Randomized algorithms and non-uniform lower bounds

Sylvain Perifel

Cours MPRI, 2010–2011

Note: parts of this text are inspired by the book of Arora and Barak *Computational Complexity: a modern approach.*

Randomized algorithms: the program can toss a coin $\rightsquigarrow$ instruction `rand` with independent tosses.

Can randomness speed up computation?

Some examples:

- Thanks to randomness, for some problems the worst case can be reduced to the average case:
  - **Quicksort** (deterministic worst case: $\Omega(n^2)$; randomized: in all cases, expected cost $O(n \log n)$);
  - **Problem**: in a word $x \in \{a, b\}^\star$ containing as many a's as b's, find the position of an a. Deterministic worst case: $\simeq n/2$ trials; randomized (test positions at random): $\Pr(i \text{ trials}) = 2^{-i} \Rightarrow$ expected number of trials $\sum_{i=1}^{+\infty} i2^{-i} = O(1)$.
- **Problem**: compute the volume of a (bounded) convex body in $\mathbb{R}^n$ given as a black box for membership.

  This can be done with error ϵ with a randomized algorithm running in time $\text{poly}(n, 1/\epsilon)$ (Dyer, Frieze, Kannan 1991).

  However, no *deterministic* polynomial-time algorithm can give an upper bound and a lower bound satisfying $\sup / \inf \leq (cn/\log n)^n$ for some fixed $c > 0$ (Bárány and Fredi, 1987).

$\rightsquigarrow$ Randomness seems to speed up computations... However:

Generic techniques for derandomization: d-wise independence, methods of conditional probabilities and expectations, pessimistic estimators...

Ad-hoc derandomizations: Primes (AKS 2002), Undirected reachability (Reingold 2004)...

Almost all "natural" problems having polynomial-time randomized algorithms turn out to have polynomial-time deterministic algorithms.

$\rightsquigarrow$ Can every problem be derandomized?

Goal of this course: formalize this question, draw links with other areas of computational complexity, be convinced that the answer is "yes".

An **obstacle**: problem Integer nullity (make a picture, explain the ideas why in BPP and why not yet in P)

How to derandomize? Make formal an old idea: a hard problem appears random to a machine with limited computational power.

1 Probabilistic classes

Notation: if A is a language and x a word, $[x \in A] = 1$ if $x \in A$, 0 otherwise.
Definition (BPP) – A language A is in the class BPP if there exist a polynomial $p(n)$ and a deterministic polynomial-time Turing machine (TM) $M(x, y)$ with $|y| = p(|x|)$ such that for all x, $\Pr_r(M(x, r) = [x \in A]) \geq 2/3$.

Definition (RP) – A language A is in the class RP if there exist a polynomial $p(n)$ and a deterministic polynomial-time TM $M(x, y)$ with $|y| = p(|x|)$ such that for all x, $x \in A \Rightarrow \Pr_r(M(x, r) = 1) \geq 2/3$ and $x \notin A \Rightarrow \Pr_r(M(x, r) = 0) = 1$ (i.e. $\forall r, M(x, r) = 0$).

Remark: $\mathsf{RP} \subseteq \mathsf{BPP}$

Proposition – The error in RP can be made $\leq 2^{-\mathrm{poly(n)}}$.

Proof. Let $A \in \mathsf{RP}$ and M and p the corresponding TM and polynomial. Let $q(n)$ be a polynomial: we show that the error can be reduced to $2^{-q(n)}$. Let $M'(x, r')$, with $r' = r_1, \ldots, r_{q(n)}$ and $|r_i| = p(n)$, be the following machine: simulate successively $M(x, r_1), \ldots, M(x, r_{q(n)})$; if one of the result is 1, then accept x, otherwise reject x. If $x \notin A$, then it is rejected. If $x \in A$, then $\Pr_{r'}(M'(x, r') = 0) \leq (1/3)^{q(n)} \leq 2^{-q(n)}$. □

The same is true for BPP: repeat $c.p(n)$ times (for some constant c) and take the majoritary answer. Chernoff bounds for the analysis.
Definition – Integer nullity (IN):
Input: an arithmetic circuit C (with gates + and $\times$) computing an integer N from the constant -1.
Question: is N equal to 0?

Proposition – IN is in coRP (that is, ${}^c\mathsf{IN} \in \mathsf{RP}$).

Proof. The idea is to evaluate C modulo m for a randomly chosen integer m.

Suppose $N \neq 0$. Let n be the number of gates of C. Then $|N| \leq 2^{2^n}$, therefore N has at most 2^n prime factors. Choose a number $m \in [2, 2^{n^2}]$ at random. By the prime number theorem, there are $\geq c2^{n^2}/n^2$ primes in this interval (for some absolute constant $c > 0$). Hence, $\Pr(m \text{ is prime}) \geq c/n^2$ and, if m is prime, $\Pr(m \text{ divides } N) \leq 2^n/(c2^{n^2}/n^2) = (n^2/c)2^{n-n^2}$. Therefore $\Pr((N \bmod m) = 0) = \Pr((N \bmod m) = 0 | m \text{ prime}).\Pr(m \text{ prime}) + \Pr((N \bmod m) = 0 | m \text{ not prime}).\Pr(m \text{ not prime}) \leq (n^2/c)2^{n-n^2} + (1 - c/n^2) \leq 1 - d/n^2$ for some $d > 0$. By choosing $\Theta(n^2)$ moduli m_i: $\Pr(\bigwedge_i((N \bmod m_i) = 0)) \leq 1/3$.

If now $N = 0$, then $(N \bmod m_i) = 0$ for all i. We deduce the coRP algorithm: choose $\Theta(n^2)$ integers $m_i \in [2, 2^{n^2}]$ at random. Evaluate C modulo m_i. If $N \bmod m_i \neq 0$ for some of them, then reject, otherwise accept. □

Proposition – $\mathsf{RP} \subseteq \mathsf{NP}$.

Proof. Let $A \in \mathsf{RP}$ and M the corresponding machine. Let M' be the following nondeterministic TM: $M'(x)$ guesses r of size $p(n)$ and simulates $M(x, r)$. If there is an accepting path then $\Pr_r(M(x, r) = 0) < 1$, therefore $x \in A$. Otherwise, $\forall r, M(x, r) = 0$ and $x \notin A$. □

Reminder: $A \in \Sigma_2^p$ if there exists a polynomial $p(n)$ and a language $B \in \mathsf{P}$ such that $x \in A \iff \exists y \in \{0,1\}^{p(n)} \forall z \in \{0,1\}^{p(n)}, (x, y, z) \in B$. Or, equivalently, $\Sigma_2^p = \mathsf{NP}^{\mathsf{NP}}$.
Proposition – $\mathsf{BPP} \subseteq \Sigma_2^p$.

Proof. Let $A \in \mathsf{BPP}$ and M be the corresponding TM with error $< 2^{-n}$. Denote by $m = n^{O(1)}$ the number of random bits used. For each x, let $S_x = \{r \in \{0,1\}^m \ : \ M(x, r) = 1\}$. Then $x \in A \Rightarrow |S_x| \geq (1 - 2^{-n})2^m$ and $x \notin A \Rightarrow |S_x| \leq 2^{-n}2^m$.

For a set $S \subseteq \{0,1\}^m$ and $u \in \{0,1\}^m$, denote by $S + u$ the set $\{u + v \ : \ v \in S\}$ where $+$ is the vector addition modulo 2.

If $|S| \leq 2^{-n}2^m$ then for all vectors $u_1, \ldots, u_m$, $\bigcup_i (S+u_i) \neq \{0,1\}^m$ because $m|S| < 2^m$.

If $|S| \geq (1-2^{-n})2^m$, let us prove that $\Pr(\bigcup_i (S+u_i) = \{0,1\}^m) > 0$, where the u_i are taken independently at random. For a fixed $v \in \{0,1\}^m$, we have $\Pr_u(v \notin \bigcup_i (S+u_i)) = \Pr_u(\bigwedge_i (v \notin (S+u_i))) = (2^{-n})^m = 2^{-nm}$. Hence $\Pr_u(\{0,1\}^m \neq \bigcup_i (S+u_i)) \leq 2^{-n}$, therefore there exist u_i such that $\{0,1\}^m = \bigcup_i (S+u_i)$.

Now, $x \in A$ can be rewritten as the following Σ_2^p statement:

$$\exists u_1, \ldots, u_m, \forall v, (v \in \bigcup_i (S+u_i)). \qquad \square$$

2 Circuits and advices

Picture. Why circuits: in order to "look inside the structure of the computation".

Definition (Boolean circuits) – A Boolean circuit is a DAG whose vertices have indegree 0, 1 or 2 and are called gates. Gates of indegree 0 are called inputs and are labelled by $x_1, \ldots, x_n$; gates of indegree 1 are labelled by $\neg$; gates of indegree 2 are labelled by $\vee$ or $\wedge$. Exactly one gate has outdegree 1: this gate is called the output. The size of the circuit is the number of vertices.

Lemma – The number of circuits of size s is $\leq (3s)^{2s}$.

Proof. Let us first count the number of circuits with n inputs. Order the gates such that the n first gates are the input gates. Each non-input gate has at most 2 inputs among $s-1$, which makes $\leq (s-1)^2$ possibilities, and has a label among $\{\wedge, \vee, \neg\}$ (3 possibilities). There are $s-n$ such gates, therefore the number of circuits is at most $(3(s-1)^2)^{s-n} \leq (3(s-1)^2)^s$.

