On Decoding Interleaved Chinese Remainder Codes

Wenhui Li and Vladimir Sidorenko
Institute of Communications Engineering
Ulm University
Ulm, Germany
{wenhui.li,vladimir.sidorenko}@uni-ulm.de

Johan S. R. Nielsen
Department of Applied Mathematics and Computer Science
Technical University of Denmark
Lyngby, Denmark
jsrn@jsrn.dk

***Abstract*—We model the decoding of Interleaved Chinese Remainder codes as that of finding a short vector in a $\mathbb{Z}$-lattice. Using the LLL algorithm, we obtain an efficient decoding algorithm, correcting errors beyond the unique decoding bound and having nearly linear complexity. The algorithm can fail with a probability dependent on the number of errors, and we give an upper bound for this. Simulation results indicate that the bound is close to the truth. We apply the proposed decoding algorithm for decoding a single CR code using the idea of "Power" decoding, suggested for Reed–Solomon codes. A combination of these two methods can be used to decode low-rate Interleaved Chinese Remainder codes.**

***Index Terms*—Interleaved Chinese Remainder codes, Power decoding, Lattice reduction**

I. Introduction

The redundancy property of the Chinese remainder representation of integers has been exploited often in theoretical computer science and many practical applications. In this paper we consider Chinese Remainder (CR) error correcting codes. It was shown by Goldreich, Ron and Sudan [1], that CR codes can be efficiently applied for distributive computations and for secret sharing.

The CR codes are similar to Reed–Solomon (RS) codes in many aspects, and in particular both constructions are maximum distance separable. Decoding algorithms also share deep structure, such as the CR decoding using a Key Equation [2], or CR decoding using Guruswami–Sudan [1], [3]. In recent years, constructions with interleaved RS (IRS) codes have been intensively studied in several publications, e.g. [4], [5]. These constructions allow decoding beyond half the minimum distance and can be applied in concatenated designs. It was also shown how the same technique can be used for decoding a single RS code up to the Sudan radius [6].

In this paper we propose such decoding algorithms for CR and Interleaved CR (ICR) codes. Algebraic similarities means that we can adapt to ICR codes a recent approach by Nielsen [7] for solving multiple Key Equations by finding short vectors in a certain space; on the other hand, algebraic differences mean that the entire analysis is different.

In Section II, we introduce CR codes and lay down notation. In Section III we give the decoder for ICR codes as well as theoretical considerations and simulation results; in Sections IV and V, we discuss how this method extends as Power decoding for single and interleaved CR codes.

W. Li and V. Sidorenko are supported by the German Research Council (Deutsche Forschungsgemeinschaft DFG) under projects Bo 867/22. V. Sidorenko is on leave from the Institute for Information Transmission Problems, Russian Academy of Sciences.

J. S. R. Nielsen gratefully acknowledges the support from the Danish National Research Foundation and the National Science Foundation of China (Grant No.11061130539) for the Danish-Chinese Center for Applications of Algebraic Geometry in Coding Theory and Cryptography.

II. Preliminaries

We begin with defining the classical *Chinese Remainder codes* (CR codes). Let n be the code length and $0 < p_1 < p_2 < \cdots < p_n$ a list $\mathcal{P}$ of n relatively prime positive integers. We construct a polyalphabetic code, where the i-th component of codeword $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ is taken from the alphabet $\mathbb{Z}_{p_i}$, being the ring of integers modulo p_i. Thus the codewords are selected from the code space $\mathbb{Z}_{p_1} \times \mathbb{Z}_{p_2} \times \cdots \times \mathbb{Z}_{p_n}$ of size $N = p_1 p_2 \ldots p_n$. Given $\mathcal{P}$, let us define the function $F(a, b)$ for integers a, b, where $0 < a \leq b \leq n$, as follows

$$F(a,b) = \prod_{i=a}^{b} p_i, \tag{1}$$

and $F(1, 0) = 0$. So, we have $N = F(1, n)$. We also need a *cardinality* K; we will mostly deal with the classical case where K is selected as $K = F(1, k)$ for some $0 < k < n$. As such, k will play a role analogous to the number of information symbols of the code. We introduce the notation that for integers x and y, we denote by $[x]_y$ the remainder when x is divided by y, $0 \leq [x]_y \leq y - 1$.

