$ fraction of n-bit integers have the form $p \cdot q$, where p, q are primes each with at least $n/3$ bits. For such numbers, known factorization algorithms seem to require superpolynomial time.

Another example is the satisfiability problem on random n-variable 3CNF formulae that are generated as follows. Pick a random number m from 1 to $27\binom{n}{3}$ (this last number is the maximum number of clauses in a 3CNF formula with no redundant clauses), and generate a random 3CNF formula with m clauses. When $m < 4n$, the formulae thus generated are overwhelmingly likely to be satisfiable and when $m > 5n$, the formulae are overwhelmingly likely to be unsatisfiable, and furthermore, known algorithms for satisfiability run in polynomial time on most such formulae. Around $m = 4.2n$, however, the running time of known algorithms turns superpolynomial on almost all instances.

Such "weakly hard" computational problems would not suffice for most cryptographic purposes, which need "strongly hard" problems that are difficult to solve on almost all inputs. A beautiful result by Yao shows that weakly hard problems can be used to define strongly hard problems. Several proofs are now known; we present Impagliazzo's proof, which we find the most intuitive. Also, we only present the proof only for boolean functions (i.e., languages), and leave the extension to many-valued functions as an exercise.

16.1 Hardness and how it is amplified with Yao's Lemma

Definition 16.1 (δ-hardness) *Let f be a boolean function on n-bit inputs, $\mathcal{D}$ a distribution on n bit strings. If $\delta < 1/2$ and $g \in [n, \frac{2^n}{n}]$ then we say that f* is δ-hard on distribution $\mathcal{D}$ for circuits of size g *if for any boolean circuit C with at most g gates:*

$$\Pr_{x \in \mathcal{D}}[f(x) = C(x)] < 1 - \delta. \tag{16.1}$$

A "weakly hard" function is one that satisfies the definition for some $\delta = 1/\text{poly}(n)$ and g that is superpolynomial in n. Yao showed that such a weakly hard function can be used to define a "strongly hard" function, satisfying the definition for any $\delta' = \frac{1}{2} - 1/\text{poly}(n)$, and g still superpolynomial. Note that strongly hard functions look like random functions to small circuits, which cannot have much more than the trivial 50/50 odds of computing f on a random input.

DEFINITION 16.2 *If $S \subseteq \{0,1\}^n$ and $\epsilon > 0$, boolean function f is a* ϵ-hard core on S for circuits of size g *if for any boolean circuit C with at most g gates:*

$$\Pr_{x \in S}[f(x) = C(x)] < \frac{1+\epsilon}{2}. \tag{16.2}$$

This notion is inbetween that of a weakly hard and strongly hard function. This is seen by the following observations (all assume that the functions are hard against circuits of size g).

- If function f is weakly hard, say $(1-\delta)$-hard, then every circuit of size g computes f incorrectly on δ fraction of inputs.
- If function f is ϵ-hard core on a set S of size $\delta 2^n$, then every circuit of size g computes f incorrectly on $(1-\epsilon)/2$ fraction of inputs in S. In particular, f is $(1-\delta/2+\epsilon\delta/2)$-hard on the uniform distribution for circuits of size g.
- If function f is $(1/2+\epsilon)$-hard core on a set S of size $\delta 2^n$, then every circuit of size g computes f incorrectly on $(1-\epsilon)/2$ fraction of inputs in S. In particular, f is $(1-\delta/2+\epsilon\delta/2)$-hard on the uniform distribution for circuits of size g.

The function of Definition 16.2 behaves like a strongly hard function on the subset S of instances, since small circuits do not have much better than 50/50 odds of computing f on a random input of S. However, the function may well be easy on points outside S. However, it is easy to check (see Problems) that the function of Definition 16.2 is $(1-\delta/2+\epsilon\delta/2)$-hard on the uniform distribution for circuits of size g. Thus every hard-core function is weakly hard. Impagliazzo shows a converse: every weakly hard function is hard-core on some set of nonnegligible size. (Below, think of g as $\gg 1/(\epsilon\delta)^2$.)

LEMMA 16.1 (IMPAGLIAZZO)
Suppose boolean function f is δ-hard for size g on n-bit strings. Then for every $\epsilon > 0$ there is a set S of size $\geq \delta 2^{n-1}$ such that f is ϵ-hard core on S for circuits of size $g\epsilon^2\delta^2/8$.

