

# Efficient root finding of polynomials over fields of characteristic 2.

Vincent HERBERT

INRIA Paris - Rocquencourt

**Abstract.** We improve the technique for finding roots of univariate algebraic equations over a finite field of characteristic 2. We evaluate the theoretical complexity in worst case of the existing algorithms and determine the best trade-off between the Berlekamp trace algorithm [Ber71] and the algorithms of Zinoviev [Zin96] (see below), depending on system parameters.

Root finding of polynomials over finite fields is a classical algebraic algorithmic problem. Several approaches like Berlekamp [Ber68] and Chien [CCO69] have been made towards the solution of this problem which possesses applications in theoretical computer science : in algebraic coding theory and in code-based cryptography. The proposed solutions are applicable for decoding binary Goppa codes and BCH codes. Decoding of code-based public-key cryptosystems employs an algebraic decoding algorithm which is often broken up in three parts : syndrome computation, solve the key equation with the extended Euclidean algorithm or Berlekamp Massey algorithm and the root finding of error locator polynomial. This last stage is the most time consuming for cryptographic parameters. It consists of computing  $t$  roots of a polynomial  $S$  which splits over  $\mathbb{F}_{2^m}$ , an extension field of degree  $m$  over the two-element field  $\mathbb{F}_2$ . These roots provide location of errors. Enhancing the root finding algorithms enables to speed up the decoding process. Since McEliece-type cryptosystems are based on binary Goppa codes, our work helps to decrease significantly the decryption time of a ciphertext. As an example, for recommended values of parameters, given in [BS08], we have to compute 32 distinct roots of a polynomial of degree 32 with coefficients in  $\mathbb{F}_{2^{11}}$ .

In conventional applications of codes, fields have smaller size and Chien search is often considered as a fitted trade-off. For decryption purpose, use of more sophisticated techniques enables a meaningful benefit [BS08]. This work is based on the algorithms proposed by Zinoviev in [Zin96], as well as the Berlekamp trace algorithm (abbreviated BTA) [Ber71]. Zinoviev computes a multiple affine polynomial in order to find the roots of polynomials of degree  $\leq 10$  over binary fields. BTA is a recursive root finding algorithm and relies on properties of trace function. The trace function  $\text{Tr}(\cdot) : \mathbb{F}_2^m \rightarrow \mathbb{F}_2$  is defined by  $\text{Tr}(z) = z + z^2 + z^{2^2} + \dots + z^{2^{m-1}}$ . A key property of the trace function is that, if  $\{\beta_1, \dots, \beta_m\}$  is any basis of  $\mathbb{F}_2^m$  over  $\mathbb{F}_2$ , then every element  $\alpha \in \mathbb{F}_2^m$  is uniquely represented by the binary  $m$ -tuple :  $(\text{Tr}(\beta_1 * \alpha), \dots, \text{Tr}(\beta_m * \alpha))$ . The basic idea of BTA is that, any  $\sigma(z) \in \mathbb{F}_2^m$  that divides  $(z^{2^m} - z)$  is factored into two polynomials  $g(z) = \text{gcd}(\sigma(z), \text{Tr}(\beta * z))$  and  $h(z) =$

$\gcd(\sigma(z), 1 + \text{Tr}(\beta * z))$ . This property of the trace ensures that if  $\beta$  iterates through the basis  $\{\beta_1, \dots, \beta_m\}$ , we can split  $\sigma(z)$  into linear factors. The drawback of BTA is the large number of recursive calls when the system parameters grow. We reduce it by mixing BTA and Zinoviev's algorithms. Thus, we obtain the best results in terms of time complexity for finding roots.

The size of the used parameters varies according to applications. So, we carry out our complexity computations on a range of parameters that we judged relevant. We take into account the increasing necessity of larger security parameters (in this case, extension field degree and degree of the polynomial) for future applications. Therefore, we chose to study solving algebraic equations of degree less than 300 over  $\mathbb{F}_{2^m}$  for extension degrees  $m = 8, 11, 12, 13, 14, 15, 16, 20, 30, 40$ .

Our idea is to compute directly the roots with Zinoviev's algorithms up to some degree  $d_{max}$  and to use recursively BTA for greater degrees. Zinoviev solves this problem for polynomials of degree  $d \leq 10$  in [Zin96]. By the use of dynamic programming and a complexity recurrence formula, we succeeded in determining the best  $d_{max}$  for  $d \leq 300$ . Simulations have been performed in Maple and C language. We determine the algorithm to use by evaluating the number of operations on the field in the worst case. Let us get back to our previous example ( $m = 11, t = 32$ ). Our simulations suggest to choose  $d_{max} = 5$  to obtain the maximum theoretical gain in terms of number of operations, precisely 93% against Chien search and 46% against BTA.

We have a significant margin in terms of time complexity, we can make use of it to enhance the speed of decryption of McEliece-type cryptosystems. A project on implementation of these techniques is still in progress and preliminary results confirm that our techniques are efficient.

**Keywords :** Worst-case complexity, Code-based public-key cryptography, Linearized polynomial, Berlekamp trace algorithm, Error locator polynomial, Algebraic decoding algorithm, Binary Goppa code, BCH code, McEliece-type cryptosystem decryption.

## References

- [Ber68] E. R. Berlekamp. *Algebraic Coding Theory*. 1968.
- [Ber71] E. R. Berlekamp. Factoring polynomials over large finite fields. In *SYMSAC '71: Proceedings of the second ACM symposium on Symbolic and algebraic manipulation*, page 223, New York, NY, USA, 1971. ACM.
- [BS08] Bhaskar Biswas and Nicolas Sendrier. McEliece cryptosystem implementation: Theory and Practice. In *PQCrypto*, pages 47–62, 2008.
- [CCO69] R. T. Chien, B. D. Cunningham, and I. B. Oldham. Hybrid methods for finding roots of a polynomial – with application to BCH decoding. IT-15:329–335, 1969.
- [Zin96] V.A. Zinoviev. On the solution of equations of degree  $\leq 10$  over finite fields  $\text{GF}(2^q)$ . *Rapport de recherche INRIA 2829*, 1996.