Definition 1 ((Classical) Chinese Remainder Code)**.** *A Chinese Remainder code $\mathcal{CR}(\mathcal{P}; n, K)$ or shortly $\mathcal{CR}(n, K)$ having cardinality $K = F(1, k)$ for some k, $0 < k < n$ and length n over alphabets $\mathcal{P}$ is defined as follows*

$$\mathcal{CR}(\mathcal{P}; n, K) = \{\ ([C]_{p_1}, \ldots, [C]_{p_n}) : C \in \mathbb{N} \text{ and } C < K\}\,.$$

Assume we have settled on a CR code $\mathcal{CR}(n, K)$, and that we transmit some codeword $\mathbf{c} = (c_1, \ldots, c_n)$ over an additive noisy channel and receive the word $\mathbf{r} = \mathbf{c} + \mathbf{e}$ where $\mathbf{e} = (e_1, \ldots, e_n)$ is the error vector. Letting $\mathbf{r} = (r_1, \ldots, r_n)$ we have $r_i = [c_i + e_i]_{p_i}\ \forall i = 1, \ldots, n$. Let t be the number of errors, i.e. the Hamming weight of $\mathbf{e}$. By the Chinese Remainder Theorem, if the receiver knows any k positions where no errors have occurred, then he can reconstruct C; a

common decoding strategy, which we will use in this paper, is therefore first to identify the erroneous positions.

A CR code is *Maximum Distance Separable* (MDS), that is, its minimum Hamming distance is $d = n-k+1$ [1]. Since each component has different alphabet size p_i, in addition to use the usual Hamming distance, let us define the weighted Hamming distance between words $\mathbf{r}$ and $\mathbf{c}$ as follows

$$d_{\mathcal{P}}(\mathbf{c}, \mathbf{r}) = \sum_{i:r_i \neq c_i} \log p_i.$$

Using the Chinese remainder theorem, we can compute R such that $[R]_{p_i} = r_i$, and likewise an E such that $[E]_{p_i} = e_i$; we then know $R \equiv C + E \mod N$. We will find the position of the errors by determining the *error-locator* Λ, defined as:

$$\Lambda = \prod_{i:r_i \neq c_i} p_i.$$

Thus $d_{\mathcal{P}}(\mathbf{c}, \mathbf{r}) = \log \Lambda$. Define $D_t = F(n-t+1, n)$ as the maximal value of Λ given that at most t errors have occurred.

An easy but important observation is, see e.g. [2], $N \mid (\Lambda E)$. This immediately leads to a Key Equation:

$$\Lambda R \equiv \Lambda C \mod N. \tag{2}$$

For a not too large number of errors, then $\Lambda R \gg N$ while $\Lambda C < D_t K \ll N$, and Λ turns out to be the only relatively small number such that $[\Lambda R]_N$ is also small:

Lemma 1 ([1], Lemma 5)**.** *If $d_{\mathcal{P}}(\mathbf{c}, \mathbf{r}) \leq \log(\sqrt{N/(K-1)})$ then the decoding algorithm in [1] finds Λ using* (2)*.*

The decoder of Lemma 1 succeeds whenever $\log D_t \leq \log(\sqrt{N/K}) < \log(\sqrt{N/(K-1)})$, but we can relax this to a decoding radius in the weighted Hamming metric. Using $D_t < p_n^t$ we thus get

$$t \leq \left\lfloor \frac{1}{2} \cdot \frac{\log(N/K)}{\log p_n} \right\rfloor. \tag{3}$$

III. Interleaved Chinese Remainder Codes

Interleaving is a technique for making long codes from shorter ones which efficiently handle burst errors. Codewords are now matrices where each row is a codeword coming from some component code. Errors are assumed to arrive in bursts, altering entire columns. One can correct each component codeword individually, but utilizing that the number of erroneous columns is low, one can do better by decoding collaboratively.

Definition 2 (Interleaved Chinese Remainder Code)**.** *Consider ℓ classical CR codes $\mathcal{CR}(\mathcal{P}; n, K_l), l \in 1, \ldots, \ell$. Denote the list $K_1, K_2, \ldots, K_\ell$ by $\mathcal{K}$. The Interleaved Chinese Remainder code $\mathcal{ICR}(\mathcal{P}; n, \mathcal{K})$ or shortly $\mathcal{ICR}(n, \mathcal{K})$ is defined as the set of matrices*

$$\begin{pmatrix} c_1^{(1)} & c_2^{(1)} & \ldots & c_n^{(1)} \\ c_1^{(2)} & c_2^{(2)} & \ldots & c_n^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ c_1^{(\ell)} & c_2^{(\ell)} & \ldots & c_n^{(\ell)} \end{pmatrix}$$

where $\mathbf{c}^{(l)} = (c_1^{(l)}, \ldots, c_n^{(l)}) \in \mathcal{CR}(n, K_l), l = 1, \ldots, \ell$.

