CSE 533: Error-Correcting Codes **(Autumn 2006)**

Lecture 11: Decoding Algorithms for Justesen Code

11/3/2006

Lecturer: Venkatesan Guruswami *Scribe: Dang-Trinh Huynh-Ngoc*

In the first part of this lecture we continue to discuss polynomial-time decoding algorithms for Reed-Solomon (RS) code. In the second part, we will talk about polynomial-time decoding algorithms for Justesen code.

1 Error & Erasure Decoding of RS Code

For an RS $[n, k, d = n - k + 1]_q$ code, we have shown a decoding algorithm that can correct from $s < d$ erasures and [WB] decoding algorithm that can correct up to $e < d/2$ errors. Another interesting case is when the channel noise consists of both erasures and errors.

Theorem 1.1. *Given an RS $[n, k, d = n - k + 1]_q$ code, one can decode in polynomial time from a combination of any e errors and s erasures, provided that $2e + s < d$.*

Proof. Let y be the received codeword, and $\bar{S}$ be the unerased positions in y. With notice that the code obtained from the given RS code projected on $\bar{S}$ is an RS $[n - s, k, n - s - k + 1]_q$ code, we can correct e errors in y by running [WB] error decoding algorithm. This works because $2e < d - s = n - k - s + 1$. Then we correct the remaining s erasures in y by running the erasure decoding algorithm. □

2 Decoding Algorithms for Justesen Code

This section addresses the issue of error decoding for Justesen code, which is an explicit binary asymptotically good code. The question is up to how much fraction of worst-case errors (which is the codeword lengths) can we correct? Can it be $\delta/2$, where δ is the relative distance of the code, which is the best we can hope for?

This section consists of two parts. The first part gives an algorithm to decode Justesen code from up to $0 < \rho < \delta/4$ fraction of worst-case errors. The second part will improve the algorithm to achieve $\rho < \delta/2$.

First of all is some notation. We consider a Justesen code which is a concatenated code of an outer RS code C of length n over $\mathbb{F}_{2^m}$ and a family of inner binary linear codes. For the sake of discussion, we assume all the inner codes are the same and refer to them as C_{in}, but the following discussion can be generalized to the case of different inner codes. Let R, D and r, d be the rates and distances of C and C_{in}, respectively. We denote the codewords in C as $(f(\alpha_0), f(\alpha_1), \ldots, f(\alpha_{n-1}))$ and codewords in the Justesen code $C \diamond C_{in}$ as

$$(C_{in}(f(\alpha_0)), C_{in}(f(\alpha_1)), \ldots, C_{in}(f(\alpha_{n-1}))).$$

Finnaly, let Z be the received codeword. We partition Z into n blocks of length m/r: $Z = z_0 \ \ldots \ z_{n-1}$.

2.1 Decoding from up to $< Dd/4$ errors

Since Justesen code is the concatenation code of an RS code and linear binary codes with known decoding algorithms, a natural decoding algorithm for Justesen code would be one that reverses the process of concatenation. Below we precisely describe the algorithm.

Algorithm

1. Decode each z_i to $y_i \in \mathbb{F}_{2^m}$ where y_i minimizes $\Delta(C_{in}(y_i), z_i)$, where ties are broken arbitrarily.
   This step takes $O(n2^m) = O(n^2)$ time.

2. Decode $\vec{y} = (y_0, \ldots, y_{n-1}) \in \mathbb{F}_{2^m}^n$ using [WB] decoding algorithms for RS code.

Theorem 2.1. *The algorithm corrects any error pattern of weight less than $Dd/4$.*

Proof. Suppose $(f(\alpha_0), \ldots, f(\alpha_{n-1}))$ was the actual message sent. Decoding fails only if $y_i \neq f(\alpha_i)$ for at least $D/2$ values of i. For each such i, since $y_i \neq f(\alpha_i)$, we must have $\Delta(C_{in}(f(\alpha_i)), z_i) \geq d/2$. Thus the decoding fails only if there are more than $Dd/4$ errors. □

2.2 Decoding from up to half the distance $< Dd/2$ errors

We now improve the "natural algorithm" in the last part so that we can correct more errors. By a closer look, we see that Step 1 of the algorithm can give more information to Step 2 besides $\vec{y}$. The closer $C_{in}(y_i)$ is to z_i, the more certain we have about the correctness of y_i in the decoded codeword. It would be better if we can somehow make use of this information. This gives rise to the following algorithm due to [Forney '66]. One step in the algorithm is randomized, but we will derandomize it later.

