Mathematical Problems of Computer Science 33, 144–149, 2010.

Construction of Explicit Irreducible Polynomials over F_2 in Cluster Computational Environment

Ofelya Manukyan and Melsik Kyuregyan

Institute for Informatics and Automation Problems of NAS of RA
e-mail: manofa81@yahoo.com, melsik@ipia.sci.am

Abstract

This paper describes a method for constructing families of explicit irreducible polynomials over F_2. The proposed method allows construction of explicit polynomials of higher degree over F_2 from a given sequence of primitive polynomials. A computational algorithm has been developed and implemented on base of this method. The program is realized in the most effective way possible in cluster computational environment. Allocation and distribution of memory resources have been implemented in a careful manner, since data size increases drastically with increasing of the amount of computations required. Program paralleling is performed using data paralleling, i.e. data is distributing among all processors, which ran the same program and each of which builds the subsequent irreducible polynomial, and finally a sequence of all the irreducible polynomials in explicit form is obtained. Moreover, the program also searches for the polynomial with the lowest possible weight among of all the polynomials of the same degree.

1. Introduction

An efficient way of software realization of elliptic curve cryptosystems is based on presentation of finite field elements by a polynomial basis, if low weight irreducable polynomials (the weight of a polynomial is the number of nonzero coefficients of the polynomial) are used to construct the field. Usage of low weight irreducible polynomials allows faster implementation of basic operations over finite fields. An efficient way of hardware implementation of these cryptosystems is based on the presentation of finite field elements by a normal basis, that is a suitably chosen irreducible polynomial such that its roots form a normal basis and with no restriction imposed on its weight, is used to construct the field. Thus, we shall present a method of construction of explicit polynomials of higher degree over F_2 from a given sequence of primitive polynomials which is interesting from both a theoretical and a practical point of view.
Thus the following application problems are of significant importance: design of parallel software packages giving efficiently implementable algorithms to construct irreducible polynomials over finite fields and determination of the periods of these irreducible polynomials. An efficiently implementable algorithm of constructing irreducible and normal polynomials over finite fields has been designed based on earlier theoretical results. The algorithm allows constructions of sequences of irreducible polynomials of higher degree from a suitably chosen

irreducible polynomial, as well as constructions of irreducible polynomials of higher degree with linearly independent roots.

The major advantage of this algorithm compared with analogous algorithms is that fewer steps are required to perform the algorithm, that withdraws the necessity of solving systems of equations with many variables, and as a result considerably enhances the efficiency of paralleling the algorithm.

Let $L^{\theta} f(x)$ be the operator of Varshamov:

$$L^{\theta} f(x) = \frac{1}{\theta(x)} \sum_{u=0}^{m} \sum_{v=0}^{n} \theta_u a_v x^{uq^v}$$

where $f(x) = \sum_{u=0}^{n} a_u x^u$ and $\theta(x) = \sum_{v=0}^{m} a_v x^v$, $a_u, \theta_v \in F_2$.

Let $\Sigma_\sigma = \{f_1(x), f_2(x), ..., f_\sigma(x)\}$ be a set of σ primitive polynomials with pairwise relatively prime degrees $n_1, n_2, ..., n_\sigma (n_i > 1)$, respectively, over F_2; $T = \prod_{i=1}^{\sigma} (2^{n_i} - 1)$; $\varphi(x)$ be an irreducible polynomial of degree n over F_2; $\gcd(n, T) = 1$; G_σ be the selection of all possible sequences $\varepsilon = (\varepsilon_1, \varepsilon_2, ..., \varepsilon_\sigma)$ of length σ, where $\varepsilon_i = 0$ or 1.

Furthermore, let for any sequences $\varepsilon \in G_\sigma$

$$f(x, \varepsilon, \Sigma_\sigma) = L^x \prod_{i=1}^{\sigma} f_i(x)^{\varepsilon_i},$$

$$xf(x, \varepsilon, \Sigma_\sigma) \equiv R^{(\varepsilon)}(x) (\mathrm{mod}\, \varphi(x)),$$

and $\psi^{(\varepsilon)}(x) = \sum_{u=0}^{n} \psi_u^{(\varepsilon)} x^u$, where $\psi_u^{(\varepsilon)}$ is a nontrivial solution of the congruence

$$\sum_{u=0}^{n} \psi_u^{(\varepsilon)} (R^{(\varepsilon)}(x))^u \equiv 0 (\mathrm{mod}\, \varphi(x)).$$

Then we have the following theorem.

