The hybrid method for finding the roots of a polynomial over finite fields based on affine expansion

Pham Khac Hoan[1], Nguyen Tien Thai [1], Lai Tien De[1], Vu Son Ha[2*]

[1]Military Technical Academy;
[2]Academy of Military Science and Technology.
[*]Corresponding author: vusonha76@gmail.com
Received 03 Oct. 2023; Revised 10 Dec. 2023; Accepted 12 Dec. 2023; Published 30 Dec. 2023
DOI: https://doi.org/10.54939/1859-1043.j.mst.CSCE7.2023.71-80

ABSTRACT

This paper proposes a hybrid method to find the roots of a polynomial in a finite field based on combining the polynomial decomposition into the sum of multiples of the affine polynomial with the analytical method to find the roots of low-order polynomials. The proposed method allows a significant reduction in complexity and is applicable in the design of low-latency BCH and Reed-Solomon decoders.

Keywords: Finite field; Coding and Information theory; Affine polynomial; Polynomial basis; Normal basis; Error control coding; BCH code; Reed-Solomon code.

1. INTRODUCTION

Finite fields such as error control coding, code-based cryptography, and elliptic curve cryptography are widely used in electronic engineering and computer science. Some problems are associated with solving equations in finite fields, such as solving key equations when decoding BCH, Reed-Solomon, and decoding Goppa with McEliece cryptosystems [1].

Some indirect methods to solve equations in finite fields include implementing iterative algorithms such as the Chien procedure or using the Fourier transform in Galois fields [2-4]. However, these methods have relatively significant computational delays and high complexity. On the other hand, in many communication systems such as SONET optical communication network, WiMax, and DVB-S2 digital television standard, BCH, and Reed-Solomon codes are often used with the number of errors being corrected not too large. Still, the code length is relatively remarkable to compromise between encoding speed and complexity. Therefore, in this case, when designing the encoder and decoder, it is a question of solving the equation on a finite field of large size with high requirements for throughput and processing delay [5-8]. The works in [9-14] have studied the analytic formula for finding roots, the transformation method, or using the lookup table to find the roots of low-order polynomials in finite fields.

Linearization polynomials and affine polynomials have several remarkable properties that allow them to simplify their root or value finding at points in a finite field. The article [15] studies the method of finding a polynomial's roots based on the affine polynomial's roots, which is the multiple of the polynomial of interest. However, only a few polynomials can be linearized as affine polynomials. Based on algebraic transformations and finding an affine polynomial that is a multiple of a given polynomial, it is possible to find this polynomial's roots among the affine polynomial's roots. The implementation complexity increases significantly when the polynomial degree is large (more than 10) [16]. With such a high-order polynomial, the most

suitable method is the Chien search procedure, but as the field size increases, many challenges arise. Based on the properties of the linearized polynomial, it is possible to calculate its value at all the elements of the finite field; on that basis, the improved Chien procedure is built.

This paper investigates a hybrid method to find the roots of a polynomial over the finite field GF(2^m) combining algebraic transformation, indirectly finding roots through finding roots of an affine polynomial that is a multiple of the given polynomial and improved Chien procedure. The paper's results form the basis for building high-speed error correction decoding devices in information transmission and storage systems. The results obtained allow for applying adaptively in different cases and can be extended to fields of any size.

The rest of the paper is organized as follows. Section 2 studies a special class of linear polynomials in a finite field and the problem of finding their roots. Section 3 proposes a hybrid method to find the roots of high-order polynomials in a finite field. Section 4 evaluates the quality of the proposed method. Finally, some conclusions are drawn in section 5.

2. LINEARIZED POLYNOMIALS AND ROOTS OF LINEARIZED POLYNOMIALS

The Chien procedure and the analytical method for finding the roots of polynomials in a finite field have a rather significant complexity when the field size is large, especially when the degree of the equation is high. A special class of polynomials exists called linear polynomials, whose roots are simpler to find. This section considers linearization polynomials and affine polynomials and finds the roots of affine polynomials.