From this Lemma, it is easy to prove Yao's result.

THEOREM 16.2 (YAO'S XOR LEMMA)
Let f be a boolean function on n-bit inputs that is δ-hard for circuits of size g on the uniform distribution. Then the function $f^{\oplus k}$ on nk-bit inputs defined as

$$f^{\oplus k}(x_1, x_2, \ldots, x_k) = f(x_1) \oplus f(x_2) \oplus \cdots \oplus f(x_k)$$

is $(\frac{1}{2} - (\epsilon + 2(1-\delta)^k))$-hard on the uniform distribution for circuits of size $g' = g\epsilon^2\delta^2/8$.

PROOF: The proof is by contradiction. Assume some circuit C of size at most g' satisfies $\Pr[C(x_1, x_2, \ldots, x_k) = f^{\oplus k}(x_1, x_2, \ldots, x_k)] \geq (1 + \epsilon + 2(1-\delta)^k)/2$.

By Impagliazzo's Lemma, f is ϵ-hard-core size for size g' on some subset, say H, of $\delta 2^n$ inputs. We construct a circuit C' of size g' that predicts f with probability at least ϵ on the hard-core set H. This will contradict the hypothesis that f is δ-hard for circuits of size g.

Partition the set of nk-bit strings into $S_0, S_1, S_2, \ldots, S_k$, where S_t consist of those $(x_1, \ldots, x_k)$ in which exactly t of the x_i's lie in H. At most $(1-\delta)^k$ fraction of inputs lie in S_0, so C must predict $f^{\oplus k}$ with probability at least $(1+\epsilon)/2$ on inputs from the other classes. Averaging shows there is a $t \geq 1$ such that C predicts $f^{\oplus k}$ with probability at least $(1+\epsilon)/2$ on inputs randomly chosen from S_t. Using nonuniformity, assume we know such a t.

Let x be a random input from H that will be an input to C'. Pick $a_1, a_2, \ldots, a_{t-1}$ randomly from H and $b_1, \ldots, b_{k-t-1}$ randomly from the complement of H. Randomly permute the k inputs $a_1, a_2, \ldots, a_{t-1}, x, b_1, \ldots, b_{k-t-1}$ to get an nk-bit string z; note that z is a random input from S_t. Apply C on z. With probability at least $(1+\epsilon)/2$, we have $C(z) = f^{\oplus k}(z)$.

By averaging, there is a way to fix $a_1, a_2, \ldots, a_{t-1}, b_1, \ldots, b_{k-t-1}$ and their random permutation such that the circuit still predicts $f^{\oplus k}$ with probability at least $(1+\epsilon)/2$ when x is randomly chosen from H. Using nonuniformity again, we hardwire these choices into the circuit. Finally, we add a NOT gate at the top iff $b = 1$ where $b = f(a_1) \oplus \cdots \oplus f(a_{t-1}) \oplus f(b_1) \oplus \cdots \oplus f(b_{k-t-1})$. This gives a circuit C' with $g' + 1$ gates (removing the "+1" is an exercise) which on input x produces $C(z) \oplus b$ as output. By definition of $f^{\oplus k}$,

$$f(x) = f^{\oplus k}(z) \oplus f(a_1) \oplus \cdots \oplus f(a_{t-1}) \oplus f(b_1) \oplus \cdots \oplus f(b_{k-t-1}),$$

so we conclude that $C'(x) = f(x)$ whenever $C(z) = f^{\oplus k}(z)$, which happens for at least $(1+\epsilon)/2$ fraction of $x \in H$. Thus C' predicts f on its hardcore set with probability at least $(1+\epsilon)/2$, which is a contradiction. □

16.2 Proof of Impagliazzo's Lemma

We will need the notion of a *fuzzy* subset of inputs (Impagliazzo calls it a "measure"). In a usual subset, an element is either in or out; a fuzzy set allows the element to be fractionally in or out. A fuzzy subset M of $\{0,1\}^n$ is a function that assigns a fraction $M(x) \in [0,1]$ to each $x \in \{0,1\}^n$. The *size* of M, written $|M|$, is $\sum_x M(x)$, and its *relative size*, denoted $\mu(M)$, is $1/2^n \sum_x M(x)$. The *distribution induced by* M, denoted D_M, is defined by $D_M(x) = M(x)/|M|$.