Now, taking into account an arbitrary number of inputs, the number of circuits is $\leq s \times (3(s-1)^2)^s \leq (3s)^{2s}$. $\square$

Definition – If $x \in \{0,1\}^n$ and C a circuit with n inputs, the value of $C(x)$ is the value of the output gate, where the value of a gate is defined inductively:

- the value of the i-th input gate is x_i;
- the value of a $\neg$ gate is the negation of the value of its input;
- the value of a $\wedge$ (resp. $\vee$) gate is the conjunction (resp. disjunction) of the values of its inputs.

Fixed input size $\rightsquigarrow$ Deciding a language $A \subseteq \{0,1\}^\star$ requires a family (C_n) of circuits, C_n having n inputs (nonuniformity). The language recognized by (C_n) is $\{x \in \{0,1\}^\star : C_{|x|}(x) = 1\}$.

Definition ($\mathsf{P/poly}$) – The class $\mathsf{P/poly}$ is the set of languages recognized by a family of polynomial-size circuits (i.e. there exists a polynomial $p(n)$ such that $|C_n| \leq p(n)$).

Remark: $\mathsf{P/poly}$ is an uncountable class!

Definition – A family (C_n) of circuits is called *uniform* if there exists a TM which, on input n, computes the encoding of C_n in time polynomial in n.

Proposition – The class P is the set of languages recognized by a *uniform* family of circuits of polynomial size.

Proof. If A has a uniform family of circuits: $M(x)$ builds the circuit $C_{|x|}$ and simulates $C_{|x|}(x)$.

In the other direction, if $A \in \mathsf{P}$ then the machine can be simulated by circuits of polynomial size and the construction is uniform. $\square$

Proposition – $\mathsf{P/poly}$ is the set of languages recognized by a polynomial-time TM with polynomial-size advice: i.e. $A \in \mathsf{P/poly}$ iff there exists a polynomial $p(n)$, a sequence (a_n) of words of size $p(n)$ and a language $B \in \mathsf{P}$ such that $x \in A \iff (x, a_{|x|}) \in B$.

Proof. If $A \in \mathsf{P/poly}$ with circuits C_n, then we give as advice the circuit C_n: the language B is $\{(x, C) : C(x) = 1\} \in \mathsf{P}$.In the other direction, you build a circuit simulating a machine M for B (level i of the circuit corresponds to the content of the tape at step i) and hard-wire the advice a_n. □

Proposition – $\mathsf{BPP} \subset \mathsf{P/poly}$.

Proof. Let $A \in \mathsf{BPP}$. There exists a machine $M(x, y)$ with $|y| = |x|^{O(1)}$ such that for all $x \in \{0,1\}^n$, $\Pr_r(M(x, r) \neq [x \in A]) < 2^{-n}$. Therefore $\Pr_r(\exists x \in \{0,1\}^n, M(x, r) \neq [x \in A]) < 1$, hence there exists r_0 suitable for every x (i.e. $\forall x, M(x, r_0) = [x \in A]$). It is enough to give r_0 as advice to the machine M. □

Picture of complexity classes $\mathsf{L}, \mathsf{P}, \mathsf{NP}, \mathsf{BPP}, \mathsf{RP}, \Sigma_2^p, \mathsf{PH}, \mathsf{PSPACE}, \mathsf{EXP}, \mathsf{NEXP}, \mathsf{P/poly}, \ldots$
Question : do exponential-time problems have polynomial-size circuits? I.e. "$\mathsf{EXP} \subset \mathsf{P/poly}$?"
Proposition – For all $c > 0$, if $d > c$ then there exists $A \in \mathsf{DTIME}(2^{n^d})$ such that A does not have circuits of size n^c.

Proof. Let us fix n and define $A^{=n}$. Let $x_0, x_1, \ldots$ be the words of $\{0,1\}^n$ in lexicographic order. We will successively decide whether $x_i \in A$ for $i = 0, 1, \ldots$

Let V_0 be the set of all circuits of size n^c. We decide that $x_0 \in A$ iff more than half of the circuits of V_0 reject x_0. Let V_1 be the set of circuits of V_0 that are correct on x_0, that is, $C(x_0) = [x_0 \in A]$. Then we decide that $x_1 \in A$ iff more than half of the circuits of V_1 reject x_1. Let V_2 be the set of circuits of V_1 that are correct on x_1, that is, $C(x_1) = [x_1 \in A]$. We go on like this: $x_i \in A$ iff more than half of the circuits of V_i reject x_i, and V_{i+1} is the set of circuits of V_i that are correct on x_i, that is, $C(x_i) = [x_i \in A]$.

Then $|V_{i+1}| \leq |V_i|/2$. Furthermore, by the lemma, the number of circuits of size n^c is at most $(3n^c)^{2n^c}$, hence for $i > 2n^c \log(3n^c)$, we have $V_i = \emptyset$: we then decide that $x_i \notin A$.

Suppose $A^{=n}$ is recognized by a circuit C of size n^c: let i_0 be the (unique) i such that $C \in V_i \setminus V_{i+1}$. Then $C(x_i) \neq [x_i \in A]$, a contradiction.

Let us now show that $A \in \mathsf{DTIME}(2^{n^d})$. Each step requires to simulate at most $(3n^c)^{2n^c}$ circuits of size n^c, which requires time $\leq n^c(3n^c)^{2n^c}$, and memorize which ones give the minority answer. As a whole, the $2n^c \log(3n^c)$ steps take time $O((3n^c)^{2n^c+3}) = O(2^{n^d})$ for $d > c$. □

3 Unpredictability implies derandomization

Questions : "$\mathsf{BPP} = \mathsf{P}$?", "$\mathsf{BPP} \neq \mathsf{EXP}$?"
Definition (Unpredictability) – Let $m : \mathbb{N} \to \mathbb{N}$. A function $f : \{0,1\}^\star \to \{0,1\}^\star$ is $m(n)$-unpredictable if

- for all $x \in \{0,1\}^\star$, $|f(x)| = m(|x|)$ and
- for all n, for every circuit C of size $\leq m(n)^2$ and for every $i \in [0, m(n) - 1]$, we have:

$$\Pr_{z \in \{0,1\}^n : r = f(z)}\big(C(r_1, \ldots, r_i) = r_{i+1}\big) < 1/2 + 1/(10m(n)).$$

Remark: The identity map $\mathrm{id} : \{0,1\}^n \to \{0,1\}^n$ is n-unpredictable.

Theorem (Yao) – If $f : \{0,1\}^\star \to \{0,1\}^\star$ is $m(n)$-unpredictable and computable in time $t(n)$, then $\mathsf{BPTIME}\big(m(u(n))\big) \subseteq \mathsf{DTIME}\big(2^{u(n)}(t(u(n)) + m(u(n)))\big)$ for any time-constructible function $u(n)$.

Proof. For simplicity, define $N = u(n)$. Let A be an algorithm for a language $L \in \mathsf{BPTIME}\big(m(N)\big)$. The idea is to replace the random bits $r \in \{0,1\}^{m(N)}$ of A by $f(z)$, where z runs through all strings of

length N, and to take the majority answer. This amounts to enumerate all $z \in \{0,1\}^N$ and for each of them, to compute $f(z)$ and to simulate $A(x, f(z))$, which gives a deterministic algorithm D running in time $2^N(t(N) + m(N))$.

Let us now show that this algorithm is correct. Suppose by contradiction that D makes a mistake on some $x_0 \in \{0,1\}^n$, that is,

$$\Pr_{z\in\{0,1\}^N}(A(x_0, f(z)) = [x \notin L]) \geq 1/2,$$

whereas, by definition of A, we have:

$$\Pr_{r\in\{0,1\}^{m(N)}}(A(x_0, r) = [x \notin L]) < 1/3.$$

We will construct a circuit C of size $\leq m(N)^2$ such that there exists $i \in [0, m(N)^2 - 1]$ satisfying

$$\Pr_{z\in\{0,1\}^N : r=f(z)}(C(r_1, \ldots, r_i) = r_{i+1}) > 1/2 + 1/(10m(N)).$$

Let us first design a probabilistic algorithm B, depending on x_0 and on $[x_0 \in L]$, before turning it into a circuit. Let $z \in \{0,1\}^N$ and $r = f(z) \in \{0,1\}^{m(N)}$: on input $r_1, \ldots, r_i$, B chooses $r'_{i+1}, \ldots, r'_{m(N)}$ at random and outputs the bit a_{i+1} defined by:

$$\begin{cases} a_{i+1} = r'_{i+1} & \text{if } A(x_0, r_1 \ldots r_i r'_{i+1} \ldots r'_{m(N)}) = [x_0 \notin L] \\ a_{i+1} = 1 - r'_{i+1} & \text{otherwise.} \end{cases}$$

