Error-correcting codes

Satyadev Nandakumar

August 21, 2026

Contents

1 Motivation **1**

2 Error-correcting code **2**

3 The Gilbert-Varshamov bound **2**

4 Explicit Codes **3**

5 Walsh Hadamard codes **3**

6 Reed-Solomon code (univariate polynomial) **4**

7 Reed-Muller code (multivariate polynomial) **4**

1 Motivation

Our overall goal is to *amplify* the hardness of worst-case hard problems so that they become *hard on average for circuits.*

Take the contrapositive of the problem: suppose we know how to solve a problem on average, then can we solve every instance of the problem - *i.e.* solve it in the worst-case?

This brings us to a concept that is well-studied in the context of telecommunications - namely, *error-correcting codes.* The setting is as follows : Alice wants to send an n-length binary string to Bob across a noisy channel. The channel may scramble some δ fraction of the bits. Bob does not know which bits have been scrambled. He only knows that δn bits may be wrong, and at least $(1-\delta)n$ bits are correct.

Can we recover all the bits? This is the question of error-free transition. One way to implement this is to send a slightly longer message so that even with the error, it is possible to recover the actual message. This is the domain of *error-correcting codes.*

2 Error-correcting code

Definition 2.1. For strings $x, y \in \{0,1\}^n$, the *fractional Hamming distance* between x and y, denoted $\Delta(x,y)$ is defined by

$$\Delta(x,y) = \frac{1}{n} \|\{i \mid x_i \neq y_i\}\| . \tag{1}$$

Definition 2.2. For every $\delta \in [0,1]$, a function $E : \{0,1\}^n \to \{0,1\}^m$ is an *error-correcting code with distance* δ if for every pair of distinct n-length strings x and y, we have $\Delta(E(x), E(y)) \geq \delta$.

That is, if $\Delta(x,y) \geq \frac{1}{n}$, then $\Delta(E(x), E(y)) \geq \delta$.

3 The Gilbert-Varshamov bound

The *Shannon entropy* associated with the probability distribution $(p, 1-p)$ is defined to be $H(p) = p \log_2 \frac{1}{p} + (1-p) \log_2 \frac{1}{(1-p)}$.

Lemma 3.1. *For every $\delta < \frac{1}{2}$ and sufficiently large n, there is an error-correcting code $E : \{0,1\}^n \to \{0,1\}^{n/(1-H(\delta))}$ w with distance δ.*

For example, when δ is either 0 or 1, we have $H(\delta) = 0$, so that the length of the error-correcting code is the same as the length of the input - this makes intuitive sense, since the input is either a string of ones, or a string of zeroes, and in either case, Bob can correct any error trivially. On the other extreme, when both 0 and 1 are equally likely, *i.e.* $\delta = 0.5$, we have $H(\delta) = 1$, and in this case, the above bound is useless. (There may still be an error-correcting code for $\delta = 0.5$, it is just that this particular bound does not guarantee a code.)

Proof. We show a slightly weaker bound, $m = 2n/(1 - H(\delta))$. The strategy is to choose a random code.

We show that the probability that a random code does not work, is less than 1. That is, the probability that there are distinct inputs x and y for which $\Delta(E(x), E(y)) < \delta$ is less than 1.

For every output string z, the number of strings that are at distance $\leq \delta$ from it is $\binom{m}{\lceil \delta m \rceil}$. Using standard Sterling's approximation-based bounds[1], we get that this number is less than $0.992^{H(\delta)m}$ for all sufficiently large m.

Hence, the probability of the set of strings which are at fractional Hamming distance $\leq \delta$ from y is less than $0.99\frac{2^{H(\delta)m}}{2^m}$.

The total number of distinct pairs is less than 2^{2n} (note: not 2^{2m}, since we are counting only distinct pairs of valid output strings.)

Hence, the probability that there is some distinct pair of output strings with fractional Hamming distance $\leq \delta$ is at most

$$0.992^{2n} \frac{2^{H(\delta)m}}{2^m} < 1. \tag{2}$$

[1] see Herbert Robbins, "A Remark on Sterling's formula", 1955, for the precise version that is used to derive this bound.

□

4 Explicit Codes

For computational applications, in addition to the existence of an error-correcting code, we must be able to easily compute an error-correcting code. These codes are called *explicit codes*.

Definition 4.1. An *explicit error-correcting code* is a function $E : \{0,1\}^n \to \{0,1\}^m$ such that the following hold.

1. E is an error-correcting code
2. There is a poly(m) function such that given x, it computes $E(x)$.
3. There is a poly(m) algorithm such that for any $x \in \{0,1\}^n$, given any $y \in \{0,1\}^m$ such that $\Delta(y, E(x)) \leq \rho$ for some ρ, the algorithm given y, outputs x.

Condition 2 stipulates efficient encoding, and condition 3 stipulates efficient decoding even when there are errrors in y. Note that condition 2 stipulates the running time in terms of m, the length of the output, not the length of the input. This allows us to treat even some codes with m as exponential in n, as an explicit code.

What follows are a few well-known explicit error-correcting codes.

5 Walsh Hadamard codes

The Walsh-Hadamard code is an explicit error-correcting code defined as follows.