Generalized Minimum Distance (GMD) Decoding Algorithm

1. Decode z_i to y_i as before.
   Set $w_i = \min(\Delta(C_{in}(y_i), z_i), d/2)$.
   Let $\vec{y} = (y_0, \ldots, y_{n-1})$.

2. With probability $\frac{2w_i}{d}$ declare y_i to be an erasure (set $y_i =?$).

3. Decode (now with erasure) $\vec{y}$ using the error & erasure decoding algorithm for RS code.

The following theorem characterizes $\vec{y}$ so that it can be input to Step 3 in the algorithm.

Theorem 2.2. $\mathbb{E}[2\bar{e} + \bar{s}] < D$, *where $\bar{e}$ and $\bar{s}$ are the number of errors and erasures in $\vec{y}$, respectively.*

Proof. Let Y_i^{err} and Y_i^{ers} be indicators for the event of error and erasure for y_i. It suffices that we prove $\mathbb{E}[2Y_i^{err} + Y_i^{ers}] \leq \frac{2e_i}{d}$, where $e_i = \Delta(C_{in}(f(\alpha_i)), z_i)$ is the number of errors in z_i and hence $\sum e_i$ is the weight of error pattern, which is less than $Dd/2$.

There are two cases:

- Case 1: $f(\alpha_i) = y_i$. Then there is no error at position i, hence $\mathbb{E}[Y_i^{err}] = \Pr[Y_i^{err} = 1] = 0$. Also, since $w_i \leq \Delta(C_{in}(y_i), z_i) = \Delta(C_{in}(f(\alpha_i)), z_i) = e_i$, we have $\mathbb{E}[Y_i^{ers}] = \frac{2w_i}{d} \leq \frac{2e_i}{d}$.

- Case 2: $f(\alpha_i) \neq y_i$. Then there is an error at position i. We have $\mathbb{E}[Y_i^{ers}] = \Pr[Y_i^{ers} = 1] = \frac{2w_i}{d}$, and $\mathbb{E}[Y_i^{err}] = \Pr[Y_i^{err} = 1] = 1 - \Pr[Y_i^{ers} = 1] = 1 - \frac{2w_i}{d}$. Then $\mathbb{E}[2Y_i^{err} + Y_i^{ers}] = 2 - \frac{2w_i}{d} \leq \frac{2e_i}{d}$. The last inequality comes from $w_i + e_i \geq d$, and this comes from

  * If $w_i = \Delta(C_{in}(y_i), z_i)$, then since $e_i = \Delta(C_{in}(f(\alpha_i)), z_i)$, by triangle inequality, $w_i + e_i \geq \Delta(C_{in}(y_i), C_{in}(f(\alpha_i))) \geq d$.
  * If $w_i = d/2$ and $w_i \leq \Delta(C_{in}(y_i), z_i)$, then by the way we choose y_i at Step 1 in the algorithm, $\Delta(C_{in}(y_i), z_i) \leq \Delta(C_{in}(f(\alpha_i)), z_i) = e_i$. Thus $e_i \geq d/2$, therefore $w_i + e_i \geq d$.

□

Next we show how to derandomize Step 2. Since the above theorem does not require that the events of two different y_i's to be declared as erasures to be independent, we can replace Step 2 with the following new step

2'. Pick θ uniformly in $[0, 1]$.
For each i, if $\theta < \frac{2w_i}{d}$, then declare y_i as erasure.

It is easy to see that each y_i is still declared as erasure with probability $\frac{2w_i}{d}$. Let $Q = \{0, 1\} \cup \{\frac{2w_0}{d}, \ldots, \frac{2w_{n-1}}{d}\}$. Since each w_i is an integer, it is easy to see that $|Q| \leq d + 1$. Denote the element in Q by q_i for $1 \leq i \leq |Q|$, where $q_i < q_j$ for any $i < j$. Then we can see that every $\theta \in [q_i, q_{i+1})$ in Step 2' produces the same $\vec{y}$. So now we are ready to derandomize the algorithm.

Deterministic GMD Algorithm

1. As before.
   Set $Q = \{0, 1\} \cup \{\frac{2w_0}{d}, \ldots, \frac{2w_{n-1}}{d}\}$.
2. For each $\theta \in Q$:
   (a) For each i, if $\theta < \frac{2w_i}{d}$, then declare y_i as erasure.
   (b) Decode $\vec{y}$ using the error & erasure decoding algorithm for RS code.
   Let J_θ be the output codeword (if any).
3. Output the J_θ that minimizes $\Delta(J_\theta, Z)$.

(To be continue in the next lecture...)