The intuition is that, since A makes a mistake on x_0 with bits $f(z)$, it should have more propensity to output the wrong answer if $r'_{i+1} = r_{i+1}$. The running time of B is $O(m(N))$. We want to show that $\Pr_{z,r'}(a_{i+1} = r_{i+1}) > 1/2 + 1/(10m(N))$. Let $p_i = \Pr_{z,r' : r=f(z)}(A(x_0, r_1 \ldots r_i r'_{i+1} \ldots r'_{m(N)}) = [x_0 \notin L])$: then $p_0 < 1/3$ and $p_{m(n)} \geq 1/2$. We have:

$$\begin{aligned} \Pr_{z,r'}(a_{i+1} = r_{i+1}) &= \Pr_{z,r' : r=f(z)}\big(A(x_0, r_1 \ldots r_i r'_{i+1} \ldots r'_{m(N)}) = [x_0 \notin L] \wedge r'_{i+1} = r_{i+1}\big) + \\ &\quad \Pr_{z,r' : r=f(z)}\big(A(x_0, r_1 \ldots r_i r'_{i+1} \ldots r'_{m(N)}) = [x_0 \in L] \wedge r'_{i+1} \neq r_{i+1}\big) \\ &= \Pr\big(A(x_0, rr') = [x_0 \notin L] \mid r'_{i+1} = r_{i+1}\big)\Pr(r'_{i+1} = r_{i+1}) + \\ &\quad \Big(1 - \Pr\big(A(x_0, rr') = [x_0 \notin L] \mid r'_{i+1} \neq r_{i+1}\big)\Big)\Pr(r'_{i+1} \neq r_{i+1}) \end{aligned}$$

But $\Pr(r'_{i+1} = r_{i+1}) = \Pr(r'_{i+1} \neq r_{i+1}) = 1/2$ and we have $\Pr\big(A(x_0, rr') = [x_0 \notin L] \mid r'_{i+1} = r_{i+1}\big) = \Pr\big(A(x_0, r_1 \ldots r_{i+1} r'_{i+2} \ldots r'_{m(N)}) = [x_0 \notin L]\big) = p_{i+1}$, hence:

$$= 1/2\Big(1 + p_{i+1} - \Pr\big(A(x_0, rr') = [x_0 \notin L] \mid r'_{i+1} \neq r_{i+1}\big)\Big).$$

Therefore

$$\Pr\big(A(x_0, rr') = [x_0 \notin L] \mid r'_{i+1} \neq r_{i+1}\big) = 1 + p_{i+1} - 2\Pr(a_{i+1} = r_{i+1}).$$

Furthermore,

$$\begin{aligned} p_i &= \Pr\big(A(x_0, rr') = [x_0 \notin L]\big) \\ &= 1/2\Big(\Pr\big(A(x_0, rr') = [x_0 \notin L] \mid r'_{i+1} = r_{i+1}\big) + \Pr\big(A(x_0, rr') = [x_0 \notin L] \mid r'_{i+1} \neq r_{i+1}\big)\Big) \\ &= 1/2\big(p_{i+1} + 1 + p_{i+1} - 2\Pr(a_{i+1} = r_{i+1})\big) \\ &= 1/2 + p_{i+1} - \Pr(a_{i+1} = r_{i+1}) \end{aligned}$$

hence

$$\Pr(a_{i+1} = r_{i+1}) = 1/2 + p_{i+1} - p_i.$$

Since $\sum_{i=0}^{m(N)-1}(p_{i+1} - p_i) = p_{m(N)} - p_0 > 1/6$, there exists i such that $p_{i+1} - p_i > 1/\big(6m(N)\big)$, hence $\Pr_{z,r'}(a_{i+1} = r_{i+1}) > 1/2 + 1/\big(10m(N)\big)$.

Therefore there exists a choice r'_0 of r' such that $\Pr_z(a_{i+1} = r_{i+1}) > 1/2 + 1/(10m(N))$. Now, our circuit C merely simulates $B(x_0, r'_0)$: its size is $m(N)^2$ (since the running time of B is $m(N)$) and

$$\Pr_{z\in\{0,1\}^N : r=f(z)}\big(C(r_1,\ldots,r_i) = r_{i+1}\big) > 1/2 + 1/\big(10m(N)\big),$$

a contradition with the unpredictability of f. □

Corollary 1 –

- If, for some $\epsilon > 0$, there exists a $2^{\epsilon n}$-unpredictable function $f : \{0,1\}^\star \to \{0,1\}^\star$ in $\mathsf{DTIME}(2^{O(n)})$, then $\mathsf{BPP} = \mathsf{P}$.
- If, for some $\epsilon > 0$, there exists a 2^{n^ϵ}-unpredictable function $f : \{0,1\}^\star \to \{0,1\}^\star$ in $\mathsf{DTIME}(2^{O(n)})$, then $\mathsf{BPP} \subseteq \cup_k \mathsf{DTIME}(2^{(\log n)^k})$.
- If there is $\alpha > 0$ such that for all $c > 0$, there exists an n^c-unpredictable function $f_c : \{0,1\}^\star \to \{0,1\}^\star$ in $\mathsf{DTIME}(2^{\alpha n})$, then $\mathsf{BPP} \subseteq \cap_\epsilon \mathsf{DTIME}(2^{n^\epsilon})$.

Proof.

- For each k, consider $u(n) = k/\epsilon \log n$: then $m(u(n)) = n^k$. Let $\alpha > 0$ be such that $f \in \mathsf{DTIME}(2^{\alpha n})$. By the theorem, $\mathsf{BPTIME}(n^k) \subseteq \mathsf{DTIME}(n^{k/\epsilon} n^{\alpha k/\epsilon} n^k)$.
- For each k, consider $u(n) = (k \log n)^{1/\epsilon}$: then $m(u(n)) = n^k$. By the theorem, $\mathsf{BPTIME}(n^k) \subseteq \mathsf{DTIME}(2^{(\log n)^{O(1)}})$.
- Fix ϵ and k. Take $u(n) = n^\epsilon$ and $c = k/\epsilon$. Then by the theorem $\mathsf{BPTIME}(n^k) \subseteq \mathsf{DTIME}(2^{O(n^\epsilon)})$.

□

4 Average case hardness implies unpredictability

Definition (Average-case and worst-case hardness) – Let $f : \{0,1\}^n \to \{0,1\}$ and $\rho \in [0,1]$.

- The ρ-average-case hardness of f, denoted $\mathsf{H}^\rho_{\mathsf{avg}}(f)$, is the largest s such that for every circuit C of size $\leq s$, $\Pr_{x\in\{0,1\}^n}(C(x) = f(x)) < \rho$.
- The worst-case hardness of f is $\mathsf{H}_{\mathsf{wrs}}(f) = \mathsf{H}^1_{\mathsf{avg}}(f)$.
- The average-case hardness of f is $\mathsf{H}_{\mathsf{avg}}(f) = \max\{s \ : \ \mathsf{H}^{1/2+1/s}_{\mathsf{avg}}(f) \geq s\}$, that is, the largest s such that for every circuit C of size $\leq s$, $\Pr_{x\in\{0,1\}^n}(C(x) = f(x)) < 1/2 + 1/s$.
- If $f : \{0,1\}^\star \to \{0,1\}$ then $\mathsf{H}^\rho_{\mathsf{avg}}(f)$, $\mathsf{H}_{\mathsf{avg}}(f)$ and $\mathsf{H}_{\mathsf{wrs}}(f)$ are the functions depending on n defined accordingly.

Remark: For $f : \{0,1\}^\star \to \{0,1\}$, $\mathsf{H}_{\mathsf{wrs}}(f) \geq \mathsf{H}_{\mathsf{avg}}(f)$.

Exercise: Show that most functions $f : \{0,1\}^\star \to \{0,1\}$ have exponential average-case hardness (that is, $\mathsf{H}_{\mathsf{avg}}(f) = 2^{\Omega(n)}$).

Remark:

- $\mathsf{NP} \not\subset \mathsf{P/poly}$ iff $\mathsf{H}_{\mathsf{wrs}}(\mathrm{SAT}) = n^{\omega(1)}$.
- Since $\mathsf{BPP} \subset \mathsf{P/poly}$, if $f \in \mathsf{BPP}$ then $\mathsf{H}_{\mathsf{avg}}(f) = n^{O(1)}$.

Theorem (Nisan, Wigderson 1988) – Let $s(n)$ be a function such that there exists $d : \mathbb{N} \to \mathbb{N}$ computable in time 2^n satisfying $d(n) \leq n/48$ and $2^{2d(n)} < s(\sqrt{nd(n)/12})$. Suppose there exists $f : \{0,1\}^\star \to \{0,1\}$ in $\mathsf{DTIME}(2^{\alpha n})$ (for some constant $\alpha \geq 2$) such that $\mathsf{H}_{\mathsf{avg}}(f) \geq s(n)$. Then there exists a $2^{d(n)/2}$-unpredictable function $g \in \mathsf{DTIME}(2^{\alpha n})$.

Proof idea: use the hard function $f : \{0,1\}^\star \to \{0,1\}$ on "almost disjoint" sets of inputs to get an unpredictable function $g : \{0,1\}^\star \to \{0,1\}^\star$. For the proof we need some lemmas.

Definition (combinatorial design) – Let m and d be integers. A family $(I_1, \ldots, I_N)$ of subsets of $\{1, \ldots, n\}$ is an (m, d)-design if

- $m \geq 2d$ and $n = 12m^2/d$;
- for all i, $|I_i| = m$;
- $|I_i \cap I_j| \leq d$ for $i \neq j$.

Remark: The parameters m, n, d as above satisfy $n \geq 48d$ (since $m \geq d$) and $n \geq 24m$ (since $d \leq m/2$).