2.1. Linearized polynomials and affine polynomials

A polynomial $L(z)$ in the field $GF(p^m)$ will be a linearized polynomial if it has the form:

$$L(z)=\sum_i L_i z^{p^i} = L_0 z + L_1 z^p + ... + L_{m-1} z^{p^{m-1}} + z^{p^m} \quad (1)$$

For example, with the element $c \in GF(p^m)$, the trace function of c is defined:

$$Tr(c)=c+c^p+c^{p^2}+...+c^{p^{m-1}} \quad (2)$$

Obviously, the trace function is a linearized polynomial.

Note that the author in [1] proved that assuming the roots $L(z)$ are in the extended field $GF(p^h)$, $h>m$, these roots form a sub-vector space in $GF(p^m)$.

A polynomial $A(z)$ in $GF(p^m)$ is called an affine polynomial if $A(z)=L(z)-u$, where $L(z)$ is a linear polynomial and $u \in GF(p^m)$.

Basic properties of linearized polynomials, affine polynomials [1].

1. With a linearized polynomial, we have: $L(z_1+z_2)=L(z_1)+L(z_2)$ where $z_1, z_2 \in GF(p^m)$.

2. A polynomial $L(z)$ is a linearized polynomial if and only if its roots form a linear space in $GF(p)$ and the roots have the same multiple and are powers of p.

3. A polynomial $A(z)$ is an affine polynomial if and only if its roots form an affine space in $GF(p)$ and the roots have the same multiple and are powers of p.

4. The greatest common divisor of two linearized polynomials is a linearized polynomial.

5. The greatest common divisor of two affine polynomials is an affine polynomial.

2.2. Roots of linearized polynomials

Given that $L(z)$ is a linearized polynomial in $GF(p^m)$, described by formula (1), assume that γ is a primitive root, $L(z)$ is a polynomial with degree p^m, the roots of this polynomial have the form:

$$\delta_0\gamma + \delta_1\gamma^p + ... + \delta_{n-1}\gamma^{p^{m-1}} \tag{3}$$

where $\delta_i \in GF(p)$.

Finding the roots of a linearized polynomial will be discussed in the following section.

Notations $\alpha^0, \alpha^1, ..., \alpha^{m-1}$ are the polynomial basis (standard basis) of the field $GF(p^m)$, we have the following propositions.

Proposition 1 [1].

Let $L(z)$ is a linearized polynomial in the field $GF(p^m)$, the element z is represented as $z = \sum_k z_k\alpha^k$, $z_k \in GF(p)$ then

$$L(z) = \sum_k z_k L(\alpha^k) \tag{4}$$

When using the polynomial representation of the elements: $L(\alpha^i) = \sum_{j=0}^{m-1} C_{i,j}\alpha^j$.

Therefore, the polynomial expansion coefficient of $L(z)$ on the polynomial basis is calculated as follows:

$$[L_0, L_1, ..., L_{n-1}] = [z_0, z_1, ..., z_{n-1}].C \tag{5}$$

where:

$$C = \begin{bmatrix} C_{0,0} & C_{0,1} & C_{0,1} & . & . & . & C_{0,n-1} \\ C_{1,0} & C_{1,1} & C_{1,2} & . & . & . & C_{1,n-1} \\ . & . & . & . & . & . & . \\ C_{n-1,0} & C_{n-1,1} & C_{n-1,2} & . & . & . & C_{n-1,n-1} \end{bmatrix} \tag{6}$$

The roots of the affine polynomial $A(z) = L(z) - u$ (also known as the affine equation) are the roots of the system of equations:

$$[z_0, z_1, ..., z_{m-1}].C = [u_0, u_1, ..., u_{m-1}] \tag{7}$$

where $(u_0, u_1, ..., u_{m-1})$ is the polynomial basis representation of u.