Theorem 1. The polynomials

$$F(x) = (\varphi(x))^{(-1)^\sigma} \frac{\prod_{\varepsilon \in G_\sigma,\ 2|(\sigma - |\varepsilon|)} \psi^{(\varepsilon)}(xf(x, \varepsilon, \Sigma_\sigma))}{\prod_{\varepsilon \in G_\sigma,\ 2\nmid(\sigma - |\varepsilon|)} \psi^{(\varepsilon)}(xf(x, \varepsilon, \Sigma_\sigma))}$$

and $\psi^{(\nu)}(x)$ of degree nT and n, respectively (where $|\varepsilon| = \sum_{i=1}^{\sigma} \varepsilon_i$ and $\nu \in G_\sigma$), are irreducible over F_2.

In this paper we present an overview of the method provided to construct explicitly irreducible polynomials of higher degrees over F_2 from a given sequence of primitive polynomials. In section 2 we give the description of a software package IPG (Irreducible Polynomial Generator) for constructing the polynomials mentioned above and bring up details of main operational blocks of IPG. In section 3 we describe how program parallelization is performed and bring up details of its implementation in Armenian Grid infrastructure.

2. Implementation

Parallel software package IPG (Irreducible Polynomial Generator) is developed for constructing of explicit polynomials of higher degree over F_2 from a given sequence of primitive polynomials. IPG is a software package which runs on a network cluster using MPI as a scheduler. Allocation and distribution of memory resources have been implemented in a careful manner, since data size increases drastically with increasing of the amount of computations required.

The field operations are performed in the most effective way possible. To compute residue polynomials $\bmod \varphi(x)$ in an optimal manner, we compute and keep the set $F[x]/(\varphi(x))$ (equivalence class) of polynomials of degrees less than $\deg(\varphi(x))$ in $F[x]$, i.e. as $\varphi(x)$ is an irreducible polynomial of degree n over F_2, or, equivalently, the field element $x = (0...010)$ is the generator of F_{2^n}, then $F[x]/(\varphi(x))$ is the set of powers of x modulo $\varphi(x)$.

For example, for F_{2^4} with irreducible polynomial $\varphi(x) = x^4 + x + 1$, the computations are summarized in Table 1.

i	$x^i \bmod x^4 + x + 1$	vector notation
0	1	(0001)
1	x	(0010)
2	x^2	(0100)
3	x^3	(1000)
4	$x + 1$	(0011)
5	$x^2 + x$	(0110)
6	$x^3 + x^2$	(1100)
7	$x^3 + x + 1$	(1011)
8	$x^2 + 1$	(0101)
9	$x^3 + x$	(1010)
10	$x^2 + x + 1$	(0111)
11	$x^3 + x^2 + x$	(1110)
12	$x^3 + x^2 + x + 1$	(1111)
13	$x^3 + x^2 + 1$	(1101)
14	$x^3 + 1$	(1001)

Table 1 The powers of $x = (0010)$ modulo $\varphi(x) = x^4 + x + 1$

To compute $p(x) \bmod \varphi(x)$, where $p(x) = \sum_{u=0}^{m} p_u x^u$, we replace the polynomial members $x^u (0 \le u \le m)$ with appropriate elements $x^i (0 \le i \le 2^n - 1)$ from the equivalence class $F[x]/(\varphi(x))$:

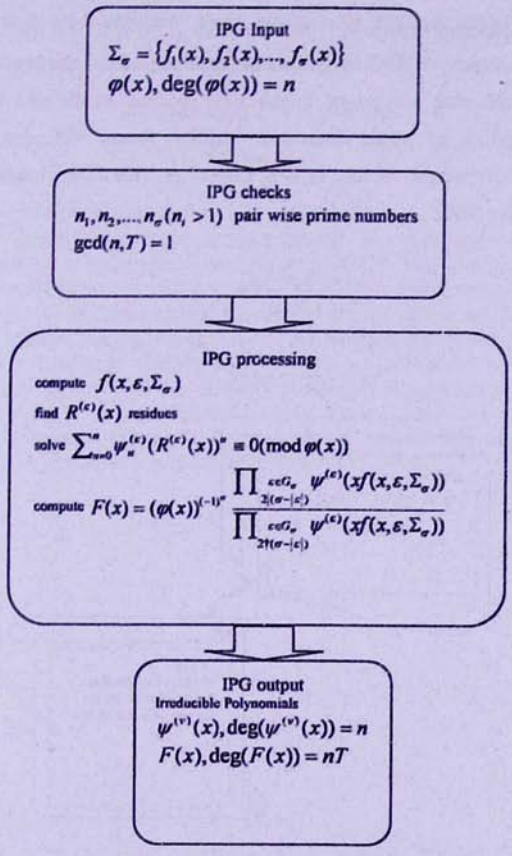
$$x^u \rightarrow x^i, \text{ where } i = u \bmod (2^n - 1)$$