Lemma – Let $m \geq 2d$ and $n = 12m^2/d$. If $I_1, \ldots, I_k$ are arbitrary subsets of size m of $\{1, \ldots, n\}$, where $k < 2^{d/2}$, then there exists $J \subset \{1, \ldots, n\}$ such that $|J| = m$ and for all i, $|J \cap I_i| \leq d$.

Proof. Build a random subset J of $\{1, \ldots, n\}$ as follows: pick every element $x \in \{1, \ldots, n\}$ with probability $2m/n < d/(6m)$. Then $E(|J|) = 2m$ and by Chebyshev's inequality: $\Pr(|J| \geq m) \geq \Pr(||J| - E(|J|)| \leq m) \geq 1 - \Pr(||J| - E(|J|)| > m) > 1 - \mathrm{Var}(|J|)/(m^2) = 1 - n(2m/n)(1-2m/n)/m^2 > 1 - 2/m > 9/10$ for $m \geq 20$.

Furthermore, for every fixed i, $\Pr(|J \cap I_i| > d) < \binom{m}{d}(d/(6m))^d < (m^d/(d!))(d/m)^d/6^d \leq 1/2^d$ because $d! > (d/3)^d$ by Stirling, hence $(d^d/d!)/6^d < 3^d/6^d = 1/2^d$. Therefore $\Pr(\exists i, |J \cap I_i| > d) < 2^{d/2}/2^d < 1/10$ (for d large enough) and $\Pr(\forall i, |J \cap I_i| \leq d) > 9/10$. Altogether, $\Pr\big((\forall i, |J \cap I_i| \leq d) \wedge |J| \geq m\big) > 4/5 > 0$, hence there exists J such that $\forall i, |J \cap I_i| \leq d$ and $|J| \geq m$. Removing arbitrary elements from J to get $|J| = m$ preserves the property. □

Lemma – For all $m \geq 2d$ and $N \leq 2^{d/2}$, an (m, d)-design $(I_1, \ldots, I_N)$ exists and can be constructed in time $< 2^{24m^2/d}$.

Proof. A greedy algorithm will work. Choose the first set I_1 arbitrarily, e.g. $I_1 = \{1, \ldots, m\}$. Then at each subsequent step, find by exhaustive search in $\{1, \ldots, 12m^2/d\}$ a subset I_{i+1} satisfying $|I_{i+1}| = m$ and for all $j \leq i$, $|I_j \cap I_{i+1}| \leq d$. Such a set exists by the preceding lemma. The time needed to find one is $\binom{12m^2/d}{d} \leq 2^{12m^2/d}$. The whole algorithm runs in time $2^{d/2}2^{12m^2/d} < 2^{24m^2/d}$. □

Definition (Nisan-Wigderson generator) – Let $f : \{0,1\}^m \to \{0,1\}$ and $(I_1, \ldots, I_N)$ be an (m, d)-design. The function $f^{\mathsf{NW}} : \{0,1\}^{12m^2/d} \to \{0,1\}^N$ is defined by $f^{\mathsf{NW}}(x) = f(x|_{I_1})f(x|_{I_2}) \ldots f(x|_{I_N})$, where $x|_{I_i}$ is $x_{i_1}x_{i_2} \ldots x_{i_m}$ if $I_i = \{i_1 < i_2 < \ldots < i_m\}$. (The reference to the design is omitted for brevity) This definition extends to functions $f : \{0,1\}^\star \to \{0,1\}$.

Proof (of the theorem). Let us fix n and define $m = \sqrt{nd(n)/12}$, so that $12m^2/d = n$. By the first assumption, $m \geq 2d$. Let $(I_1, \ldots, I_{2^{d/2}})$ be an (m, d)-design and let $g = f^{\mathsf{NW}}$. Then $g : \{0,1\}^n \to \{0,1\}^{2^{d/2}}$. Furthermore, $g \in \mathsf{DTIME}(2^{\alpha n})$ since it is enough to build the design (time $2^{24m^2/d} = 2^{2n} \leq 2^{\alpha n}$) and to compute f on each subset (time $2^{d/2}.2^{\alpha m} < 2^{\alpha n}$).

Suppose by contradiction that g is not $2^{d/2}$-unpredictable, that is, there exist a circuit C of size $\leq 2^d$ and $i \in [0, 2^{d/2} - 1]$ such that

$$\Pr_{z \in \{0,1\}^n : r = g(z)}(C(r_1, \ldots, r_i) = r_{i+1}) \geq 1/2 + 1/(10.2^{d/2}).$$

We will obtain a contradiction with the hardness of f. Since $r_j = f(z|_{I_j})$, we have:

$$\Pr_{z \in \{0,1\}^n}\big(C(f(z|_{I_1}), \ldots, f(z|_{I_i})) = f(z|_{I_{i+1}})\big) \geq 1/2 + 1/(10.2^{d/2}).$$

Call z' the m variables $z|_{I_{i+1}}$, z'_j the $\leq d$ variables $z|_{I_j \cap I_{i+1}}$, z'' the $n-m$ variables $z|_{[1,n]\setminus I_{i+1}}$ and z''_j the $\geq m-d$ variables $z|_{I_j \setminus I_{i+1}}$. Then

$$\Pr_{z' \in \{0,1\}^m, z'' \in \{0,1\}^{n-m}} \big(C(f(z'_1, z''_1), f(z'_2, z''_2), \ldots, f(z'_i, z''_i)) = f(z')\big) \geq 1/2 + 1/(10.2^{d/2}).$$

Hence there exists a fixed z'' such that

$$\Pr_{z' \in \{0,1\}^m} \big(C(f(z'_1, z''_1), f(z'_2, z''_2), \ldots, f(z'_i, z''_i)) = f(z')\big) \geq 1/2 + 1/(10.2^{d/2}).$$

Now, since z'' is fixed, $f(z'_j, z''_j)$ depends only on the $\leq d$ variables z'_j and has therefore a circuit of size $d2^d$ (the circuit merely computes the truth table). Combined with C, this gives a circuit D of size $id2^d + 2^d \leq 2^{d/2} d2^d + 2^d$ such that

$$\Pr_{z' \in \{0,1\}^m} (D(z') = f(z')) \geq 1/2 + 1/(10.2^{d/2}).$$

This is a contradiction with the hardness of f as soon as $2^{d/2} d2^d + 2^d < s(m)$. Since $m = \sqrt{nd/12}$, it is enough that $2^{2d(n)} < s(\sqrt{nd/12})$, which is true by assumption. □

Corollary 2 –

- If there exists $f \in \mathsf{DTIME}(2^{O(n)})$ such that $\mathsf{H}_{\mathsf{avg}}(f) \geq 2^{\epsilon n}$ for some $\epsilon > 0$, then $\mathsf{BPP} = \mathsf{P}$.
- If there exists $f \in \mathsf{DTIME}(2^{O(n)})$ such that $\mathsf{H}_{\mathsf{avg}}(f) \geq 2^{n^\epsilon}$ for some $\epsilon > 0$, then $\mathsf{BPP} \subseteq \cup_k \mathsf{DTIME}(2^{(\log n)^k})$.
- If there is $\alpha > 0$ such that for all $c > 0$ there exists $f_c \in \mathsf{DTIME}(2^{\alpha n})$ such that $\mathsf{H}_{\mathsf{avg}}(f_c) \geq n^c$, then $\mathsf{BPP} \subseteq \cap_{\epsilon > 0} \mathsf{DTIME}(2^{n^\epsilon})$.

Proof.

- Wlog $\epsilon \leq 1$. For $d(n) = (\epsilon^2/50)n$, the function $s(n) = 2^{\epsilon n}$ satisfies the assumptions of the theorem: $s(\sqrt{nd/12}) = s((\epsilon/\sqrt{600})n) = 2^{(\epsilon^2/\sqrt{600})n} > 2^{(\epsilon^2/25)n} = 2^{2d(n)}$. Hence there exists a $2^{n/100}$-unpredictable function $g \in \mathsf{DTIME}(2^{O(n)})$. By Corollary 1, this implies $\mathsf{BPP} = \mathsf{P}$.
- Wlog $\epsilon < 2$. For $d(n) = 1/2(n/12)^{\epsilon/2}$, the function $s(n) = 2^{n^\epsilon}$ satisfies the assumptions of the theorem: there is $\eta \in]0,1[$ such that $s(\sqrt{nd/12}) = s(\eta n^{1/2+\epsilon/4}) = 2^{\eta^\epsilon n^{\epsilon/2+\epsilon^2/4}} > 2^{2n^{\epsilon/2}} = 2^{2d(n)}$. Hence there exists a $2^{O(n^{\epsilon/2})}$-unpredictable function $g \in \mathsf{DTIME}(2^{O(n)})$. By Corollary 1, this implies $\mathsf{BPP} \subseteq \cup_k \mathsf{DTIME}(2^{(\log n)^k})$.
- For $d(n) = (c/4) \log n$, the function $s(n) = n^c$ satisfies the assumptions of the theorem: $s(\sqrt{nd/12}) = ((c/48)n \log n)^{c/2} > n^{c/2} = 2^{2d(n)}$. Hence there exists an $n^{c/8}$-unpredictable function $g \in \mathsf{DTIME}(2^{2\alpha n})$. By Corollary 1, this implies $\mathsf{BPP} \subseteq \cap_\epsilon \mathsf{DTIME}(2^{n^\epsilon})$.