To solve the system of linear equations (7), in [16], the author has presented an effective method based on linear transformations on columns to get an idempotent triangular matrix. Note that according to the property of linear polynomials, the roots of an affine polynomial consist of an eigen root and the sum of the eigen roots and the roots of the polynomial $L(z)$.

3. HYBRID METHOD OF SOLVING EQUATIONS IN THE FINITE FIELD

3.1. Quadratic equation

In the special case of the second-order affine polynomial in $GF(2^m)$, it is possible to compute its roots without solving a system of linear equations. The quadratic equation can be reduced to the canonical form:

$$y^2 + y = u \tag{8}$$

Equation (8) has roots in $GF(2^m)$ if and only if $Tr(u)=0$.

In the field GF(2^m) exists a normal basis $\{\gamma, \gamma^2, \gamma^4, ..., \gamma^{\vartheta}\}$, where $\vartheta = 2^{m-1}$ for the generator element γ. The element on the field GF(2^m) can be represented in the normal basis form: $u = u_0\gamma + u_1\gamma^2 + ... + u_\vartheta\gamma^\vartheta$.

Then, two roots of the equation (8) are [12, 13]:

$$y' = \sum_{i=0}^{m-1} y_i \gamma^{2^i}, y'' = 1 + y' \tag{9}$$

The coefficients y_i are determined according to the formula:

$$y_0 = 0;\ \ y_1 = d_1;\ \ y_2 = d_1 + d_2; ...;\ \ y_{n-1} = \sum_{i=1}^{n-1} d_i \tag{10}$$

Assume that y_* is a root of (8), then y_*^2 is a root of $y^2 + y = u^2$. On that basis, it is possible to divide the values u (with $Tr(u)=0$) into cyclotomic classes $\{u, u^2, u^4, ..., u^{2^{n-1}}\}$.

In the field $GF(2^m)$, half of the elements have a trace of 0, and the other half have a trace of 1. By dividing the field into cyclotomic adjacent classes, it is possible to store roots more efficiently, then using formulas (9) -(10) to find the roots. When using the orbit representation based on cyclotomic adjacent classes, the number of considered elements decreases from 2^m to about m representing adjacent classes, so the amount of required memory decreases about $2^m / m$ times. Table 1 shows the orbit representation with the parameter u according to the adjacent class representation and the corresponding roots in the fields GF(2^m) with the generating polynomials $x^3 + x + 1;\ \ x^4 + x + 1;\ \ x^5 + x^2 + 1;$ $x^6 + x + 1$. Similarly, it is possible to construct orbitals for larger fields and store them in the memory used to compute the roots of quadratic canonical equations. Note that in this table, the field elements are represented by the exponent $log_\alpha(\alpha^i)$.

Table 1. *The orbit representation with the parameter u according to cyclotomic classes and roots.*

m	u	y'	y''	m	u	y'	y''
3	1	2	6	6	0	21	42
4	0	6	10		1	17	47
	1	7	9		3	14	52
	5	1	4		7	1	6
5	1	3	29		9	22	56
	7	2	5		13	30	46
	15	21	25		27	36	56

3.2. Cubic equation

Most of the polynomials on $GF(2^m)$ are not affine polynomials. Third-degree polynomials $f(z)=z^3+az^2+bz+c$ can find affine polynomials of the form:

$$A(z)=(z^3+az^2+bz+c)(z+a)=z^4+(a^2+b)z^2+(ab+c)z+ac \tag{11}$$

Among the roots of this affine polynomial, choosing one root z_1 of $f(z)$, analyzing the polynomial $f(z)=(x+z_1)\left[x^2+(a+z_1)x+c/z_1\right]$. Then finding the roots of the equation $x^2+(a+z_1)x+c/z_1=0$ thanks to the method of representing orbits based on adjacent classes.

