A simple algorithm for decoding both errors and erasures of Reed-Solomon codes

Sergei V. Fedorenko
Department of safety in information systems
St.Petersburg State University of Aerospace Instrumentation
190000, Bolshaya Morskaia, 67, St.Petersburg, Russia
Email: sfedorenko@ieee.org

Abstract
A simple algorithm for decoding both errors and erasures of Reed-Solomon codes is described.

I. INTRODUCTION

In this paper, the Gao algorithm modification is given. In the author’s opinion, the suggested algorithm is the simplest for algebraic codes with short lengths for any implementation.

II. DEFINITIONS AND NOTATIONS

Let us define the \((n, k, d)\) Reed-Solomon code over \(GF(q)\) with length \(n = q - 1\), number of information symbols \(k\), designed distance \(d = n - k + 1\), where \(q\) is prime power.

The message polynomial of the Reed-Solomon code is

\[ M(x) = \sum_{i=0}^{k-1} m_i x^i. \]

The component \(c_i\) of the codeword \(C(x)\) is computed as

\[ c_i = M(\alpha^i), \quad i \in [0, n - 1]. \]

The received vector is represented as a polynomial

\[ R(x) = \sum_{i=0}^{n-1} r_i x^i = C(x) + E(x) = \sum_{i=0}^{n-1} c_i x^i + \sum_{i=0}^{n-1} e_i x^i, \]

where \(C(x)\) is the codeword, \(E(x)\) is the error vector.

The error vector \(E(x)\) has \(t\) errors with a set of error positions \(\{i_1, i_2, \ldots, i_t\}\). Let us define that \(Z_1 = \alpha^{i_1}, Z_2 = \alpha^{i_2}, \ldots, Z_t = \alpha^{i_t}\) are error locations.

The error locator polynomial is

\[ W(x) = \prod_{i=1}^{t} (x - Z_i), \]

where \(t\) is the number of errors, \(Z_i\) is the error location of the error vector \(E(x)\).
The error vector $E(x)$ has $l$ erasures with a set of erasure positions $S = \{j_1, j_2, \ldots, j_l\}$. 

$X_1 = \alpha^{j_1}, X_2 = \alpha^{j_2}, \ldots, X_l = \alpha^{j_l}$ are erasure locations.

The erasure locator polynomial is

$$\Lambda(x) = \prod_{i=1}^{l} (x - X_i),$$

where $l$ is the number of erasures, $X_i$ is the erasure location of the error vector $E(x)$.

The inequality $2t + l < d$ is well known [1].

We construct an interpolating polynomial $T(x)$ such that

$$T(\alpha^i) = r_i, \quad i \in [0, n-1],$$

where $\deg T(x) < n$, and an interpolating polynomial $\mathcal{T}(x)$ such that

$$\mathcal{T}(\alpha^i) = r_i, \quad i \in [0, n-1] \setminus S,$$

where $\deg \mathcal{T}(x) < n - l$.

### III. Existing Algorithms

We describe here two versions of the Gao algorithm [2], [3], [4], [5].

The first version is for decoding errors only. Let $P(x) = W(x)M(x)$. The key equation is

$$\begin{cases} W(x)T(x) \equiv P(x) \mod x^n - 1 \\ \deg W(x) \leq \frac{d-1}{2} \\ \text{maximize } \deg W(x). \end{cases}$$

(1)

The asymptotic complexity of this algorithm is $O(n(\log n)^2)$.

The second version is for decoding both errors and erasures. The key equation is

$$\begin{cases} W(x)\mathcal{T}(x) \equiv P(x) \mod \frac{x^n - 1}{\Lambda(x)} \\ \deg W(x) \leq \frac{d-l-1}{2} \\ \text{maximize } \deg W(x). \end{cases}$$

(2)

The direct computation by this algorithm has complexity $O(n^2)$.

Next, we consider the key equation derivation for the Truong algorithm [6] for decoding both errors and erasures. Let

$$Q(x) = P(x)\Lambda(x) = W(x)M(x)\Lambda(x).$$

From (1) we have

$$W(x)((T(x)\Lambda(x)) \equiv (P(x)\Lambda(x)) \mod x^n - 1$$

and the key equation is

$$\begin{cases} W(x)((T(x)\Lambda(x)) \equiv Q(x) \mod x^n - 1 \\ \deg W(x) \leq \frac{d-l-1}{2} \\ \text{maximize } \deg W(x). \end{cases}$$

(3)

The asymptotic complexity of this algorithm coincides with the complexity of decoding algorithms [2], [3], [4], [5].
IV. SUGGESTED ALGORITHM

We introduce the following lemma.

**Lemma:**

\[ T(x) \equiv T(x) \mod \frac{x^n - 1}{\Lambda(x)}. \]

**Proof:** From Newton’s interpolation formula we obtain

\[ T(x) = \frac{x^n - 1}{\Lambda(x)} U(x) + T(x), \]

where \( U(x) \) is a polynomial.

From (2) and the lemma we get a new key equation

\[
\begin{cases}
W(x)T(x) \equiv P(x) \mod \frac{x^n - 1}{\Lambda(x)} \\
\deg W(x) \leq \frac{d-l-1}{2} \\
\text{maximize } \deg W(x).
\end{cases} \tag{4}
\]

The description of the three algorithms for decoding both errors and erasures is in table 1.

V. CONCLUSION

The suggested algorithm has replaced the computation using Newton’s interpolation formula by the fast computation of the discrete Fourier transform. The algorithm complexity is less than the Truong algorithm [6] complexity because the suggested algorithm does not contain some of the intermediate computations.

REFERENCES

<table>
<thead>
<tr>
<th>Step</th>
<th>Gao’s algorithm</th>
<th>Truong’s algorithm</th>
<th>Suggested algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$T(x)$</td>
<td>$T(x)\Lambda(x)$</td>
<td>$\Lambda(x)$</td>
</tr>
<tr>
<td>1</td>
<td>$T(x) = \frac{x^n - 1}{\Lambda(x)}$</td>
<td>$T(x)\Lambda(x)$</td>
<td>$\frac{P(x)}{W(x)}$</td>
</tr>
<tr>
<td>2a</td>
<td>$W(x)T(x) = P(x)$</td>
<td>$Q(x)\Lambda(x)$</td>
<td>$\frac{Q(x)}{W(x)}$</td>
</tr>
<tr>
<td>2b</td>
<td>$\deg W(x) \leq \frac{d-l-1}{2}$</td>
<td>$\deg W(x) \leq \frac{d-l-1}{2}$</td>
<td>$M(x) = \frac{P(x)}{W(x)}$</td>
</tr>
<tr>
<td>3</td>
<td>$M(x) = \frac{P(x)}{W(x)}$</td>
<td>$M(x) = \frac{Q(x)}{W(x)}$</td>
<td>$\max \deg W(x)$</td>
</tr>
</tbody>
</table>

Complexity:
- Gao’s algorithm: $O(n^2)$
- Truong’s algorithm: $O(n^2)$
- Suggested algorithm: $O(n(\log n)^2)$