□

5 Worst case hardness implies average case hardness

Idea: consider $f : \{0,1\}^n \to \{0,1\}$ as a word of $\{0,1\}^{2^n}$ and suppose you have a good error-correcting code E. Then, if one can compute $E(f)$ correctly on a fraction of its inputs (say, 3/4), we can compute (recover) f exactly. Therefore, if f is hard in the worst case, then $E(f)$ is hard in the average case.

5.1 Error-correcting codes

Let Σ be a finite set.
Definition (Fractionnal Hamming distance) – For $x, y \in \Sigma^m$, the fractionnal Hamming distance of x, y is $\Delta(x, y) = |\{i \ : \ x_i \neq y_i\}|/m$.

Definition (Error-correcting code) – For $\delta \in [0, 1]$, an error-correcting code with distance δ is a function $E : \Sigma^n \to \Sigma^m$ such that $x \neq y \Rightarrow \Delta(E(x), E(y)) \geq \delta$.

This definition generalizes to functions $E : \Sigma^\star \to \Sigma^\star$ satisfying $|x| = |y| \Rightarrow |E(x)| = |E(y)|$.

Remark: Over $\{0, 1\}$, polynomial-sized codes exist for every distance $\delta < 1/2$ but not for $\delta = 1/2$, exponential-sized codes exist for $\delta = 1/2$, and no code exists for $\delta > 1/2$ (this is not true over Σ is $|\Sigma| > 2$).

Remark: If $E : \Sigma^n \to \Sigma^m$ is a code with distance δ and $x \in \Sigma^n$, one can recover x given any $z \in \Sigma^m$ satifsying $\Delta(z, E(x)) < \delta/2$: this is the nearest $y \in \mathrm{Im}(E)$.

For $x, y \in \{0, 1\}^n$, the inner product of x, y is $x \odot y = \sum_{i=1}^n x_i y_i \bmod 2$.
Definition (Walsh-Hadamard code) – If $x \in \{0, 1\}^n$, let $\mathsf{WH}(x) = z_0 z_1 \dots z_{2^n-1}$, where $z_j = x \odot j$ (here, j is seen as its binary encoding over $\{0, 1\}^n$). We see WH as a function from $\{0, 1\}^\star$ to $\{0, 1\}^\star$ where $|\mathsf{WH}(x)| = 2^{|x|}$.

Lemma – The function WH is a code with distance $1/2$.

Proof. Let $x, y \in \{0, 1\}^n$ be distinct on bit i. For all $u \in \{0, 1\}^{i-1}$ and $v \in \{0, 1\}^{n-i-1}$, either $x \odot u0v \neq y \odot u0v$ or $x \odot u1v \neq y \odot u1v$, therefore $x \odot j \neq y \odot j$ for one half of the j. □

For the definition of the Reed-Muller code, we need a lemma. By "individual degrees", we mean the degrees for each variable.
Lemma – Suppose k divides d and $d/k \leq |\Sigma|$. For any set $S \subseteq \Sigma$ of size $1 + d/k$, the application ϕ defined by $\phi(p) = f : x \in S^k \mapsto p(x)$ is a bijection between the set of k-variate polynomials $p : \Sigma^k \to \Sigma$ of individual degrees $\leq d/k$ (and thus of total degree $\leq d$) and the set of functions $f : S^k \to \Sigma$.

Proof. Let us show by induction on k that ϕ is indeed a bijection. It is enough to show that each f determines uniquely a polynomial p.

For $k = 1$, by interpolation $d + 1$ values characterize our polynomial p. For $k > 1$: $p(x_1, \dots, x_k) = \sum_{i=0}^{d/k} q_i(x_1, \dots, x_{k-1}) x_k^i$. The case $k = 1$ shows that for each $s \in S^{k-1}$, $q_i(s)$ is uniquely determined. Now, by induction, each q_i is uniquely determined. □

Definition (Reed-Muller code) – Let Σ be a finite field and $k \leq d$ be integers such that $d < |\Sigma|$ and k divides d. A k-variate polynomial

$$p(x_1, \dots, x_k) = \sum_{d_1 \leq d/k, \dots, d_k \leq d/k} c_{d_1, \dots, d_k} x_1^{d_1} \dots x_k^{d_k}$$

of individual degrees $\leq d/k$ (and thus of total degree $\leq d$) over Σ is characterized by the list of its values on all inputs from $\{\sigma_0, \dots, \sigma_{d/k}\}^k$, where $\sigma_i \in \Sigma$ are arbitrary distinct elements.

The function $\mathsf{RM} : \Sigma^{(1+d/k)^k} \to \Sigma^{|\Sigma|^k}$ is defined on such a list characterizing a polynomial p, by the list of values that p takes over Σ^k, that is, $\mathsf{RM}(p) = (p(a_1, \dots, a_k))_{a_1, \dots, a_k \in \Sigma}$.

Remark: This is not the usual definition of the Reed-Muller code: usually, the input is the list of coefficients of p. The present version prevents us from doing interpolation, which would not be efficient enough for decoding.

From now on, Σ is a finite field.
Lemma – The function RM is a code with distance $1 - d/|\Sigma|$.

Proof. Let $p \neq q$ be two k-variate polynomials of degree $\leq d$ over Σ: then $p - q$ is a nonzero k-variate polynomial of degree $\leq d$.

Claim (Schwartz-Zippel lemma) - If r is a nonzero k-variate polynomial of degree d over Σ, then $\Pr_{a \in \Sigma^k}(r(a) \neq 0) \geq 1 - d/|\Sigma|$

The proof is by induction on k. For $k = 1$, there are at most d roots and the claim follows. For $k > 1$: consider $r(x_1, \ldots, x_k) = \sum_i r_i(x_2, \ldots, x_k)x^i$ where r_i are $(k-1)$-variate polynomials of degree $\leq d - i$. Since r is nonzero, there is i such that r_i is nonzero: take i_0 to be the maximal such i. By induction hypothesis, $\Pr_{a_2,\ldots,a_k}(r_{i_0}(a_2, \ldots, a_k) \neq 0) \geq 1 - (d - i_0)/|\Sigma|$, and if $r_{i_0}(a_2, \ldots, a_k) \neq 0$ then $r(x_1, a_2, \ldots, a_k)$ is a nonzero univariate polynomial of degree i_0. Therefore $\Pr_{a_1,\ldots,a_k}(r(a_1, \ldots, a_k) \neq 0) \geq (1 - (d - i_0)/|\Sigma|)(1 - i_0/|\Sigma|) \geq 1 - d/|\Sigma|$. The claim is proven.

Therefore $p \neq q$ on a fraction $\geq 1 - d/|\Sigma|$ of the inputs Σ^k, which proves the lemma. □

Definition – Suppose that $|\Sigma|$ is a power of 2 (this will be the case in our applications).

- A code $E : \Sigma^n \to \Sigma^m$ can be seen as a function $E : \{0,1\}^{n \log |\Sigma|} \to \Sigma^m$ by encoding each input symbol from Σ using $\log |\Sigma|$ symbols from $\{0,1\}$. Similarly, a code $F : \{0,1\}^{\log |\Sigma|} \to \{0,1\}^t$ can be seen as a function $F : \Sigma \to \{0,1\}^t$.
- If $E : \Sigma^n \to \Sigma^m$ and $F : \{0,1\}^{\log |\Sigma|} \to \{0,1\}^t$ are two codes, their *concatenation* is the code $F \circ E : \{0,1\}^{n \log |\Sigma|} \to \{0,1\}^{mt}$ defined by $(F \circ E)(x) = F(z_1) \ldots F(z_m)$, where $z = E(x) \in \Sigma^m$.

Lemma – If $E : \Sigma^n \to \Sigma^m$ has distance δ_E and $F : \{0,1\}^{\log |\Sigma|} \to \{0,1\}^t$ has distance δ_F, then $F \circ E$ has distance $\delta_E \delta_F$.

Proof. Let $x, y \in \Sigma^n$ be distinct. Then $E(x)$ and $E(y)$ differ on at least $\delta_E m$ positions. If i is such a position (i.e. $E(x)_i \neq E(y)_i$), then $F(E(x)_i)$ and $F(E(y)_i)$ differ on at least $\delta_F t$ positions. As a whole, $(F \circ E)(x)$ and $(F \circ E)(y)$ differ on at least $\delta_E m \delta_F t$ positions. □

Lemma – The concatenation $\mathsf{WH} \circ \mathsf{RM} : \{0,1\}^{(1+d/k)^k \log |\Sigma|} \to \{0,1\}^{|\Sigma|^{k+1}}$ of the Reed-Muller code and the Walsh-Hadamard code is a function computable in time polynomial in $|\Sigma|^k$.

Proof. Using Lagrange interpolation, the coefficients of a polynomial p can be recovered from the list of its values on S^k in time polynomial in $|\Sigma|^k$. Then it is easy to evaluate p on every point from Σ^k in time polynomial in $|\Sigma|^k$. □

Suppose, in order to build an average-case hard function from a worst-case hard one, that we plan to apply the idea mentioned at the beginning of this section. Since no binary code has distance $< 1/2$, it seems that we cannot recover more than $1/4$ of the errors, which is not enough. This issue is solved by "list decoding". The following lemma shows that there is only a small number of candidates. Note that this lemma is given as an illustration, but it is not useful in the sequel.