For the remainder of this section, consider some received matrix with rows $\mathbf{r}^{(1)}, \ldots, \mathbf{r}^{(\ell)}$ each with $\mathbf{r}^{(l)} = \mathbf{c}^{(l)} + \mathbf{e}^{(l)}$ for some error row $\mathbf{e}^{(l)}$. We now define a complete error-locator which identifies all columns having any errors, i.e.,

$$\Lambda = \prod_{i:\exists l: r_i^{(l)} \neq c_i^{(l)}} p_i.$$

When we refer to "the number of errors", it is also the number of factors in the above product.

A. Solving Key Equations

For each $\mathbf{c}^{(l)}$ and $\mathbf{r}^{(l)}$ corresponds a $C^{(l)}$ respectively $R^{(l)}$. For a particular row l, even though the complete error-locator Λ might be a multiple of that row's error-locator, the Key Equation (2) still holds with Λ; thus, to collaboratively decode the ICR code, we want to solve a system of ℓ Key Equations as follows

$$\begin{cases} \Lambda R^{(1)} & \equiv & \Lambda C^{(1)} & \mod N \\ \Lambda R^{(2)} & \equiv & \Lambda C^{(2)} & \mod N \\ \quad \vdots & & & \\ \Lambda R^{(\ell)} & \equiv & \Lambda C^{(\ell)} & \mod N \end{cases}. \tag{4}$$

Recently, Nielsen [7] used a module minimization approach to solve multiple Key Equations over some polynomial ring $\mathbb{F}[x]$, such as those arising when decoding Interleaved Reed–Solomon codes. We will apply essentially the same approach for our Key Equations, but the algebraic differences between $\mathbb{F}[x]$ and $\mathbb{Z}$ implies fundamental differences in the final algorithms.

The l-th Key Equation means that there exists some $v_l \in \mathbb{Z}$ such that $\Lambda R^{(l)} - v_l N = \Lambda C^{(l)}$. We can collect these ℓ equations into one in a vectorized form and say that $\mathbf{s} = (\Lambda, \Lambda C^{(1)}, \ldots, \Lambda C^{(\ell)})$ must be a vector in the $\mathbb{Z}$-row space of the matrix

$$\mathbf{M} = \begin{pmatrix} 1 & R^{(1)} & R^{(2)} & \ldots & R^{(\ell)} \\ 0 & N & 0 & \ldots & 0 \\ 0 & 0 & N & \ldots & 0 \\ & & & \ddots & \\ 0 & 0 & 0 & \ldots & N \end{pmatrix}. \tag{5}$$

The crucial observation is now that whenever few errors have occurred, $\mathbf{s}$ is often the *shortest* vector in the row space of $\mathbf{M}$; we will explain and formalize this later with Theorem 1. By "short" we mean the L_2 norm, and to increase the probability that $\mathbf{s}$ is the shortest vector, we will actually regard the row space of $\mathbf{M}_\omega$, a weighted version of $\mathbf{M}$, where we scale the i-th column with some $\omega_i \in \mathbb{Z}$. We are thus seeking a $\mathbf{s}_\omega = (\Lambda\omega_0, \Lambda C^{(l)}\omega_1, \ldots, \Lambda C^{(l)}\omega_\ell)$. We get back to how exactly we assign the ω_i in Corollary 1.

Computing the shortest vector in the row space of a matrix under the L_2 norm is unfortunately an $\mathcal{NP}$-hard problem [8]; however, the Lenstra–Lenstra–Lovász (LLL) algorithm [9] is an efficient method to find a vector which is *close* to the shortest one, i.e. it finds a vector whose L_2 norm is at most $\gamma\|\mathbf{v}\|$, where $\mathbf{v}$ is a shortest vector and γ is a constant. In the worst case, $\gamma = \sqrt{2}^{\ell+1}$, where $\ell+1$ is the dimension of the row

space; however, experiments indicate that in random instances the LLL and its modifications usually do much better, with $\gamma \approx 1.02^{\ell+1}$ [10]. To be certain that our computation will lead us to $\mathbf{s}_\omega$, we must therefore not only be sure that $\mathbf{s}_\omega$ is the shortest vector in the row space, but that there are no other vectors of length at most $\gamma\|\mathbf{s}_\omega\|$.

