Class 11 Exercises

CS250/EE387, Winter 2025

Today, we'll see a list-decoding algorithm (which might look somewhat familiar...) for a class of codes called *Chinese Remainder Codes* (c.f. Problem 2.1 on HW3). Below, $\mathbb{Z}_N$ refers to the integers $\{0, 1, \ldots, N-1\}$ with arithmetic mod N.

These codes are based on the *Chinese Remainder Theorem*:

Theorem 1. *Let $p_1, \ldots, p_t$ be relatively prime. Let $P = \prod_{i=1}^{t} p_i$. Fix $a_1, \ldots, a_t \in \mathbb{Z}_P$. There is a unique $m \in \mathbb{Z}_P$ so that $m \equiv a_i \mod p_i$ for all $i \in [t]$.*

This inspires the following code[1]:

Definition 1. *Fix $p_1 < p_2 < \cdots < p_n$ relatively prime. Let $N = \prod_{i=1}^{n} p_i$ and let $K = \prod_{i=1}^{k} p_i$. Define an encoding map $E : \mathbb{Z}_K \to \mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2} \times \cdots \times \mathbb{Z}_{p_n}$ given by*

$$E(m) = (m \mod p_1, m \mod p_2, \ldots, m \mod p_n).$$

The Chinese Remainder Code *with parameters k and n defined by $p_1, \ldots, p_n$ is the set of codewords $\{E(m) : m \in \mathbb{Z}_K\}$.*

In your homework (HW3, problem 2.1), you will show that these codes have distance at least $n - k + 1$, matching RS codes. But what about list-decoding?

1. Consider the following list-decoding algorithm. Let $y = (y_1, \ldots, y_n) \in \mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_n}$ be a received word. Our goal is to find all of the $m \in \mathbb{Z}_K$ so that $\text{dist}(E(m), y) \leq \rho n$.

   Input: $y \in \mathbb{Z}_{p_1} \times \cdots \times \mathbb{Z}_{p_n}$, parameters ℓ, F to be determined.

   - Let $r \in \mathbb{Z}_N$ be the unique element so that $r \equiv y_i \mod p_i$ for all $i \in [n]$.
   - **Interpolation Step:** Find $a = (a_0, a_1, \ldots, a_{\ell-1})$ so that $a \neq \vec{0}$ and so that the following hold:
     - $|a_i| \leq F/K^i$ for all $i = 0, \ldots, \ell - 1$.
     - $\sum_{i=0}^{\ell-1} a_i r^i \equiv 0 \mod N$.
   - **Root-finding Step:** Return the roots of $Q(Y) = \sum_{i=0}^{\ell-1} a_i Y^i$. (Here, this polynomial is over the integers, not modulo anything).

   There is no question for this part, just make sure the algorithm parses.

[1] Notice that the alphabet is different for each symbol, so it doesn't strictly match our definition of a code, but let's go with it.

2. Suppose that we can do the **Interpolation Step** with our chosen ℓ, F. Let $m \in \mathbb{Z}_K$ and suppose that $\text{dist}(E(m), y) \leq \rho n$. Show that, if ρ is not too large, then $Q(m) = 0$, where $Q(Y) = \sum_i a_i Y^i$.

   How big can ρ be, in terms of ℓ, F, and the p_i's? (It will be useful later to simplify your answer to be in terms of ℓ, F and p_1, the smallest of the p_i's).

   Hint: Follow the following outline:

   (a) Suppose that $E(m)$ and y agree in position i. Explain why $Q(m) \equiv 0 \mod p_i$.

   (b) By (i), if $\text{dist}(E(m), y) \leq \rho n$, then there are $(1-\rho)n$ values of i so that $Q(m) \equiv 0 \mod p_i$. Use the conditions on the a_i to bound $|Q(m)| \leq [something]$ and use the Chinese Remainder Theorem to conclude that $Q(m) \equiv 0$, provided that ρ is not too big.

3. Observe that the previous part shows that, *if* we can do the Interpolation Step, and *if* ρ is not too big, any m that satisfies $\text{dist}(E(m), y) \leq \rho n$ will be returned in the root-finding step. That is, we will have a correct list-decoding algorithm, up to radius ρ!

   Give a bound on the list size, in terms of ℓ.

4. Towards doing the Interpolation Step, prove the following lemma.

   Lemma 2. *Fix $r \in \mathbb{Z}_N$. Suppose that $B_0, \ldots, B_{\ell-1} \in \mathbb{Z}$ are such that $B_i > 0$, and $\prod_{i=0}^{\ell-1} B_i > N$. Show that there exist $a_0, \ldots, a_{\ell-1} \in \mathbb{Z}$ (not all zero), so that $|a_i| < B_i$ for all i, and so that*

$$\sum_{i=0}^{\ell-1} a_i r^i \equiv 0 \mod N.$$

   Hint: Consider the map $f : \mathbb{Z}_{B_0} \times \cdots \times \mathbb{Z}_{B_{\ell-1}} \to \mathbb{Z}_N$ given by $f(x_0, \ldots, x_{\ell-1}) = \sum_{i=0}^{\ell-1} x_i r^i \mod N$. Use the pigeonhole principle.

5. Suppose that you don't care about the efficiency of the **Interpolation Step.** Using the previous part, what relationship do N, F, K, ℓ need to satisfy in order for you to guarantee the **Interpolation Step** can be done? Translate this to a guarantee on p_n, n, k as well as F, ℓ.

6. Choose $\ell = \sqrt{n/k}$. Put the previous parts together (and pick an appropriate F) to produce a statement like "as long as $\rho \leq$ ____, the code is $(\rho,$ ____$)$-list-decodable with the algorithm above." The ___'s should be in terms of k, n, and the p_i's. It might be convenient to get a guarantee in terms of $\kappa := \log(p_n)/\log(p_1)$.

   You may also assume that $p_n \gg \ell$ and use big-Oh notation in your bound to simplify it.

7. Compare this (both the algorithm and the result) with the Sudan (or Guruswami-Sudan) algorithm for Reed-Solomon codes.

   (Bonus.) Fun thing to think about, if you are familiar with polynomial quotient rings: With the CRT codes, the i'th symbol was $m \mod p_i$. One way to view an RS code is that the i'th symbol is $f(X) \mod (X - \alpha_i)$. Push this analogy as far as you can in the context of the algorithm we just developed.

8. **(Bonus).** What if you want the **Interpolation Step** to be efficient? Would you have to change the parameters?