Lemma (Johnson bound, 1962) – If $E : \{0,1\}^n \to \{0,1\}^m$ is an error-correcting code with distance $1/2 - \epsilon$, then for every $z \in \{0,1\}^m$, there are at most $k = 1/(2\epsilon)$ vectors $z^1, \ldots, z^k \in E(\{0,1\}^n) \subseteq \{0,1\}^m$ such that $\Delta(z, z^i) \leq 1/2 - \sqrt{\epsilon}$ (for every i).

Proof. Define k vectors $a^i \in \mathbb{R}^m$ by $a^i_j = 1$ if $z^i_j = z_j$, and $a^i_j = -1$ otherwise. Let $w = \sum_i a^i$.

Then for all i, $\sum_j a^i_j \geq 2m\sqrt{\epsilon}$ because $\Delta(z, z^i) \leq 1/2 - \sqrt{\epsilon}$ (they coincide on at least $(1/2 + \sqrt{\epsilon})m$ positions), therefore $\sum_i w_i = \sum_i \sum_j a^i_j \geq 2km\sqrt{\epsilon}$. Hence, by the following claim, $\langle w, w \rangle = \sum_i w_i^2 \geq (2km\sqrt{\epsilon})^2/m \geq 4\epsilon k^2 m$.

Claim - If $\sum_{i=1}^m x_i \geq \alpha \geq 0$, then $\sum_{i=1}^m x_i^2 \geq \alpha^2/m$.

The proof is by induction on m. Obvious for $m = 1$. For $m > 1$: $\sum_{i=1}^{m-1} x_i \geq \alpha - x_m$ and $\sum_{i=1}^m x_i^2 = x_m^2 + \sum_{i=1}^{m-1} x_i^2 \geq x_m^2 + (\alpha - x_m)^2/(m-1)$ by induction. But $x_m^2 + (\alpha - x_m)^2/(m-1) = (mx_m^2 + \alpha^2 - 2\alpha x_m)/(m-1) \geq \alpha^2/m$ because $m^2 x_m^2 + \alpha^2 - 2m\alpha x_m = (\alpha - mx_m)^2 \geq 0$. This proves the claim.

Furthermore, since E is a code with distance $1/2 - \epsilon$, for $i' \neq i$ we have $\Delta(z^{i'}, z^i) \geq 1/2 - \epsilon$ (they coincide on at most $(1/2 + \epsilon)m$ positions), i.e. $\sum_{j=1}^m a^i_j a^{i'}_j \leq 2\epsilon m$ ($z^i_j = z^{i'}_j$ iff $a^i_j a^{i'}_j = 1$). Hence $\langle w, w \rangle = \sum_{i=1}^k \langle a^i, a^i \rangle + \sum_{i \neq j} \langle a^i, a^j \rangle \leq km + 2\epsilon k^2 m$.

Therefore $4\epsilon k^2 m \leq km + 2\epsilon k^2 m$, that is, $k \leq 1/(2\epsilon)$. □

As a preparation to the local list decoding of the Reed-Muller code, we need some lemmas. The first lemma shows that the list in the Johnson bound can be efficiently recovered in the case of the Reed-Muller code where $k = 1$:

Lemma (Sudan, 1996) – There is a polynomial-time algorithm which, given an integer $d < m/4$ and m pairs $(a_1, b_1), \ldots, (a_m, b_m)$ of elements of Σ, returns the list of all degree d polynomials $g : \Sigma \to \Sigma$ such that the number of indices i for which $g(a_i) = b_i$ is $> 2\sqrt{dm}$. This list contains at most $\sqrt{m/d}$ polynomials.

Proof. Suppose $Q(x, y)$ is a nonzero bivariate polynomial over Σ such that $Q(a_i, b_i) = 0$ for every i, and $\deg_x(Q) \leq \sqrt{md}$ and $\deg_y(Q) \leq \sqrt{m/d}$. If g is in the list, then:

- $Q(x, g(x)) = 0$ because the univariate polynomial $Q(x, g(x))$ is of degree $\leq \sqrt{md} + d\sqrt{m/d} = 2\sqrt{md}$ and vanishes on $> 2\sqrt{md}$ points.
- $g(x) - y$ is a factor of $Q(x, y)$: consider $Q'(y) = Q(x, y) \in (\Sigma[x])[y]$, then $g(x)$ is a root of $Q'(y)$, therefore $y - g(x)$ divides $Q'(y)$, that is, there exists $R' \in (\Sigma[x])[y]$ such that $Q'(y) = (y - g(x))R'(y)$: in other words, there exists $R \in \Sigma[x, y]$ such that $Q(x, y) = (y - g(x))R(x, y)$. The number of such factors is $\leq \deg_y(Q) \leq \sqrt{m/d}$.

Therefore it is enough to find such a polynomial $Q(x, y)$ and to factorize it using, e.g., Berlekamp's algorithm. In order to find Q: write down the m linear equations that the coefficients must fulfill. There are $(1 + \sqrt{md})(1 + \sqrt{m/d}) > m$ coefficients, therefore the system has a nonzero solution which can be found by Gaussian elimination. □

The next two lemmas show that from a corrupted version of a polynomial, one can efficiently recover the initial polynomial. The first concerns univariate polynomials, whereas the second is about multivariate polynomials. Note that the polylog-time bound (in the size of the input, which is exponential in n) is required since we will apply this algorithm to an input of exponential size.

Lemma (Berlekamp and Welch, 1986) – There is a polynomial-time algorithm that, given an integer $d < |\Sigma|$ and a list $(a_1, b_1), \ldots, (a_m, b_m)$ of $m > d$ pairs of elements from Σ such that there exists a degree d polynomial $p : \Sigma \to \Sigma$ satisfying $p(a_i) = b_i$ for at least $t > m/2 + d/2$ numbers $1 \leq i \leq m$, returns the list of coefficients of p. (Remark that p is unique because it is of degree $\leq d$ and specified on $t > d$ points.)

Proof. The algorithm proceeds as follows.

- Find a degree $m - t + d$ polynomial $A(x)$ and a degree $m - t$ nonzero polynomial $B(x)$ such that $A(a_i) = b_i B(a_i)$ for all $1 \leq i \leq m$. Such polynomials are guaranteed to exist since for B one can take a degree $\leq m - t$ nonzero polynomial vanishing on the $\leq m - t$ points a_i for which $p(a_i) \neq b_i$, and then $A = pB$ is of degree $\leq m - t + d$.

  To find A and B, it is enough to solve a linear system with $2m - 2t + d + 2$ unknowns (the coefficients of A and B) and m equations (one for each i): this can be done in polynomial time by Gaussian elimination.
- Call Q the quotient of the Euclidean division of A by B, and output Q. Since $A(x) - p(x)B(x)$ is of degree $\leq m - t + d < m/2 + d/2 < t$ and is zero on $\geq t$ points, we have $A(x) = p(x)B(x)$, hence $Q(x) = p(x)$.

□

Definition – If y is a string, an algorithm A is said to have random access to y (denoted by A^y) if it has an oracle which, on input i, answers the bit y_i.

Lemma (Reed-Muller local decoder) – Let $d < (|\Sigma| - 1)/3$. There exists a probabilistic algorithm D running in time $\text{poly}(|\Sigma|, k, d)$ such that, if $f : \Sigma^k \to \Sigma$ agrees with a degree d polynomial P on a $17/18$ fraction of the inputs, then $\Pr(D^f(x) = P(x)) \geq 5/6$.

Proof. The algorithm D works as follows on input x :

- Choose $z \in \Sigma^k$ randomly and consider the line $L_z = \{x+tz \ : \ t \in \Sigma\}$. If $X \subset \Sigma^k$ is the set of inputs on which f and P differ, then let $N = |L_z \setminus \{x\} \cap X|$: we have $E(N) = \sum_{t \in \Sigma \setminus \{0\}} \Pr_z(x+tz \in X) \leq (|\Sigma|-1)/18$ therefore, by Markov inequality, $\Pr_z(N > (|\Sigma|-1)/3) < (|\Sigma|-1)/18/((|\Sigma|-1)/3) = 1/6$. Hence, with probability $> 5/6$, the number of points in $L_z \setminus \{x\}$ on which f and P coincide is at least $2(|\Sigma|-1)/3 > (|\Sigma|-1)/2 + d/2$.
- Query f on all the $|\Sigma|-1$ points of $L_z \setminus \{x\}$ to obtain the set of points $\{(t, f(x+tz)) \ : \ t \in \Sigma \setminus \{0\}\}$.
- If $Q(t) = P(x+tz)$, then Q is a univariate polynomial of degree $\leq d$. Run the Berlekamp-Welch procedure on the points $\{(t, f(x+tz)) \ : \ t \in \Sigma \setminus \{0\}\}$ to recover a polynomial Q' (supposed to equal Q).
- Output $Q'(0)$.

With probability $> 5/6$, $L_z \setminus \{x\}$ contains more than $(|\Sigma|-1)/2 + d/2$ points on which f and P coincide, hence the Berlekamp-Welch procedure outputs Q, therefore the algorithm outputs $Q(0) = P(x)$. □

The following two lemmas concern random curves of degree 3. In the sequel, when we speak of polynomials $q : \Sigma \to \Sigma^k$ (curves parametrized by $t \in \Sigma$), we use the product $tu = (tu_1, \ldots, tu_k)$ if $t \in \Sigma$ and $u \in \Sigma^k$, $u = (u_1, \ldots, u_k)$.