To quantify how well a circuit computes a function f on a fuzzy set M we look at $\Pr_{x \in D_M}[f(x) = C(x)]$. Alternatively, we can look at the *advantage* of the circuit, which is $Adv_M(C, f) = \Pr_{x \in D_M}[f(x) = C(x)] - \Pr_{x \in D_M}[f(x) \neq C(x)]$. We assume without loss of generality that this quantity is positive since either the circuit or its negation agrees with f at least half the time. Since $\Pr_{x \in D_M}[f(x) = C(x)] + \Pr_{x \in D_M}[f(x) \neq C(x)] = 1$ the two quantities are related by

$$\Pr_{x \in D_M}[C(x) = f(x)] = \frac{1 + Adv_M(C, f)}{2} \tag{16.3}$$

We say that a function f is ϵ-hard-core for a fuzzy set M if $\Pr_{x \in D_M}[C(x) = f(x)] < (1+\epsilon)/2$, in other words, $Adv_M(C, f) < \epsilon$.

Hard-core fuzzy sets can be changed easily into hard-core sets by random sampling (see the Problems), so it suffices to prove Impagliazzo's lemma for fuzzy sets.

LEMMA 16.3 (IMPAGLIAZZO)
Suppose boolean function f is δ-hard for size g on n-bit strings. Then for every $\epsilon > 0$ there is a fuzzy set M of size $\geq \delta 2^{n-1}$ such that f is ϵ-hard core on M for circuits of size $g' = g\epsilon^2\delta^2/4$.

PROOF: We define a game between two players, called SET and CIRCUIT. They take turns to make moves; SET makes the first move. The ith move consists of SET presenting a fuzzy set M_i of size at least $\delta 2^n$, and CIRCUIT responding with a circuit C_i of size at most g'. The game stops if either of two things happen.

- If C_i does not satisfy $Adv_{M_i}(C_i, f) \geq \epsilon$, then SET wins.
- If $D_i = Maj(C_1, C_2, \ldots, C_i)$ satisfies $\Pr_x[f(x) = D_i(x)] > 1 - \delta$ then CIRCUIT wins. Here $Maj(C_1, C_2, \ldots, C_i)$ is the circuit that, given an input x, evaluates $C_1, C_2, \ldots, C_i$ on x and outputs the majority answer, and answering 1 if there is a tie.

Clearly, SET has a winning strategy iff f is ϵ-hard-core on some fuzzy set of size at least $\delta 2^n$. The proof will describe such a winning strategy. The next observation is the only place where we will use the hardness of f.

OBSERVATION: *If f is δ-hard on the uniform distribution for circuits of size g, then* CIRCUIT *cannot win in fewer than $g/2g'$ moves.*

This follows just from noticing that majority of i bits is computable with a circuit of size $O(i)$, so $D_i = Maj(C_1, C_2, \ldots, C_i)$ is a circuit of size $g'i + O(i) < 2g'i$, and hence it cannot predict f with probability more than δ on the uniform distribution so long as $2g'i < g$.

The observation above gives comfort to SET, since he has $g/2g' \geq 2/\epsilon^2\delta^2$ moves in which to fashion a victory. Now we describe a strategy that allows SET to do precisely that.

Let $R_i(x) = 1$ if $C_i(x) = f(x)$ and -1 otherwise, and let $N_i(x) = \sum_i R_i(x)$. Thus $N_i(x)$ is the margin by which the majority decision of $C_1, C_2, \ldots, C_i$ is correct on x, and $N_i(x)$ is positive iff this majority decision is correct. Denoting $\gamma = \epsilon\delta$, the following is how SET chooses M_{i+1}:

$$M_{i+1}(x) = \begin{cases} 0 & \text{if } N_i(x) \geq 1/\gamma. \\ 1 & \text{if } N_i(x) \leq 0 \\ 1 - \gamma N_i(x) & \text{otherwise} \end{cases} \tag{16.4}$$

Note that if the game is not over after i rounds, then $N_i(x) \leq 0$ on at least δ fraction of x's, which implies $\mu(M_{i+1}) \geq \delta$. Thus M_{i+1} is a valid move for player SET.