Let $x \odot y = \sum_{i=1}^{n} x_i y_i \mod 2$ as the dot product of x and y - that is, it is the XOR of the bitwise ANDs of the corresponding individual bits. For example, $0010 \odot 1010 = 01 \oplus 00 \oplus 11 \oplus 00 = 0 \oplus 0 \oplus 1 \oplus 0 = 1$.

The Walsh-Hadamard code of an n-length string x, denoted WH(x), is the 2^n length bitstring produced by concatenating the results $x \odot 0^n$, $x \odot (0^{n-1}1)$, $\ldots$, $x \odot 1^n$ in lexicographic order.

For example, the Walsh Hadamard code of 01 is

$$(01 \odot 00)(01 \odot 01)(01 \odot 10)(01 \odot 11) = 0101.$$

This code is explicit because it can be computed in poly(2^n) time, with 2^n being the length of the output. It is clearly not in poly(n) time, since the code.

Lemma 5.1. *The Walsh-Hadamard code has distance* $1/2$.

Proof. Let x and x' be distinct n-length strings. Let $z = x \oplus x'$ be the the bitwise xor of the two strings. Since x and x' are distinct, $z \neq 0^n$. Consider some position of z, say $z[i]$ which is equal to 1 (there must be such a position).

Take two strings y, and $\hat{y}$ such that $y[i] \neq \hat{y}[i]$. Thus exactly one of these bits is 1, and the other is 0. It follows that in $\mathrm{WH}(z)$, the positions corresponding to y and $\hat{y}$ have to be different.

Thus the set of all 2^n strings can be partitioned into two subsets of equal size, the first where the corresponding bit of $\mathrm{WH}(z)$ is 0 and the other where the bits are 1. Hence, $WH(z)$ has half zeroes and half ones.

What we need to show is that $\mathrm{WH}(x)$ and $\mathrm{WH}(x')$ differ at exactly half the positions. We note that WH is a linear function (HW 1). We have

$$\begin{aligned} \mathrm{WH}(x) \oplus \mathrm{WH}(x') &= \mathrm{WH}(x \oplus x') && \text{[linearity]} \\ &= \mathrm{WH}(z), \end{aligned}$$

which has exactly half zeroes and half ones. Ones correspond exactly to the positions where $\mathrm{WH}(x)$ and $\mathrm{WH}(x')$ differ, and zeroes exactly correspond to the positions where they are the same. Thus they differ at exactly half the positions, which establishes the result. □

6 Reed-Solomon code (univariate polynomial)

The Walsh-Hadamard code had length exponential in the length of the input. Can we have shorter error-correcting codes? We will use the idea of evaluating polynomials, and sending their evaluations at many points. Even when some values are scrambled, we can recover the polynomial uniquely. First, however, we will generalize our setting from binary to *finite fields* in general.

Let Σ be an arbitrary finite alphabet. The fractional Hamming distance between two n-length strings x and y is $\Delta(x, y) = \frac{1}{n}|\{i \mid x_i \neq y_i\}|$. Let $0 < \delta < 1$. We say that a function $E : \Sigma^n \to \Sigma^m$ is an error-correcting code if for every $x \neq y \in \Sigma^n$, we have $\Delta(E(x), E(y)) \geq \delta$.

Let $\mathbb{F}$ be a finite field, and let n, m be numbers such that $n \leq m \leq |\mathbb{F}|$.

Definition 6.1. The *Reed-Solomon code* from $\mathbb{F}^n \to \mathbb{F}^m$ is the function RS such that $\mathrm{RS}(a_0, \ldots, a_n) = z_0 \ldots z_{m-1}$ where $z_j = \sum_{i=0}^{n-1} a_i f_j^i$, and f_1, f_2, …, f_{m-1} are the elements of $\mathbb{F}^m$ under some ordering.

Lemma 6.2. *The Reed Solomon code defined above has distance* $1 - \frac{n}{m}$.

Proof. The Reed-Solomon code is a linear function: for every x and y, we have $\mathrm{RS}(x + y) = \mathrm{RS}(x) + \mathrm{RS}(y)$, where $+$ denotes the componentwise addition in $\mathbb{F}$ (HW1). The result follows if for every $z \neq 0^n$, we show that $\mathrm{RS}(z)$ has at most n co-ordinates which are zero. This follows from the fact that a univariate, non-zero n-degree polynomial has at most n roots. Hence, for distinct values x and y, $\mathrm{RS}(x)$ and $\mathrm{RS}(y)$ differ in at least $m - n$ places, hence the fractional Hamming distance is at least $\frac{m-n}{m}$. □

The question we are leaving unspecified for now is: how short is this code in comparison with the Walsh-Hadamard code? and how worse is its error-correction in comparison? We will postpone this important issue for now, since the answer will depend on $|\mathbb{F}|$. We will determine this when we discuss efficient decoding.

7 Reed-Muller code (multivariate polynomial)

This variant encodes the evaluations of a multivariate polynomial.

Definition 7.1. Let $\mathbb{F}$ be a finit field, and let ℓ, d be numbers with $d < |\mathbb{F}|$.