Lemma – Let $S \subseteq \Sigma^k$ be a fixed subset and let $x \in \Sigma^k$ be fixed. Let $q(t) = x + tu + t^2v + t^3w$ be a random degree 3 polynomial where $t \in \Sigma$ and $u, v, w \in \Sigma^k$, defining a curve $Q = q(\Sigma)$. Then for all $t \in \Sigma \setminus \{0\}$, $\Pr(q(t) \in S) = |S|/|\Sigma|^k$. Furthemore, for all $t \neq t' \in \Sigma$, the events $q(t) \in S$ and $q(t') \in S$ are independent.

Proof. First suppose that $t \neq 0$ and $t' \neq 0$. Let us first show that, if $y, y' \in \Sigma^k$ are fixed, then the events $q(t) = y$ and $q(t') = y'$ are independent. We have $q(t) = y \iff u = (y - x - t^2v - t^3w)/t$, therefore the choice of v and w is free but then u is fixed, hence $\Pr(q(t) = y) = 1/|\Sigma|^k$ and similarly for $q(t') = y'$. Furthermore, $(q(t) = y \wedge q(t') = y') \iff (tu + t^2v + t^3w = y - x \wedge t'u + t'^2v + t'^3w = y' - x) \iff (u = (x - t^2v + t^3w)/t \wedge (t't^2 - tt'^2)v + (t't^3 - tt'^3)w = t'(y-x) - t(y'-x))$: w can be chosen freely but then u and v are fixed, therefore this happens with probability $1/|\Sigma|^{2k} = \Pr(q(t) = x)\Pr(q(t') = y)$.

Now, $\Pr(q(t) \in S) = \sum_{y \in S} \Pr(q(t) = y)$ because these events are disjoint, $= |S|/|\Sigma|^k$, similarly for $\Pr(q(t') \in S)$. On the other hand, $\Pr(q(t) \in S \wedge q(t') \in S) = \sum_{y,y' \in S} \Pr(q(t) = y \wedge q(t') = y')$ (disjoint events) $= |S|^2/|\Sigma|^{2k}$.

Now, if one of t or t' is zero, say $t = 0$, then either $x \in S$ and $\Pr(q(t) \in S) = 1$, or $x \notin S$ and $\Pr(q(t) \in S) = 0$: in both cases, the events $q(t) \in S$ and $q(t') \in S$ are independent. □

Lemma – Let $S \subseteq \Sigma^k$ be a fixed set such that $\Pr_y(y \in S) \geq \epsilon$, where $\epsilon < 1/2$, and let $x \in \Sigma^k$ be fixed. Let $q(t) = x + tu + t^2v + t^3w$ be a random degree 3 polynomial where $t \in \Sigma$ and $u, v, w \in \Sigma^k$, defining a curve $Q = q(\Sigma)$. Then $\Pr(|Q \cap S| \geq 2\epsilon|\Sigma|/15) \geq 1 - 100/(\epsilon|\Sigma|)$.

Proof. Since q is a degree 3 polynomial, every $y \in \Sigma^k$ cannot have more than 3 preimages. Hence $|Q \cap S| \geq (\sum_{t \in \Sigma \setminus \{0\}} x_t)/3$ where $x_t = 1$ if $q(t) \in S$ and $x_t = 0$ otherwise. By the preceding lemma, this is a sum of pairwise independent random variables, where $\Pr(x_t = 1) \geq \epsilon$. We have: $E(|Q \cap S|) \geq \epsilon(|\Sigma|-1)/3$, $\mathrm{Var}(|Q \cap S|) \geq (|\Sigma|-1)\epsilon(1-\epsilon)/9$ and by Chebyshev's inequality: $\Pr(|Q \cap S| \leq 2\epsilon|\Sigma|/15) \leq \Pr(||Q \cap S| - E(|Q \cap S|)| \geq \epsilon(|\Sigma|-1)/3 - 2\epsilon|\Sigma|/15)$. But since $|\Sigma| \geq 2$, $(|\Sigma|-1)/3 - 2|\Sigma|/15 \geq |\Sigma|/30$, hence: $\Pr \leq \Pr(||Q \cap S| - E(|Q \cap S|)| \geq \epsilon|\Sigma|/30) \leq (|\Sigma|-1)\epsilon(1-\epsilon)/(9(\epsilon|\Sigma|/30)^2) \leq 100/(\epsilon|\Sigma|)$. □

For our purpose, we need extremely efficient decoding algorithms.

Definition (local list decoder) – Let $E : \Sigma^n \to \Gamma^m$ be a code and let $0 < \eta < 1$. A probabilistic algorithm D is called a *local list decoder for E handling η errors in time t with advice of size s* if, for every $x \in \Sigma^n$ and $z \in \Gamma^m$ satisfying $\Delta(E(x), z) \leq \eta$, there exists $a \in \{0,1\}^\star$, $|a| \leq s$ (a binary advice of size s) such that for every $i \leq n$, D on input (a, i) and with random access to z, runs in time t and outputs x_i with probability $\geq 2/3$.

Remark:

- In the sequel, we will consider local list decoders working in time polylogarithmic in m with advice of size logarithmic in n.
- If $\mathcal{F}$ is the set of functions $f : \Sigma^k \to \Sigma$, then the Reed-Muller code can be considered as taking its values into $\mathcal{F}$.
- In the following proposition, we want to local-list decode Reed-Muller code: the integer a (advice) must depend only on f and must therefore be the same for all $x \in \Sigma^k$.

Proposition (local list decoding of the Reed-Muller code) – If $|\Sigma| \geq d^2$ then the Reed-Muller code has a local list decoder handling $1 - 30\sqrt{d/|\Sigma|}$ errors in time $\text{poly}(|\Sigma|, k)$ with advice of size $(k+1)\log|\Sigma|$.

More precisely, there is a probabilistic algorithm D running in time $\text{poly}(|\Sigma|, k)$ such that, if $f : \Sigma^k \to \Sigma$ agrees with a degree d polynomial p on a fraction $30\sqrt{d/|\Sigma|}$ of the inputs, then there is $(a, b) \in \Sigma^k \times \Sigma$ such that for all $x \in \Sigma^k$, $\Pr(D^f(x, a, b) = p(x)) \geq 2/3$.

Proof. Let $\epsilon = (2\sqrt{d/|\Sigma|})^{1/2}$. Let us first propose a probabilistic algorithm, working on $\geq 1 - \epsilon$ of the $x \in \Sigma^k$, which has a probability of success $\geq 1 - \epsilon$.

1. Choose at random $r \in \Sigma$ ($r \neq 0$) and $v, w \in \Sigma^k$, and define $u = (a - x - vr^2 - wr^3)/r$ and $q(t) = x + tu + t^2v + t^3w$ (for $t \in \Sigma$), so that the degree 3 curve $Q = q(\Sigma)$ satisfies $q(0) = x$ and $q(r) = a$.
2. Query f on all the points of Q to obtain the pairs $(t, f(q(t)))$ for all $t \in \Sigma$.
3. Run Sudan's algorithm to obtain a list $g_1, \ldots, g_p$ of all degree $3d$ polynomials such that $g_i(t) = f(q(t))$ for at least $4\sqrt{d|\Sigma|}$ points $t \in \Sigma$. (Note that Sudan's algorithm returns polynomials which coincides with at least $2\sqrt{3d|\Sigma|} < 4\sqrt{d|\Sigma|}$ points, but we can ignore those who agree on $< 4\sqrt{d|\Sigma|}$ points)
4. If there is a unique i such that $g_i(r) = b$ then output $g_i(0)$. Otherwise halt (no output).

In order to prove that a correct pair (a, b) exists, we will use a probabilistic argument. Fix $x \in \Sigma^k$, take $a \in \Sigma^k$ at random and $b = p(a)$. The random choices are then a, r, v and w. Then the first step of the algorithm amounts to choose a random curve Q such that $q(0) = x$, and the last step amounts to choose a random $r \in \Sigma$ and check if $g_i(r) = p(q(r))$. We claim that the probability that $g_i = p \circ q$ is high. Indeed, thanks to the lemma, if $S = \{y : f(y) = p(y)\}$, then by assumption $\Pr_y(y \in S) \geq 30\sqrt{d/|\Sigma|}$, therefore by the lemma, $\Pr_{a,r,v,w}(|Q \cap S| \geq 4\sqrt{d/|\Sigma|}) \geq 1 - 4/\sqrt{d|\Sigma|}$. Therefore, $p \circ q$ lies in the list $g_1, \ldots, g_p$ with probability $\geq 1 - 4/\sqrt{d|\Sigma|}$. Furthermore, the probability that no other polynomial g_i satisfies $g_i(r) = p(q(r))$ is high: g_i and $p \circ q$, as distinct polynomials of degree $\leq 3d$, only agree on $\leq 3d$ points, therefore $\Pr_{a,r,v,w}(g_i(r) = p(q(r))) \leq 3d/|\Sigma|$. As a whole, since there are $\leq \sqrt{|\Sigma|/(3d)}$ polynomials g_i, we have $\Pr_{a,r,v,w}(i \text{ unique st } g_i(r) = p(q(r))) \geq 1 - \sqrt{|\Sigma|/(3d)}3d/|\Sigma| = 1 - \sqrt{3d/|\Sigma|}$. Thus with probability $\geq 1 - 4/\sqrt{d|\Sigma|} - \sqrt{3d/|\Sigma|} \geq 1 - 2\sqrt{d/|\Sigma|} = 1 - \epsilon^2$ (for sufficiently large d) we identify uniquely $p \circ q$ and output $p \circ q(0) = p(x)$. Hence $\Pr_{a,r,v,w}(D^f(x, a, b) = p(x)) \geq 1 - \epsilon^2$.