3.3. Quartic equation

In the field $GF(2^m)$, performing the combined method of algebraic transformation to affine polynomials to find roots efficiently as follows:

Consider the quadratic equation in the field GF(2^m):

$$x^4+Ax^3+Bx^2+Cx+D=0 \tag{12}$$

Thanks to the substitution $x=y^{-1}+\sqrt{C/A},\ A\neq 0,$ it can be brought to the equation:

$$a_3y^4+a_2y^2+a_1y+a_0=0, \tag{13}$$

where

$$a_3=D+BC/A+(C/A)^2;\ \ a_2=B+\sqrt{AC},a_1=A,a_0=1. \tag{14}$$

When $a_3,\ a_2\neq 0$, substitude $y=z\sqrt{a_2/a_3}$ we have

$$z^4+z^2+E_1z+E_2=0, \tag{15}$$

where $E_1=a_1\sqrt{a_3/a_2^3};\ \ E_2=a_3/a_2^2.$

In this case, the left side of (15) is an affine polynomial, and its roots can be found by solving a system of linear equations as described in section 3.2. Once the roots are found, use inverse substitution to find the roots of the original equation. One solution that compromises latency and memory capacity is to use memory that stores the values of $L(z)=z^4+z^2+E_1z$ with each $E_1\in GF(2^m)$. Then, compare the value in this table with E_2.

3.4. Quintic equation

For a quintic equation, find roots through the roots of an affine polynomial, a multiple of the 5th degree polynomial, using the following algorithm in [1, 15].

Algorithm 1 [1]

1. Calculate $z^k \bmod f(z)$ with $k = 5,6,7,8;$

2. Based on the result in step 1, calculate the residuals $r^{(0)}(z), r^{(1)}(z), ..., r^{(i)}(z),$ with $r^{(i)}(z) = z^{p^i} \bmod f(z);$

3. Solve a system of linear equations

$$\left[u, L_0, L_1, ..., L_{d-1}\right] \begin{bmatrix} 0 & 0 & ... & 0 & -1 \\ r_{(d-1)}^{(0)} & ... & ... & r_1^{(0)} & r_0^{(0)} \\ r_{(d-1)}^{(1)} & ... & ... & r_1^{(1)} & r_0^{(1)} \\ ... & ... & ... & ... & ... \\ r_{(d-1)}^{(d-1)} & ... & ... & r_1^{(d-1)} & r_0^{(d-1)} \end{bmatrix} \tag{16}$$

Example 1. In the field GF(2^5) with $\alpha^5 + \alpha^2 + 1 = 0$ consider the polynomial $f(z) = z^5 + \alpha^{22} z^4 + \alpha^{18} z^3 + \alpha^{19} z^2 + \alpha^{16} z + \alpha^{13}$. Find an affine polynomial that is a multiple of $f(z)$. Caculate:

$r^{(i)}(z) = z^{2^i} \bmod f(z);\ r^{(0)}(z) = z^1 \bmod f(z) = z;\ r^{(1)}(z) = z^2 \bmod f(z) = z^2;$

$r^{(2)}(z) = z^4 \bmod f(z) = z^4;$

$z^8 \bmod f(z) = \alpha^3 z^4 + \alpha^{15} z^3 + \alpha^6 z^2 + \alpha^6 z + \alpha^{10};$

$z^{16} \bmod f(z) = \alpha^6 z^8 + \alpha^{30} z^6 + \alpha^{12} z^4 + \alpha^{12} z^2 + \alpha^{20} = \alpha^{29} z^4 + \alpha^{28} z^3 + \alpha^5 z^2 + \alpha^6 z + \alpha^{15};$

We have a system of equations:

$$\left[U\ L_0\ L_1\ L_2\ L_3\ L_4\right] \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ \alpha^3 & \alpha^{15} & \alpha^6 & \alpha^6 & \alpha^{10} \\ \alpha^{29} & \alpha^{28} & \alpha^5 & \alpha^6 & \alpha^{15} \end{bmatrix}.$$

Solve this system of equations to find the root

$$\left[U\ L_0\ L_1\ L_2\ L_3\ L_4\right] = \left[\alpha^4\ \alpha^{20}\ \alpha^{18}\ \alpha^{30}\ \alpha^{13}\ 1\right]$$

Then, the polynomial $L(z) = \alpha^{20} z + \alpha^{18} z^2 + \alpha^{30} z^4 + \alpha^{13} z^8 + z^{16}$.

