In the last lecture, we studied the Reed-Muller code, $RM_2(t, v)$ and saw the “majority logic decoder” for such codes. In today’s lecture, we will start off with a formal statement of the algorithm and then prove its correctness.

1 Majority Logic Decoding

Below is the formal statement of the majority logic decoding algorithm.

**INPUT:** $y = \langle y_a \rangle_{a \in \mathbb{F}_2^v}$ such that there exists $P(x_1, \ldots, x_v)$ of degree at most $t$ with $\Delta(y, \langle P(a) \rangle_{a \in \mathbb{F}_2^v}) < 2^{r-t-1}$.

**OUTPUT:** Compute $P(x_1, \ldots, x_r)$

1. $P \equiv 0$, $r \leftarrow t$

2. (a) For all $S \subseteq [v]$, such that $|S| = t$, set $C_S$ to be the majority over $b \in \mathbb{F}_{2^{v-t}}$ of $\sum_{a \in \mathbb{F}_2^v, a_S = b} y_a$. Set $P \leftarrow P + C_S \prod_{j \in S} x_j$

   (b) For all $a \in \mathbb{F}_2^v$, $y_a \leftarrow y_a - \sum_{S \subseteq [v], |S| = t} C_S \prod_{j \in S} a_j$

3. $r \leftarrow r - 1$

4. If $r < 0$ output $P$, else go to step 2.

Note that this is an $O(n^3t)$ algorithm, where $n = 2^v$. This is true because the number of iterations in step 2 (a) is at most $\binom{v}{t} \leq n$, and computing the majority in that step takes time $O(n^2)$. Finally, step 2 is repeated at most $t$ times.

2 Correctness of the algorithm

We need one further result to prove the correctness of the majority logic decoder, namely the lemma from the last lecture.

**Lemma 2.1.** For all $t \geq 0$ and $S \subseteq [v]$ such that $|S| = t$, any $v$-variate polynomial $P$ of degree at most $t$, for every $b \in \mathbb{F}_{2^{v-t}}$, has $\sum_{a \in \mathbb{F}_2^v, a_S = b} P(a) = C_S$. 


At this point, we need a new notation. Given a subset $S$ of $[v]$, define

$$R_S(x_1, x_2, \ldots, x_v) \triangleq \prod_{j \in S} x_j.$$ 

We will need the following two observations.

**Observation 2.2.** For all $S \subseteq [v]$ and $T \subset S$, for all $b \in \mathbb{F}_2^{v-|S|}$, \[\sum_{a \in \mathbb{F}_2^v, a_S = b} R_T(a) = 0.\]

**Observation 2.3.** For all $S \subseteq [v]$ and $b \in \mathbb{F}_2^{v-|S|}$, \[\sum_{a \in \mathbb{F}_2^v, a_S = b} R_S(a) = 1.\]

Subject to the proof of these two observations (which we will do later), we are now ready to prove Lemma 2.1.

**Proof of Lemma 2.1** Let $P_b$ denote the polynomial obtained from $P$ by substituting the variables \(\{x_i | i \notin S\}\) according to $b$. $P_b$ now only has monomials of the form $R_Y(x_1, x_2, \ldots, x_v)$ for $Y \subseteq S$. In other words, $P_b(x_1, \ldots, x_v) = C_SR_{S}(x_1, \ldots, x_v) + \sum_{T \subset S} C'_T R_T(x_1, \ldots, x_v).$

The definition of $P_b$ and the above relation implies the following:

$$\sum_{a \in \mathbb{F}_2^v, a_S = b} P(a) = \sum_{a \in \mathbb{F}_2^v, a_S = b} P_b(a) = C_S \sum_{a \in \mathbb{F}_2^v, a_S = b} R_S(a) + \sum_{T \subset S} C'_T \sum_{a \in \mathbb{F}_2^v, a_S = b} R_T(a) = C_S,$$

where the last equality follows from Observations 2.3 and 2.2.

This proves Lemma 2.1. We still must prove the two observations, first, Observation 2.2.

**Proof of Observation 2.2** Consider the sum $\sum_{a \in \mathbb{F}_2^v, a_S = b} R_T(a)$. Fix some $i \in S \setminus T$. We can divide this into the sum of two parts: $\sum_{a \in \mathbb{F}_2^v, a_S = b, a_i = 0} R_T(a) + \sum_{a \in \mathbb{F}_2^v, a_S = b, a_i = 1} R_T(a)$. Since $R_T(x)$ does not depend on $x_i$, the two parts are equal, and the sum is zero since it is computed over $\mathbb{F}_2$.

We now move to the proof of Observation 2.3.

**Proof of Observation 2.3** Note that $R_S(a) = 1$ if and only if for all $i \in S$, $a_i = 1$. Notice that this is true for exactly one value in \(\{a \in \mathbb{F}_2^v | a_S = b\}\).

### 3 Construction of explicit binary asymptotically good codes

We now return to the question of explicit binary codes with both $R$ and $\delta$ greater than zero. Recall that the Reed-Muller codes give us $R = \frac{1}{2}$ and $\delta = \frac{1}{\sqrt{n}}$, which falls short of this goal. The Reed-Solomon code, as a binary code, comes closer - it gives us the same rate, and $\delta = \frac{1}{\log n}$, as we discuss next.
Consider the Reed-Solomon over $\mathbb{F}_{2^s}$ for some large enough $s$. It is possible to get a code with (e.g.) a rate of $\frac{1}{2}$, and have an $[n, \frac{n}{2}, \frac{n}{2} + 1]_2$ code. We now consider a Reed-Solomon codeword, where every symbol in $\mathbb{F}_{2^s}$ is represented by an $s$-bit vector. Now, the “obvious” binary code created by viewing symbols from $\mathbb{F}_{2^s}$ as bit vectors as above is an $[ns, \frac{ns}{2}, \frac{n}{2} + 1]_2$ code. Note that the distance of this code is only $\Theta\left(\frac{N}{\log N}\right)$, where $N = ns$ is the block length of the final binary code. Recall that $n = 2^s$ and so $N = n \log n$.

The reason for the poor distance is that the bit vectors corresponding to two different symbols in $\mathbb{F}_{2^s}$ may only differ by one bit. Thus, $d$ positions which have different $\mathbb{F}_{2^s}$ symbols might result in a distance of only $d$ as bit vectors.

To fix this problem, we can consider applying a function to the bit-vectors to increase the distance between those bit-vectors that differ in smaller numbers of bits. Note that such a function is simply a code, and Forney introduced this idea of “concatenating” in 1966.

More formally, consider a conversion function that maps $\mathbb{F}_{2^s} \to (\mathbb{F}_2)^{s'}$ in such a fashion that, even if $\Delta(x, y) = 1$, $\Delta(f(x), f(y)) \geq d'$. If we find such a function, we can construct a code with $R > 0, \delta > 0$ as long as the “inner distance”, $d'$, is $\Omega(s')$. In the next lecture, we will formally define code concatenation and consider the problem of finding good inner codes.