Let A be the random event (depending on a) $\Pr_{r,v,w}(D^f(x, a, b) = p(x)) \geq 1 - \epsilon$. Then $\Pr_{a,r,v,w}(D^f(x, a, b) = p(x)) = \Pr_{a,r,v,w}(D^f(x, a, b) = p(x) | a \in A)\Pr(A) + \Pr_{a,r,v,w}(D^f(x, a, b) = p(x) | a \notin A)(1 - \Pr(A)) \leq \Pr(A) + (1 - \epsilon)(1 - \Pr(A)) = 1 - \epsilon + \epsilon\Pr(A)$. Hence $1 - \epsilon + \epsilon\Pr(A) \geq 1 - \epsilon^2$, that is, $\Pr(A) \geq 1 - \epsilon$. Hence, a random a is valid for x with probability $1 - \epsilon$.

Since the argument is valid for an arbitrary $x \in \Sigma^k$, we have: $\Pr_x\Pr_a(a \text{ is valid for } x) = \Pr_a\Pr_x(a \text{ is valid for } x) \geq 1 - \epsilon$. Therefore there exists a specific $a \in \Sigma^k$ which is valid for a fraction $\geq 1 - \epsilon$ of the inputs $x \in \Sigma^k$.

It remains to get the result on all inputs $x \in \Sigma^k$: it is enough to apply the Reed-Muller local decoder in order to recover the polynomial p which was computed correctly on $\geq 1 - \epsilon > 17/18$ of the inputs. As a whole, the probability of success is $\geq 1 - (\epsilon + 1/6) \geq 2/3$. □

The next proposition is given without proof.

Proposition (local list decoding of the Walsh-Hadamard code) – For every $\epsilon > 0$, the Walsh-Hadamard code $\mathsf{WH} : \{0,1\}^n \to \{0,1\}^{2^n}$ has a local list decoder handling $(1/2 - \epsilon)$ errors in time $\text{poly}(n, 1/\epsilon)$ with advice of size $O(\log(n/\epsilon))$.

Finally, the concatenation of two locally list-decodable codes is again locally list-decodable.

Lemma – Suppose $E : \{0,1\}^n \to \Sigma^m$ and $F : \Sigma \to \{0,1\}^q$ are locally list-decodable handling $(1-\epsilon_E)$ and $(1/2-\epsilon_F)$ errors in time t_E and t_F with advices of size s_E and s_F respectively, where $\epsilon_F < \epsilon_E 2^{s_F-1}$. Then $F \circ E : \{0,1\}^n \to \{0,1\}^{mt}$ is locally list-decodable handling $(1/2-\epsilon)$ errors in time $t_E t_F$ with advice of size $s_E + s_F$, where $\epsilon = \epsilon_E 2^{s_F}$.

Proof. To decode $F \circ E$ on input $z \in \{0,1\}^{mt}$, we use a pair of advices (a_E, a_F), where a_E is an advice for E and a_F for F. The word z is made of m blocks of q bits, each block being the corrupted image by F of a letter from Σ. We run the decoder for E with advice a_E, building its input "on the fly": each time it asks for a value $y_i \in \Sigma$, we decode the i-th block of z using F with advice a_F.

We claim that there is a choice of (a_E, a_F) such that this decoder can handle $(1/2-\epsilon)$ errors. Indeed, suppose $\Delta(z, F \circ E(x)) < 1/2-\epsilon$: they agree on at least $(1/2+\epsilon)qm$ bits. Therefore, the number b of blocks of q bits on which they agree on at least $(1/2+\epsilon_F)q$ bits satisfies $bq + (m-b)(1/2+\epsilon_F)q > (1/2+\epsilon)qm$, that is, $b > (\epsilon-\epsilon_F)/(1/2-\epsilon_F)m > 2(\epsilon-\epsilon_F)m > \epsilon m$. Let $N = 2^{s_F}$ be the total number of possible advices a_F: then $b > \epsilon_E N m$, therefore there is an advice a_F such that the word $y' \in \Sigma^m$ decoded from z by F with advice a_F agrees with $E(x) \in \Sigma^m$ on more than $\epsilon_E m$ positions. We can then use the advice for E for x and this y' to get $x \in \{0,1\}^n$. □

Corollary – There is a constant c such that $\mathsf{WH} \circ \mathsf{RM}$ has a local list decoder handling $1/2-\epsilon$ errors in time $(k|\Sigma|)^c$ with advice of size $ck \log|\Sigma|$, where $\epsilon = \sqrt{d/|\Sigma|}(\log|\Sigma|)^c$.

Proof. The local list decoder for WH handling $\epsilon/2$ errors in the previous proposition has running time $\mathrm{poly}(|\Sigma|)$ and advice size $O(\log|\Sigma|)$. The local list decoder for RM as in the proposition above has running time $\mathrm{poly}(|\Sigma|, k)$ and advice size $(k+1)\log|\Sigma|$. Therefore the composition of the two has running time $\mathrm{poly}(|\Sigma|, k)$ and advice size $O(k \log|\Sigma|)$. □

5.2 Application

Theorem (Worst-case to average-case hardness) – There is a constant a such that, if $f \in \mathsf{DTIME}(2^{O(n)})$ is such that $\mathsf{H}_{\mathsf{wrs}}(f) \geq s(n)$ for some time-constructible nondecreasing function $s : \mathbb{N} \to \mathbb{N}$ satisfying $s(n) \geq n^a$, then there exists a function $g \in \mathsf{DTIME}(2^{O(n)})$ such that $\mathsf{H}_{\mathsf{avg}}(g) \geq s(n)^{1/5}$ for n large enough.

Proof. Let $a = 16c$, where c is the constant from the previous corollary (local list decoding of $\mathsf{WH} \circ \mathsf{RM}$). The restriction of f to $\{0,1\}^n$ is considered as a string over $\{0,1\}$ of size 2^n. Let $\Sigma = GF(2^m)$ such that $|\Sigma| = s(n)^{4/a}$, let $d = \sqrt{|\Sigma|}$ and k such that $(1+d/k)^k \log|\Sigma| = 2^n$. In particular, remark that $d = s(n)^{2/a} \geq n^2$, $k \leq n$ (therefore $d/k \geq |\Sigma|^{1/4}$) and even $(k/4)\log|\Sigma| \leq k\log(1+d/k) + \log\log|\Sigma| = n$, that is, $k \log|\Sigma| \leq 4n$.

Let $g = \mathsf{WH} \circ \mathsf{RM}(f) \in \{0,1\}^{|\Sigma|^{(k+1)}}$, that is, $g : \{0,1\}^{(k+1)\log|\Sigma|} \to \{0,1\}$. The function g is computable in time $2^{O(n)}$ because $\mathsf{WH} \circ \mathsf{RM}$ is computable in time polynomial in $|\Sigma|^k$: it is enough to compute $f(x)$ for all $x \in \{0,1\}^n$ in order to obtain the characteristic string χ_f of f, and then to compute $\mathsf{WH} \circ \mathsf{RM}(\chi_f)$ in time polynomial in $|\Sigma|^k \leq 2^{4n}$.

Suppose $h = \mathsf{H}_{\mathsf{avg}}(g) < |\Sigma|^{1/5}$: g can be computed by a circuit C of size $h+1$ on a fraction $\geq 1/2+1/h$ of the inputs. Let $x \in \{0,1\}^n$: we want to compute $f(x)$. It is enough to run the local list-decoder from the previous corollary for $\mathsf{WH} \circ \mathsf{RM}$: indeed, $\sqrt{d/|\Sigma|}(\log|\Sigma|)^c < 1/h$ since $\sqrt{d/|\Sigma|} = |\Sigma|^{-1/4}$. Then the advice needed is of size $ck\log|\Sigma|$; the time needed to decode is $(k|\Sigma|)^c$. Such a decoder can be turned into a circuit D for f of size $(k|\Sigma|)^{2c}$, with calls to g: as a whole, if the calls to g are replaced by the circuit C, we obtain a circuit for f of size $(k|\Sigma|)^{2c}(h+1)$. This is a contradiction since $(k|\Sigma|)^{2c}(h+1) < s(n)$, because $n^{2c}s(n)^{8c/a+1/5} = n^{a/8}s(n)^{7/10} < s(n)$. □

Corollary –

- If there is $f \in \mathsf{DTIME}(2^{O(n)})$ such that $\mathsf{H}_{\mathsf{wrs}}(f) \geq 2^{\epsilon n}$, then there is $g \in \mathsf{DTIME}(2^{O(n)})$ such that $\mathsf{H}_{\mathsf{avg}}(g) \geq 2^{\epsilon n/5}$.
- If there is $f \in \mathsf{DTIME}(2^{O(n)})$ with $\mathsf{H}_{\mathsf{wrs}}(f) \geq n^c$ (for c large enough), then there is $g \in \mathsf{DTIME}(2^{O(n)})$ with $\mathsf{H}_{\mathsf{avg}}(g) \geq n^{c/5}$.