We can find the roots of the equation $L(z) = \alpha^4$ thanks to solving the system of equations presented in section 3.2 and get eight roots in which the roots in GF(2^5) – the roots of $f(z)$ are $\alpha^6, \alpha^8, \alpha^{17}$.

Total complexity when solving quintic equations by the affine multiple method in field $GF(2^m)$ is $m^2 + 11m + 206$ [13].

3.5. Affine expansion and the problem of finding roots of high-order polynomials over finite fields

With polynomials of degree 6 or higher, directly transforming or finding an affine polynomial that is a multiple of the given polynomial becomes difficult because solving additional equations when transforming or the multiple affine polynomials has too large degrees is necessary. To deal with that problem, an improved Chien procedure based on the affine expansion described below is proposed. Using the property of linear polynomials (formula (4)), it is possible to calculate $L_i(z)$ at the elements of a finite field by precomputing the values at the polynomial bases of the field $L_i(\alpha^j)$. It is, therefore, possible to determine or calculate linearized polynomials at each point of the finite field $GF(2^m)$ using very small and previously calculated tables.

The polynomial $f(z)=\sum_{k=0}^{d} f_k z^k$ can be decomposed to the sum of polynomials that are multiples of affine polynomials as follows:

$$f(z)=f_3z^3+\left\{A_0(z)+z^5\left[A_1(z)+z^5\left(A_2(z)+z^5\left(A_3(z)+...\right)\right)\right]\right\} \tag{17}$$

where: $A_i(z)=f_{5k}+L_i(z),\ L_i(z)=\sum_{s=0}^{3} f_{5k+2^s}z^{2^s}$.

Polynomials of degree no more than eight can perform the affine expansion as follows:

$$\begin{aligned} f(z)&=\sum_{k=0}^{8} f_k z^k = A_1(z)+z^3\left(A_2(z)+f_6z^3\right)\\ A_1(z)&=f_0+L_1(z)=f_0+f_1z+f_2z^2+f_4z^4+f_8z^8\\ A_2(z)&=f_3+L_2(z)=f_3+f_5z^2+f_7z^4 \end{aligned} \tag{18}$$

Polynomials of degree no more than 17 can perform the affine expansion as follows:

$$\begin{aligned} f(z)&=A_1(z)+z^3\left\{A_2(z)+z^3\left[f_6+z^3\left(A_3(z)+z^3\left(A_4(z)+f_{15}z^3\right)\right)\right]\right\}\\ A_1(z)&=f_0+L_1(z)=f_0+f_1z+f_2z^2+f_4z^4+f_8z^8+f_{16}z^{16}\\ A_2(z)&=f_3+L_2(z)=f_3+f_5z^2+f_7z^4\\ A_3(z)&=f_9+L_3(z)=f_9+f_{10}z+f_{11}z^2+f_{13}z^4\\ A_4(z)&=f_{12}+f_{14}z^2 \end{aligned} \tag{19}$$

To calculate the values of the affine polynomials at all the elements of the field when finding the roots of $f(z)$ using the special property of linearized polynomials, first of all, the elements of the field $GF(2^m)$ are sorted by Gray code, denoted by g_j with $g_0=0,\ j=0,1,...2^m-1$. Example of a Gray code of the elements of the field GF(2^3) with generative polynomial x^3+x+1 are 000, 001, 011, 010, 110, 111, 101, 100 corresponding to 0; 1; $\alpha^3,\alpha,\alpha^4,\alpha^5,\alpha^6,\alpha^2$. We have:

$$A(g_j)=A(g_{j-1})+L(g_j+g_{j-1})=A(g_{j-1})+L(\alpha^{\delta(g_j,g_{j-1})}) \tag{20}$$

where: $\delta(g_j, g_{j-1})$ indicates the position in which g_j differs from g_{j-1} in its vector representation.