Theorem 1 essentially says that whenever not too many errors have occurred, this is indeed almost always the case. Therefore, one can construct $\mathbf{M}_\omega$, apply the LLL algorithm to find a short vector in it, and with high probability, the output will be $\mathbf{s}_\omega$. This immediately leads to the decoding algorithm given as Algorithm 1.

Algorithm 1: Decoding ICR code $\mathcal{ICR}(n, \mathcal{K})$

Input: The lists $\mathcal{P}$ and $\mathcal{K}$, the received words $\mathbf{r}^{(l)},\ l = 1, \ldots, \ell,\ N$

Output: The error-locator Λ or Fail

Preprocessing: $\omega_0, \ldots, \omega_\ell$ according to Corollary 1

1 Compute $R^{(1)}, \ldots, R^{(\ell)}$.
2 Construct $\mathbf{M}$ as in (5) and multiply the ith column by ω_i for $i = 0, \ldots, \ell$.
3 Run the LLL algorithm which returns a short vector $\mathbf{v}_\omega$.
4 If the zeroth position of $\mathbf{v}_\omega$ has the form $\omega_0\Lambda$ where Λ is a valid error-locator, then return Λ. Otherwise, return Fail.

B. Failure Probability

With the overall idea explained, we can go on to analyze the probability that the above algorithm will fail, and from this derive how to assign the ω_i. Our failure probability will depend on the unknown Λ, but we discuss in Section III-C how this can be interpreted as a decoding radius.

The algorithm fails when there is a vector in the row space $\mathbf{M}_\omega$ different from $\mathbf{s}_\omega$ but which has L_2 norm within $\gamma\|\mathbf{s}_\omega\|$, and we will upper bound this probability. Our theorem will assume that certain values behave as independent, uniformly distributed random variables, and so we will need the following lemma:

Lemma 2. *Given some $N, T \in \mathbb{Z}$ with $T < N$ and $c_1, \ldots, c_\ell \in \mathbb{Z}_+$, and let $X_1, \ldots, X_\ell$ be independent discrete random variables, uniformly distributed on $0, \ldots, N-1$. Then*

$$\text{Prob}[c_1X_1 + \ldots + c_\ell X_\ell < T] \leq \frac{T^\ell}{\ell! N^\ell c_1 \cdots c_\ell}.$$

Theorem 1. *Let A be a random variable, uniformly distributed on $1, \ldots, \lfloor T/\omega_0 \rfloor$, where T is defined below, and assume then that we can regard $(AR^{(l)} \mod N)$ for $l = 1, \ldots, \ell$ as ℓ independent random variables, uniformly distributed on $0, \ldots, N-1$.*

Assume that the LLL algorithm finds a vector whose L_2 norm is at most $\gamma\|\mathbf{v}_\omega\|$ where $\mathbf{v}_\omega$ is a shortest vector in the row space of $\mathbf{M}_\omega$. For a random error-locator Λ, the probability of decoding failure $P_f(\Lambda)$ satisfies

$$P_f(\Lambda) \leq 1 - \left(1 - \frac{T^\ell}{\ell! N^\ell \omega_1 \cdots \omega_\ell}\right)^{T/\omega_0}$$

where $T = \tilde{\gamma} \max\{\omega_0\Lambda, \omega_1\Lambda K_1, \ldots, \omega_\ell\Lambda K_\ell\}$ and $\tilde{\gamma} = \sqrt{\gamma(\ell+1)}$.

Proof: (sketch) The decoder can only fail if there is a vector $\mathbf{v}_\omega \neq \mathbf{s}_\omega$ with $\|\mathbf{v}_\omega\| < \gamma\|\mathbf{s}_\omega\|$, i.e.,

$$\sum_{j=0}^{\ell} (\omega_j v_j)^2 < \gamma \left((\omega_0\Lambda)^2 + \sum_{j=1}^{\ell} (\omega_j \Lambda C^{(j)})^2 \right)$$

