WIKIPEDIA
The Free Encyclopedia

Berlekamp–Welch algorithm

The **Berlekamp–Welch algorithm**, also known as the **Welch–Berlekamp algorithm**, is named for Elwyn R. Berlekamp and Lloyd R. Welch. This is a decoder algorithm that efficiently corrects errors in Reed–Solomon codes for an RS(n, k), code based on the Reed Solomon original view where a message $m_1, \cdots, m_k$ is used as coefficients of a polynomial $F(a_i)$ or used with Lagrange interpolation to generate the polynomial $F(a_i)$ of degree < k for inputs $a_1, \cdots, a_k$ and then $F(a_i)$ is applied to $a_{k+1}, \cdots, a_n$ to create an encoded codeword $c_1, \cdots, c_n$.

The goal of the decoder is to recover the original encoding polynomial $F(a_i)$, using the known inputs $a_1, \cdots, a_n$ and received codeword $b_1, \cdots, b_n$ with possible errors. It also computes an error polynomial $E(a_i)$ where $E(a_i) = 0$ corresponding to errors in the received codeword.

The key equations

Defining e = number of errors, the key set of n equations is

$$b_i E(a_i) = E(a_i) F(a_i)$$

Where E(a_i) = 0 for the e cases when $b_i \neq F(a_i)$, and $E(a_i) \neq 0$ for the n - e non error cases where b_i = F(a_i) . These equations can't be solved directly, but by defining Q() as the product of E() and F():

$$Q(a_i) = E(a_i) F(a_i)$$

and adding the constraint that the most significant coefficient of $E(a_i)$ = e_e = 1, the result will lead to a set of equations that can be solved with linear algebra.

$$b_i E(a_i) = Q(a_i)$$
$$b_i E(a_i) - Q(a_i) = 0$$
$$b_i(e_0 + e_1 a_i + e_2 a_i^2 + \cdots + e_e a_i^e) - (q_0 + q_1 a_i + q_2 a_i^2 + \cdots + q_q a_i^q) = 0$$

where q = n - e - 1. Since e_e is constrained to be 1, the equations become:

$$b_i(e_0 + e_1 a_i + e_2 a_i^2 + \cdots + e_{e-1} a_i^{e-1}) - (q_0 + q_1 a_i + q_2 a_i^2 + \cdots + q_q a_i^q) = -b_i a_i^e$$

resulting in a set of equations which can be solved using linear algebra, with time complexity $O(n^3)$.

The algorithm begins assuming the maximum number of errors e = $\lfloor(n\text{-}k)/2\rfloor$. If the equations can not be solved (due to redundancy), e is reduced by 1 and the process repeated, until the equations can be solved or e is reduced to 0, indicating no errors. If Q()/E() has remainder = 0, then F() = Q()/E() and the code word values F(a_i) are calculated for the locations where E(a_i) = 0 to recover the original code word. If the remainder ≠ 0, then an uncorrectable error has been detected.

Example

Consider RS(7,3) (n = 7, k = 3) defined in $GF(7)$ with α = 3 and input values: a_i = i-1 : {0,1,2,3,4,5,6}. The message to be systematically encoded is {1,6,3}. Using Lagrange interpolation, $F(a_i)$ = 3 x^2 + 2 x + 1, and applying $F(a_i)$ for a_4 = 3 to a_7 = 6, results in the code word {1,6,3,6,1,2,2}. Assume errors occur at c_2 and c_5 resulting in the received code word {1,5,3,6,3,2,2}. Start off with e = 2 and solve the linear equations:

$$\begin{bmatrix} b_1 & b_1 a_1 & -1 & -a_1 & -a_1^2 & -a_1^3 & -a_1^4 \\ b_2 & b_2 a_2 & -1 & -a_2 & -a_2^2 & -a_2^3 & -a_2^4 \\ b_3 & b_3 a_3 & -1 & -a_3 & -a_3^2 & -a_3^3 & -a_3^4 \\ b_4 & b_4 a_4 & -1 & -a_4 & -a_4^2 & -a_4^3 & -a_4^4 \\ b_5 & b_5 a_5 & -1 & -a_5 & -a_5^2 & -a_5^3 & -a_5^4 \\ b_6 & b_6 a_6 & -1 & -a_6 & -a_6^2 & -a_6^3 & -a_6^4 \\ b_7 & b_7 a_7 & -1 & -a_7 & -a_7^2 & -a_7^3 & -a_7^4 \end{bmatrix} \begin{bmatrix} e_0 \\ e_1 \\ q0 \\ q1 \\ q2 \\ q3 \\ q4 \end{bmatrix} = \begin{bmatrix} -b_1 a_1^2 \\ -b_2 a_2^2 \\ -b_3 a_3^2 \\ -b_4 a_4^2 \\ -b_5 a_5^2 \\ -b_6 a_6^2 \\ -b_7 a_7^2 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 6 & 0 & 0 & 0 & 0 \\ 5 & 5 & 6 & 6 & 6 & 6 & 6 \\ 3 & 6 & 6 & 5 & 3 & 6 & 5 \\ 6 & 4 & 6 & 4 & 5 & 1 & 3 \\ 3 & 5 & 6 & 3 & 5 & 6 & 3 \\ 2 & 3 & 6 & 2 & 3 & 1 & 5 \\ 2 & 5 & 6 & 1 & 6 & 1 & 6 \end{bmatrix} \begin{bmatrix} e_0 \\ e_1 \\ q0 \\ q1 \\ q2 \\ q3 \\ q4 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 2 \\ 2 \\ 1 \\ 6 \\ 5 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e_0 \\ e_1 \\ q0 \\ q1 \\ q2 \\ q3 \\ q4 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \\ 4 \\ 3 \\ 3 \\ 1 \\ 3 \end{bmatrix}$$

Starting from the bottom of the right matrix, and the constraint e_2 = 1:

$$Q(a_i) = 3x^4 + 1x^3 + 3x^2 + 3x + 4$$

$$E(a_i) = 1x^2 + 2x + 4$$

$F(a_i) = Q(a_i)/E(a_i) = 3x^2 + 2x + 1$ with remainder = 0.

E(a_i) = 0 at a_2 = 1 and a_5 = 4 Calculate F(a_2 = 1) = 6 and F(a_5 = 4) = 1 to produce corrected code word {1,6,3,6,1,2,2}.

See also

- Reed–Solomon error correction

External links

- MIT Lecture Notes on Essential Coding Theory – Dr. Madhu Sudan (http://people.csail.mit.edu/madhu/FT02/)
- University at Buffalo Lecture Notes on Coding Theory – Dr. Atri Rudra (https://web.archive.org/web/20110606191907/http://www.cse.buffalo.edu/~atri/courses/coding-theory/fall07.html)
- Algebraic Codes on Lines, Planes and Curves, An Engineering Approach – Richard E. Blahut
- Welch Berlekamp Decoding of Reed–Solomon Codes – L. R. Welch
- US 4,633,470 (https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=US4,633,470), Welch, Lloyd R. & Berlekamp, Elwyn R., "Error Correction for Algebraic Block Codes", published September 27, 1983, issued December 30, 1986 – The patent by Lloyd R. Welch and Elewyn R. Berlekamp