The above formula allows for constructing an improved Chien procedure based on calculating the affine polynomial's value at all point of the finite field. The improved Chien procedure based on the affine expansion consists of the following steps:

1. Calculate the values $L_k(\alpha^i)$ with $i = 0,1,...,m-1$ of linear polynomials in the affine expansion.

2. Sort the elements of the field $GF(2^m)$ according to the Gray code g_j and calculate the value $A(g_j)$ according to the retrieval formula (20).

3. Calculate the values $f(g_j)$ by affine decomposition (17)-(19). If $f(g_j) = 0$ then g_j is a root. If not, then check the next element g_{j+1} by repeating steps 2 and 3.

4. ANALYSIS OF THE COMPLEXITY OF THE PROPOSED METHOD

To evaluate the complexity of the proposed algorithm, we pay attention to some of the following characteristics.

The methods' device-time complexity is estimated with multiplication, square, and addition operations. The implementation complexity also depends on the choice of the irreducible polynomial and the basis used to represent the finite field elements. With affine decomposition, the multiplications are significantly reduced and replaced by less complex squares.

Complexity when using the Chien procedure

$$W_{Ch} = (C_{add} + C_{mul})d(2^m - 1). \tag{21}$$

An affine expansion of (18) at each point of the field requires nine multiplications, four squares, and nine additions. Applying the recursive formula (20) to each element will need more $2^m - 1$ additions and m calculations $L_1(\alpha^i), L_2(\alpha^i)$ where $i = 0,1,...,m-1$. The simultaneous calculation process $L_1(\alpha^i), L_2(\alpha^i)$ requires six multiplications, three squares, and three additions. So the total complexity using the improved Chien procedure by affine expansion (18) is

$$\begin{aligned} W_8 &= (9C_{mul} + 4C_{sq} + 9C_{add}) + (2^m - 1)C_{add} + m(6C_{mul} + 3C_{sq} + 3C_{add}) \\ &= (6m + 9)C_{mul} + (3m + 4)C_{sq} + (2^m + 3m + 8)C_{add}, \end{aligned} \tag{22}$$

where C_{mul}, C_{sq}, C_{add} is the complexity of the multiplier, squarer, and adder in the finite field, respectively.

Note that with the most common implementation using a polynomial basis, the complexity of multiplication is about $m^2/2$ XOR circuit, while square has a complexity of about 2*m* OR circuit, the complexity of addition is *m* XOR circuit.

When solving 8th degree equations in the field GF(2^{10}) ($m = 10, d = 8$) using the proposed method, the device complexity is reduced by about 5.5 times. The processing gain increases when m increases as the number of multiplications (highest complexity) decreases by about $d(2^m - 1)/6m$ times.

5. CONCLUSIONS

The proposed method of solving quadratic canonical equations by orbit representation based on cyclotomic adjacent classes reduces the storage capacity of lookup tables by $2^m / m$ times. At the same time, the article proposes a method to solve cubic and quartic equations in a finite field directly by transforming them into affine equations. The roots of the quintic polynomial are found through the roots of the affine polynomial, which is a multiple of the given polynomial. For polynomials of degree six or higher, the affine expansion and an iterative procedure are built to calculate the value of a polynomial with the elements of a finite field when the polynomial basis is sorted in Gray code form. Compared with the conventional method using the Chien search procedure by trying all the elements of the field when solving the 8^{th} degree equation in the field GF(2^{10}), the complexity is reduced by about 5.5 times. Therefore, the hybrid method of finding the roots of the equation in the finite field allows the construction of high-speed and very low-latency devices, allowing utilization in high-speed information systems, error correction circuits in chip design, memory, and especially in cryptosystems based on coding.