where $\omega_j v_j$ are the components of $\mathbf{v}_\omega$. Let $\tilde{T} = \max\{\omega_0\Lambda, \omega_1\Lambda K_1, \ldots, \omega_\ell\Lambda K_\ell\}$; then the above can only occur if $\sum_{j=0}^{\ell} (\omega_j v_j)^2 < \gamma(\ell+1)\tilde{T}^2$. Due to the Cauchy–Schwartz inequality, that implies

$$\sum_{j=0}^{\ell} \omega_j v_j < \sqrt{\gamma(\ell+1)}\tilde{T} = T. \tag{6}$$

We will upper bound $P_f(\Lambda)$ by the probability that a vector $\mathbf{v}_\omega \neq \mathbf{s}_\omega$ satisfying (6) is in the row space. Such a vector can be written in the form

$$\big(\omega_0 A,\ \omega_1(AR^{(1)} \bmod N),\ \ldots,\ \omega_\ell(AR^{(\ell)} \bmod N)\big)$$

where $\omega_0 A \in \{1, \ldots, T-1\}$. Now we use Lemma 2 to over-approximate the probability that for a given $A \in \{1, \ldots, \lfloor T/\omega_0 \rfloor\}$, the associated vector of the above form satisfies (6):

$$\begin{aligned} P &= \text{Prob}\Big[\sum_{j=1}^{\ell} \omega_i (AR^{(j)} \bmod N) < T - \omega_0 A \Big] \\ &< \text{Prob}\Big[\sum_{j=1}^{\ell} \omega_i (AR^{(j)} \bmod N) < T \Big] \\ &\leq \frac{T^\ell}{\ell! N^\ell \omega_1 \ldots \omega_\ell}. \end{aligned} \tag{7}$$

Thus the probability that none of the T/ω_0 choices of A satisfies (6) becomes at least $(1-P)^{T/\omega_0}$, and the statement follows. ∎

Though we have not proved that the following choice of weights is optimal, it seems intuitive:

Corollary 1. *With the weights chosen as $\omega_0 = K_\ell$ and $\omega_i = K_\ell/K_i,\ i = 1, \ldots, \ell$, the failure probability becomes*

$$P_f(\Lambda) \leq 1 - \left(1 - \frac{\tilde{\gamma}^\ell \Lambda^\ell \prod_{l=1}^{\ell} K_l}{\ell! N^\ell}\right)^{\tilde{\gamma}\Lambda}.$$

C. Discussion on the decoding radius

It would be nice to give a statement such as "Algorithm 1 can decode up to t errors" for some definition of t. The guaranteed unique decoding radius t_g of the ICR code is given by (3), where $K = \max\{K_l\}$ since any unique decoder must fail with non-zero probability when $t > t_g$. However, one could almost always find Λ using the best protected code, giving a "usual" decoding radius t_u from $K = \max\{K_l\}$. Thus, the traditional definition of decoding radius is not very useful. Exactly the same applies for the collaborative decoders of Reed–Solomon codes [4], [5].

An alternative is to define a threshold, and say that "Algorithm 1 decodes a random error pattern of weight t with probability $1-\phi$". One could then set ϕ satisfactorily low. For this definition, one can use the failure probability estimated in Theorem 1 as a starting point. Given t, since all error patterns are equally likely, each Λ with t factors occurs equally often; thus the probability of failure for a given number of errors t is

$$\bar{P}_f(t) = \binom{n}{t}^{-1} \sum_I P_f(p_{I_1} \cdots p_{I_t})$$

where the sum runs over all subsets of $\{1, \ldots, n\}$ of size t.

We are in the process of performing this analysis, but our preliminary results suggests that for a reasonably defined ϕ, the decoding radius of our algorithm, in the above sense, would be of the form

$$t \lesssim \left\lfloor \alpha \frac{\ell}{\ell+1} \frac{\log(N/\bar{K})}{\log p_n} \right\rfloor \tag{8}$$

where $\bar{K} = \sqrt[\ell]{K_1 \cdots K_\ell}$ and α is some constant close to 1 which depends on the code parameters and ϕ. To decode a single CR code ($\ell = 1$), (8) coincides with (3) for $\alpha = 1$.

D. Complexity

Let us begin our complexity analysis by discussing step 3 of Algorithm 1, namely running the LLL. By [11, Theorem 16.11], this performs $O(\ell^4 \log Z)$ operations on integers of bit-length $O(\ell \log Z)$ where Z is the greatest integer in $\mathbf{M}_\omega$. Choosing the ω_i as in Corollary 1, we clearly have $Z = NK_\ell/K_1$ which means $\log Z < (n + k_\ell - k_1)\log(p_n) < 2n\log(p_n)$. It is quite easy to see that the remaining computations of Algorithm 1 can be performed faster than this. In particular, since we know which primes are allowed to divide a valid error-locator, the check in step 4 can be done efficiently.

