Chien search

From Wikipedia, the free encyclopedia

In abstract algebra, the **Chien search**, named after Robert T. Chien, is a fast algorithm for determining roots of polynomials defined over a finite field. The most typical use of the Chien search is in finding the roots of error-locator polynomials encountered in decoding Reed-Solomon codes and BCH codes.

Algorithm [edit]

We denote the polynomial (over the finite field GF(q)) whose roots we wish to determine as:

$$\Lambda(x) = \lambda_0 + \lambda_1 x + \lambda_2 x^2 + \cdots + \lambda_t x^t$$

Conceptually, we may evaluate $\Lambda(\beta)$ for each non-zero β in GF(q). Those resulting in 0 are roots of the polynomial.

The Chien search is based on two observations:

- Each non-zero β may be expressed as α^{i_β} for some i_β, where α is a primitive element of $\mathrm{GF}(q)$, i_β is the power number of primitive element α. Thus the powers α^i for $0 \leq i < (q-1)$ cover the entire field (excluding the zero element).
- The following relationship exists:

$$\begin{array}{lllllllllll}
\Lambda(\alpha^i) & = & \lambda_0 & + & \lambda_1(\alpha^i) & + & \lambda_2(\alpha^i)^2 & + & \cdots & + & \lambda_t(\alpha^i)^t \\
& \triangleq & \gamma_{0,i} & + & \gamma_{1,i} & + & \gamma_{2,i} & + & \cdots & + & \gamma_{t,i} \\
\Lambda(\alpha^{i+1}) & = & \lambda_0 & + & \lambda_1(\alpha^{i+1}) & + & \lambda_2(\alpha^{i+1})^2 & + & \cdots & + & \lambda_t(\alpha^{i+1})^t \\
& = & \lambda_0 & + & \lambda_1(\alpha^i)\,\alpha & + & \lambda_2(\alpha^i)^2\,\alpha^2 & + & \cdots & + & \lambda_t(\alpha^i)^t\,\alpha^t \\
& = & \gamma_{0,i} & + & \gamma_{1,i}\,\alpha & + & \gamma_{2,i}\,\alpha^2 & + & \cdots & + & \gamma_{t,i}\,\alpha^t \\
& \triangleq & \gamma_{0,i+1} & + & \gamma_{1,i+1} & + & \gamma_{2,i+1} & + & \cdots & + & \gamma_{t,i+1}
\end{array}$$

In other words, we may define each $\Lambda(\alpha^i)$ as the sum of a set of terms $\{\gamma_{j,i} | 0 \leq j \leq t\}$, from which the next set of coefficients may be derived thus:

$$\gamma_{j,i+1} = \gamma_{j,i}\,\alpha^j$$

In this way, we may start at $i = 0$ with $\gamma_{j,0} = \lambda_j$, and iterate through each value of i up to $(q-1)$. If at any stage the resultant summation is zero, i.e.

$$\sum_{j=0}^{t} \gamma_{j,i} = 0,$$

then $\Lambda(\alpha^i) = 0$ also, so α_i is a root. In this way, we check every element in the field.

When implemented in hardware, this approach significantly reduces the complexity, as all multiplications consist of one variable and one constant, rather than two variables as in the brute-force approach.

References [edit]

- Chien, R. T. (October 1964), "Cyclic Decoding Procedures for the Bose-Chaudhuri-Hocquenghem Codes", *IEEE Transactions on Information Theory*, IT-10 (4): 357–363, doi:10.1109/TIT.1964.1053699, ISSN 0018-9448
- Lin, Shu; Costello, Daniel J. (2004), *Error Control Coding: Fundamentals and Applications* (second ed.), Englewood Cliffs, NJ: Prentice-Hall, ISBN 978-0130426727
- Gill, John, *EE387 Notes #7, Handout #28* (PDF), Stanford University, pp. 42–45, retrieved April 21, 2010

Categories: Error detection and correction | Finite fields