A Progressive Chase Type List Decoding Algorithm for Reed-Solomon Codes

Jiancheng Zhao, Li Chen
School of Information Science and Technology, Sun Yat-sen University, Guangzhou, China

Abstract—In this paper, a progressive Chase type list decoding algorithm for Reed-Solomon (RS) codes is proposed. The decoding starts with constructing a set of $2^n$ ($\eta > 0$) test-vectors which will be arranged in a descending order according to their potential of yielding the intended message. Interpolation and factorization will be performed based on the ordered test-vectors. It is shown significant performance improvement can be achieved over Koetter-Vardy (KV) algorithm with a reduced complexity.

I. INTRODUCTION

We consider an $(n, k)$ Reed-Solomon (RS) code defined in the finite field $\mathbb{F}_q$, and $n = q - 1$. Given a message vector $m = (m_0, m_1, \ldots, m_{k-1}) \in \mathbb{F}_q^k$, the corresponding message polynomial is $m(x) = \sum_{u=0}^{k-1} m_u x^u$. Let $\{\alpha_0, \alpha_1, \ldots, \alpha_{n-1}\} \in \mathbb{F}_q$, the codeword is generated by $c = (c_0, c_1, \ldots, c_{n-1}) = (m(\alpha_0) m(\alpha_1) \cdots m(\alpha_{n-1})) \in \mathbb{F}_q^n$.

In the Chase type list decoding (CLD) algorithm [1], the reliability matrix $\Pi \in \mathbb{F}_{q \times n}$ will be given as the soft information observed from the channel. Let $\pi_{y_j}$ denote the entry of $\Pi$, indicating $\Pr[c_j = \rho_j]$, and $\rho_j \in \mathbb{F}_q$. Let $y_1^j$ and $y_2^j$ denote the most likely and the second most likely symbols for $c_j$. We define $\gamma_j = \Pr[c_j = y_1^j] / \Pr[c_j = y_2^j]$ as the reliability of the $j$th received symbol. With $\gamma_j \rightarrow 1$, the symbol is less reliable and the $\eta$ ($\eta > 0$) least reliable symbols will be identified. We define $\Phi = \{\phi_1, \phi_2, \ldots, \phi_\eta\}$ as the set of those symbols’ indices, and $\overline{\Phi} = \{0, 1, \ldots, n - 1\} \setminus \Phi$.

The test-set is constructed as the set of all vectors $y_i = (y_{i,0}, y_{i,1}, \ldots, y_{i,n-1})$, where $i = 0, 1, \ldots, 2^n - 1$, $y_{i,j} = y_j^i$, if $j \in \Phi$, and $y_{i,j} \in \{y_1^j, y_2^j\}$, if $j \notin \Phi$. Moreover, the $k$ most reliable symbols will be selected to perform the re-encoding transform. For the $2^n$ test-vectors, $2^n$ bivariate polynomials $Q_i(x, y)$ will be found by interpolation which is performed based on the re-encoding outcome, forming a polynomial set $\{Q_0(x, y), Q_1(x, y), \ldots, Q_{2^n-1}(x, y)\}$. After factorizing all these polynomials, at most $2^n$ message vectors can be attained and the one that corresponds to the most likely codeword will be selected. In order to facilitate the decoding, the CLD can be performed in a progressive manner [2].

II. PROGRESSIVE CHASE TYPE LIST DECODING

For the $n$ received symbols of each test-vector, $n$ corresponding reliability entries can be picked up from each column of $\Pi$. In the progressive CLD (PCLD) algorithm, all the test-vectors will be arranged in a descending order according to their potential of yielding the intended message and the potential can be assessed by the following function $\Omega_i = \prod_{j=0}^{\eta-1} \{\pi_{y_j}[y_{i,j} = \rho_j]\}$. If a test-vector $y_i$ has larger $\Omega_i$ value, it is more reliable and should be guaranteed a priority to be decoded. Hence, interpolation and factorization will be performed for the test-vectors according to the above order. Once a decoded codeword satisfies the maximum likelihood (ML) criterion, the decoding will be terminated. Otherwise, the most likely codeword will again be selected.

III. SIMULATION RESULT

Fig. 1 shows the performance of the PCLD algorithm in decoding the $(15, 11)$ RS code. It can be seen that with $\eta > 1$, the PCLD algorithm prevails the optimal Koetter-Vardy (KV) decoding performance. Table I shows the average number of finite field arithmetic computations that is required to decode a codeword frame. It shows the PCLD algorithm offers a significant complexity reduction over the CLD and KV (with decoding output list size $l = 3$) algorithms. More importantly, its complexity is channel dependent and can be reduced by increasing the SNR. This is because with the above ordering, a valid output can be delivered at an earlier decoding stage.

REFERENCES