REFERENCES

[1]. Elwyn R. Berlekamp, "*Algebraic Coding Theory (Revised Edition),"* World Scientific Publishing Co. Pte. Ltd. (2015).

[2]. F. J. M. Williams, N. J. A. Sloane, "*The theory of error correction codes,"* Elselvier (1977).

[3]. Tood K. Moon, "*Error correction coding: Mathematical methods and algorithms,"* John & sons, Inc. (2005).

[4]. K. Deerganghao, "*Channel coding technique for wireless communications,"* Springer, India (2015).

[5]. D. Strukov, *"The area and latency tradeoffs of binary bit-parallel BCH decoders for prospective nano electronic memories"*, ACSSC Papers, (2007).

[6]. X. Zhang, Z. Wang, *"A low complexity three error correcting for optical transport network"*, IEEE Transaction on circuit and systems, Vol. 59, Issue 10, (2012).

[7]. D. Strukov, "*The area and latency tradeoffs of binary bit-parallel BCH decoders for prospective nano electronic memories*," ACSSC Papers, (2007).

[8]. J. Fredenberger, "*A configurable Bose Chauhuri Hocquenghem codec architecture for flash controller applications*," Journal of circuits, systems and computer, Vol. 23, No. 02, (2013).

[9]. K. Huber et al, *"Solving Equations in Finite Fields and Some Results Concerning the Structure of GF(p^m)"*, IEEE Trans. on Information Theory 38(3):1154-1162, (1992).

[10]. K. P. Yiu, "*On the root computation of polynomial over a finite field using a stored table approach*", *Proceeding of the IEEE,* Vol.71, No.4, (1983).

[11]. Муттер ."*Основы помехоустойчивой телепередачи информации*", Л.: Энергоатомиздат, Ленинградское отделение, (1990).

[12]. J. Fredenberger, B. N. Bailon, M. Shafies, "*Reduced complexity hard and soft decoding of BCH codes with application in concatenated codes*," EIT circuit, devices and systems, pp. 284-296, (2021).

[13]. B. Bhaskar, H. Vincent, "*Efficient root finding of polynomilas over fields of characteristic 2*", Hal, 00626997, (2009).

[14]. Hoan Pham Khac, Thai Ha Tran, Son Ha Vu, "*An algeraic transformation method to solving equations in the extened alois field*", Journal of science and technique, Vol. 17, No. 4, (2022).

[15]. Pham Khac Hoan, Nguyen Tien Thai, Vu Son Ha, "*Low latency BCH decoder using the*

affine polynomial over the finite field", Journal of Military Science and Technology, Special issue No 6, (2022).

[16]. Fedorenko S. V., Trifonov P. V. "*Finding roots of polynomials over finite fields, IEEE Transactions on Communications*", Vol. 50, Issue 11, (2002).

TÓM TẮT

Phương pháp lai tìm nghiệm của đa thức trên trường hữu hạn dựa trên khai triển affine

Bài báo đề xuất phương pháp lai tìm nghiệm của đa thức trên trường hữu hạn dựa trên việc kết hợp phân rã đa thức thành tổng của các bội đa thức affine với phương pháp giải tích tìm nghiệm của các đa thức bậc thấp. Phương pháp đã đề xuất cho phép ứng dụng trong thiết kế bộ giải mã mã BCH, Reed-Solomon có độ phức tạp và độ trễ thấp.

Từ khoá: Trường hữu hạn; Lý thuyết thông tin và mã hoá; Mã hóa sửa lỗi; Đa thức affine; Cơ sở đa thức; Cơ sở chuẩn hoá; Mã BCH; Mã Reed-Solomon.