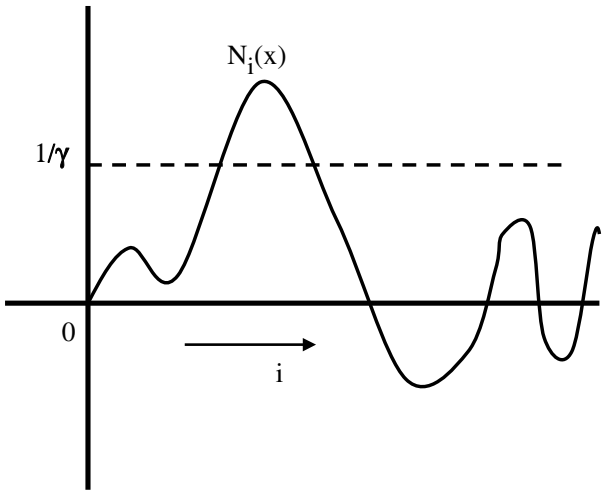

Figure 16.1: $N_i(x)$ as a function of i. Note that $M_{i+1}(x)$ is nonzero iff $N_i(x) < 1/\gamma$.

Define $A_t(x) = \sum_{i \le t} R_i(x) M_i(x)$. If the game has not stopped in t steps, we have the following lowerbound

$$\begin{aligned} \sum_x A_t(x) &= \sum_x \sum_{i \le t} R_i(x) M_i(x) = \sum_{i \le t} \sum_x R_i(x) M_i(x) \\ &= \sum_{i \le t} |M_i| \, Adv_{M_i}(C_i, f) \\ &\ge t\epsilon\delta 2^n = \gamma t 2^n \end{aligned} \tag{16.5}$$

Now we give an upperbound.

CLAIM: For any x, $A_t(x) \le 1/\gamma + \gamma t/2$.
PROOF: For any a, let $i_1, i_2, \ldots,$ be the respective times steps when $N_i(x) = a$. For now assume $a > 0$. Since $N_i(x)$ is a cumulative count of $R_i(x)$, the following "interleaving" property must hold. If $N_i(x)$ increased at $i = i_1$, namely, $R_{i_1 - 1}(x) = 1$, then $N_i(x)$ must have decreased at step $i_2 - 1$, namely, $R_{i_2 - 1}(x) = -1$. (See Figure 16.1.) The reverse is true if $N_i(x)$ decreased at $i = i_1$. Thus in the sum $A_t(x) = \sum_{i \le t} R_i(x) M_i(x)$ the the terms for $i = i_1$ and $i = i_2 - 1$ contribute $R_{i_1}(x) M_{i_1}(x) + R_{i_2 - 1}(x) M_{i_2 - 1}(x)$, which is at most $|M_{i_1}(x) - M_{i_2 - 1}(x)|$. Since $|N_{i_1}(x) - N_{i_2 - 1}(x)| \le 1$, the definition in (16.4) shows that $|M_{i_1}(x) - M_{i_2 - 1}(x)| \le \gamma$. Thus the terms in the sum can be paired so that each pair contributes at most γ; with possibly one terms left unpaired for each $a \le 1/\gamma$. Thus the total is at most $1/\gamma + \gamma i t/2$. □

The Claim shows that $\sum_x A_t(x)$ cannot rise above $2^n(1/\gamma + \gamma t/2)$ whereas we already showed it is at least $\gamma t 2^n$, so it follows that the game cannot continue past $t = 2/\gamma^2$. Since CIRCUIT cannot win in fewer than $g/2g' > 2/\gamma^2$ moves, we conclude that SET must have won. □

16.3 An approach to hardness amplification using polynomials

As already noted, derandomization calls for a different type of hard functions than cryptography does. Yao's XOR lemma was geared to cryptography, and can be replaced by a simpler algebraic technique for obtaining hard functions. These hard functions suffice for derandomization.

To be described: Sudan-Trevisan-Vadhan approach for constructing hard functions using polynomial extensions

Problems

§1 Show that if a boolean function on n bits is ϵ-hard core on a set of size $\delta 2^n$ for circuits of size g, then f is $(1 - \delta/2 + \epsilon\delta/2)$-hard on the uniform distribution for circuits of size g.

§2 Show that if f is ϵ-hard core on a fuzzy set M for circuits of size g then there is a set S of size $|M|/2$ such that f is 2ϵ-hard core on S for circuits of size $g'/2$. (Hint: pick S randomly according to distribution D_M.)