Thus, the complexity of Algorithm 1 is $O(n\ell^4 \log(p_n))$ operations on integers of bit-length $O(n\ell \log(p_n))$.

E. Test Results

We have done quite extensive testing of the algorithm, and have in general observed that the failure probability of Theorem 1 corresponds rather well with experiments: setting $\gamma = 1$, i.e. expecting the LLL to always find the shortest vector, sometimes proves slightly too optimistic, while setting $\gamma = \sqrt{2}^{\ell+1}$, i.e. the worst case, is overly pessimistic. The difference between these two is usually within only a few errors, though.

As an example, consider the ICR code $\mathcal{ICR}(n = 20, \mathcal{K} = [3, 5])$, i.e., interleaving factor $\ell = 2$, and with the prime list $\mathcal{P} = [101, 103, \ldots, 197]$. The guaranteed decoding radius for this ICR code is $t_g = 7$, while the "usual" radius is $t_u = 8$. Choosing the weights as in Corollary 1, we have run 10,000 tests with this code, creating random error patterns of weights ranging from 7 up to 12. For each number of errors t, we then calculated the following aggregate statistics

$$A_{Obs} = \#\text{failures}/\#\text{Tests}_t$$
$$A_T^{\hat{\gamma}} = \textstyle\sum_{\Lambda \in \text{Tests}_t} P_f^{\gamma=\hat{\gamma}^{\ell+1}}(\Lambda)/\#\text{Tests}_t,$$

the latter calculated for $\hat{\gamma} \in \{1, 1.02, \sqrt{2}\}$. We have also calculated $P_T^{\hat{\gamma}} = P_f^{\gamma=\hat{\gamma}^{\ell+1}}(D_t)$, i.e., the failure probability of the biggest Λ.

Table I summarizes our results. We see that the observed decoding failure rather sharply goes from 0% to 100%, and we see that the theoretical failure probabilities come very close to the observed behavior. It is interesting to note that with $\alpha = 1$, (8) evaluates to 10.

t	A_{Obs}	A_T^1	$A_T^{1.02}$	$A_T^{\sqrt{2}}$	P_T^1	$P_T^{1.02}$	$P_T^{\sqrt{2}}$
9	0%	0%	0%	0%	0%	0%	0%
10	0%	0%	0%	0.01%	0.25%	0.27%	1.18%
11	96.06%	98.49%	98.68%	99.86%	100%	100%	100%
12	100%	100%	100%	100%	100%	100%	100%

TABLE I
FAILURE PROBABILITIES ($n = 20$)

At a much higher rate, for example consider the ICR code $\mathcal{ICR}(n = 100, \mathcal{K} = [81, 81, 82, 82, 83])$, i.e., $\ell = 5$, with the prime list $\mathcal{P} = [101, 103, \ldots, 691]$. The guaranteed decoding radius for this ICR code is $t_g = 8$, while the "usual" radius is $t_u = 9$. Running 10,000 tests with patterns of weights ranging from 14 to 18, and aggregating as before, we got the test results as given on Table II. We see that the upper bounds of $P_T^{\hat{\gamma}}$ are quite pessimistic estimates on the average failure probability, and that it is slightly too optimistic to assume $\hat{\gamma} \leq 1.02$. According to (8) with $\alpha = 1$, the decoding radius is $t = 14$, which is also pessimistic.

t	A_{Obs}	A_T^1	$A_T^{1.02}$	$A_T^{\sqrt{2}}$	P_T^1	$P_T^{1.02}$	$P_T^{\sqrt{2}}$
14	0%	0%	0%	0%	0%	0%	0%
15	0%	0%	0%	0%	0%	0%	0%
16	4.68%	3.51%	3.81%	10.71%	99.77%	99.98%	100%
17	89.66%	87.25%	87.79%	94.84%	100%	100%	100%
18	99.94%	99.95%	99.95%	100%	100%	100%	100%

TABLE II
FAILURE PROBABILITIES ($n = 100$)

IV. POWER DECODING OF LOW RATE CR CODE

The key equation (2) for a single CR code can be "virtual extended" to multiple Key Equations whenever $K \ll N$; this technique, called "Power decoding", was described for Reed–Solomon in [6]. The resulting "virtually interleaved" code can be decoded by interleaved coding techniques beyond the unique decoding bound.

Each element of the received word is powered to be an element of a new CR code, i.e.,

$$\begin{aligned}\mathbf{r}^{(l)} &= ([r_1^l]_{p_1}, [r_2^l]_{p_2}, \ldots, [r_n^l]_{p_n}) \\ &= ([(c_1 + e_1)^l]_{p_1}, [(c_2 + e_2)^l]_{p_2}, \ldots, [(c_n + e_n)^l]_{p_n}) \\ &= ([c_1^l]_{p_1} + \tilde{e}_1, [c_2^l]_{p_2} + \tilde{e}_2, \ldots, [c_n^l]_{p_n} + \tilde{e}_n),\end{aligned}$$

where $\tilde{e}_i = [(c_1 + e_1)^l - c_1^l]_{p_i}$. Note that the error positions do not change under powering. Therefore, a single CR code is virtually extended to an ICR code where each row has the same error-locator. The cardinality of the new code $K_j = K^j$

can not be expressed by $F(\cdot)$, so these codes are not part of the classical definition. The obvious generalized definition is (see e.g. [12]) to allow any $0 \leq K \leq N$ as the code cardinality.

The Key Equation (2) can easily becomes virtually extended as well, recalling that $N \mid \Lambda E$:

$$\begin{aligned} \Lambda R^l \mod N &\equiv \Lambda(C+E)^l \mod N \\ &\equiv \Lambda C^l \mod N. \end{aligned}$$

Let ℓ be the greatest integer such that $\Lambda C^\ell < N$; the ℓ Key Equations for $l = 1, \ldots, \ell$ can be used to collaboratively determine Λ, and we can use exactly the same approach as we did for ICR codes.

Consider the $\mathbb{Z}$-row space of the matrix:

$$\left(\begin{array}{c|ccc} 1 & R & [R^2]_N & \ldots & [R^\ell]_N \\ \hline \mathbf{0} & \multicolumn{3}{c}{N\mathbf{I}} \end{array}\right), \tag{9}$$

where $\mathbf{I}$ is the $\ell \times \ell$ identity matrix. By the above Key Equations, the vector $(\Lambda, \Lambda C, \ldots, \Lambda C^\ell)$ will be in this space, and as in the case for ICR codes, it will be surprisingly short. We should choose weights for the columns, and emulating the choice of Corollary 1, we let $\omega_0 = K^\ell$ and $\omega_j = K^{\ell-l}$ for $l = 1, \ldots, \ell$.

The failure probability of Theorem 1 can be reused for this case. However, one should be noted that this is under heavier assumptions of randomness, since the various R-values, $R, [R^2]_N, \ldots, [R^\ell]_N$ obviously are more connected than for the usual ICR setting. We have by simulation confirmed that the approach works and we can decode beyond the unique decoding bound; however, more experimentation is needed for proper verification that the failure probabilities are well-estimated.

Above, we chose ℓ depending on the unknown Λ and C, which is obviously problematic. Instead, one could choose a decoding radius t and choose ℓ maximal such that $D_t K^\ell < N$. However, since more interleaving allows higher decoding radius, there is a non-trivial connection here. Furthermore, since random Λ are usually much lower than D_t and random C lower than K, it might be the case that the decoding case at hand would benefit from a higher interleaving factor. We have not thoroughly investigated this issue.

V. Decoding of Low-Rate ICR Codes

Power decoding can be straightforwardly combined with the ICR decoder, whenever one interleaves CR codes of low rate; this idea was first proposed for Reed–Solomon codes in [5]. We will briefly sketch the idea, but we have not yet deeply analyzed this setting.

Consider a code $\mathcal{ICR}(\mathcal{P}; n, \mathcal{K} = [K_1, \ldots, K_\ell])$ as well as received matrix with rows $\mathbf{r}_1, \ldots, \mathbf{r}_\ell$. Define the corresponding $R_1, \ldots, R_\ell$. For each of these, we can get virtually extended Key Equations $\Lambda[R_i^j]_N \equiv \Lambda C_i^j \mod N$ for $j = 1, \ldots, \rho_i$, where ρ_i is chosen maximally such that $\Lambda C_i^{\rho_i} < N$. This means that vector $\left(\Lambda, \Lambda C_1, \ldots, \Lambda C_1^{\rho_1}, \ldots, \Lambda C_\ell, \ldots, \Lambda C_\ell^{\rho_\ell}\right)$ is in the row space of the matrix:

$$\left(\begin{array}{c|ccccccc} 1 & [R_1^1]_N & \ldots & [R_1^{\rho_1}]_N & \ldots & [R_\ell^1]_N & \ldots & [R_\ell^{\rho_\ell}]_N \\ \hline \mathbf{0} & \multicolumn{7}{c}{N\mathbf{I}} \end{array}\right)$$

where $\mathbf{I}$ is an appropriately sized identity matrix From here the decoding algorithm progress as in Algorithm 1.

The issue with how to choose the ρ_i is even more compounded in this setting than for the Power decoding, and more analysis is needed for determining the right choice while minimizing computational effort. We also note that one can perform a "mixing" of the Key Equations, as for IRS codes in [13], to get a larger matrix and decode more errors.

VI. Conclusion

We proposed a collaborative LLL-based decoding algorithm for Interleaved Chinese Remainder codes. The time complexity of the algorithm is nearly linear in the length of the code. We analyzed the failure probability, and simulation results showed that these bounds well characterize observed behavior. Just as for the case of Reed–Solomon codes, the ICR decoder extends straightforwardly to Power decoding of a single, low-rate Chinese Remainder code, and both techniques can be combined for Interleaved Chinese Remainder codes with low rate.

Deeper analysis is needed for providing a simple, closed-form characterization of the decoding radius, as well as optimal application of the Power decoding technique.

References

[1] O. Goldreich, D. Ron, and M. Sudan, "Chinese remaindering with errors," *IEEE Trans. Inform. Theory*, vol. 46, no. 4, pp. 1330–1338, Jul. 2000.

[2] W. Li, "On Syndrome Decoding of Chinese Remainder Codes," in *The 13th Int. Workshop on Algebraic and Combinatorial Coding Theory*, 2012.

[3] V. Guruswami, A. Sahai, and M. Sudan, ""Soft-decision" decoding of Chinese remainder codes," in *Proc. of the 41st IEEE Symposium on Foundations of Computer Science*, 2000, pp. 159–168.

[4] G. Schmidt, V. Sidorenko, and M. Bossert, "Collaborative decoding of interleaved Reed–Solomon codes and concatenated code designs," *IEEE Trans. Inform. Theory*, vol. 55, no. 7, pp. 2991–3012, 2009.

[5] ——, "Enhancing the Correcting Radius of Interleaved Reed-Solomon Decoding using Syndrome Extension Techniques," in *IEEE Int. Symposium on Inform. Theory*. IEEE, Jun. 2007, pp. 1341–1345.

[6] ——, "Syndrome Decoding of Reed–Solomon Codes Beyond Half the Minimum Distance Based on Shift-Register Synthesis," *IEEE Trans. Inform. Theory*, vol. 56, no. 10, pp. 5245–5252, Oct. 2010.

[7] J. S. R. Nielsen, "Generalised Multi-sequence Shift-Register Synthesis using Module Minimisation," in *IEEE Int. Symposium on Inform. Theory*, 2013.

[8] M. Ajtai, "The shortest vector problem in l2 is np-hard for randomized reductions (extended abstract)," in *Proc. of the 30th annual ACM symposium on Theory of computing*, ser. STOC '98. New York, NY, USA: ACM, 1998, pp. 10–19.

[9] A. K. Lenstra, "Factoring multivariate polynomials over finite fields," *J. Computer System Sciences*, vol. 30, no. 2, pp. 235–248, 1985.

[10] P. Q. Nguyen and D. Stehlé, "LLL on the average," in *Algorithmic Number Theory*. Springer, 2006, pp. 238–256.

[11] J. v. z. Gathen and J. Gerhard, *Modern Computer Algebra*. Cambridge University Press, Jul. 2003.

[12] W. Li and V. Sidorenko, "On the error-erasure-decoder of the Chinese remainder codes," in *Problems of Redundancy in Inform. and Control Systems, XIII Int. Symposium on*. IEEE, 2012, pp. 37–40.

[13] A. Wachter-Zeh, A. Zeh, and M. Bossert, "Decoding interleaved Reed–Solomon codes beyond their joint error-correcting capability," *Designs, Codes and Cryptography*, Jul. 2012.