Scheme 1 overviews the main operational blocks of IPG.

3. Parallelization

Program paralleling is performed using data paralleling, i.e. data is distributed among all processors, which run the same program and each of which builds the subsequent irreducible polynomial, and finally a sequence of all the irreducible polynomials in explicit form is obtained. Moreover, the program also searches for the polynomial with the lowest possible weight among of all the polynomials of the same degree.

Our objective is to construct irreducible polynomials with both even/odd and higher degrees from initially given $\Sigma_\sigma = \{f_1(x), f_2(x), \ldots, f_\sigma(x)\}$ set of σ primitive polynomials over F_2 with pairwise relatively prime degrees $n_1, n_2, \ldots, n_\sigma (n_i > 1)$.

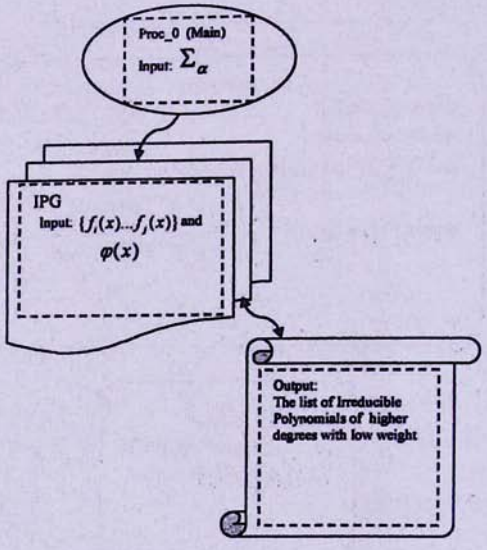


Scheme 1. The main operational blocks of IPG

Let Σ_α be the set of all possible pairs, triples,...,$(\sigma-1)$-tuples of the set Σ_σ and Σ_σ itself:

$$\Sigma_\alpha = \left\{ \begin{array}{l} \{f_1(x), f_2(x)\}, \{f_1(x), f_3(x)\}, ..., \{f_{\sigma-1}(x), f_\sigma(x)\}, \\ \{f_1(x), f_2(x), f_3(x)\}, \{f_1(x), f_3(x), f_4(x)\}, ..., \{f_{\sigma-2}(x), f_{\sigma-1}(x), f_\sigma(x)\}, \\ ..... \\ \Sigma_\sigma = \{f_1(x), f_2(x), ..., f_\sigma(x)\} \end{array} \right\}$$

Task paralleling in the program is realized as follows: one of the processes is considered as the main (Scheme 2 overviews the parallelization in IPG). It distributes the elements of Σ_α among the other processors and registers the outcome. Upon receiving these elements of Σ_α as IPG input, the other processors run IPG and inform the main process on the obtained results, namely send the found irreducible polynomial to the main process. In the end we obtain a list of irreducible polynomials of higher degrees in explicit forms. We also obtain irreducible polynomials with low weight, which, as it is known, provide faster implementation of basic operations over finite fields.



Scheme 2. Parallelization in IPG

References:

[1] M. Kyuregyan, "Recurrent methods for constructing irreducible polynomials over $GF(2^s)$", *Finite Fields and Their Applications 8*, pp. 52-68, 2002.

[2] R. Lidl and H. Niederreiter, Finite Fields , Cambridge University Press 1987.

[3] A. J. Menezes, I. F. Blake, X. Gao, R. C. Mullin, S. A. Vanstone and T. Yaghoobian, "Applications of finite fields", Kluwer Academic Publishers, Boston, Dordrecht, Lancaster, 1993.

F_2 վերջավոր դաշտի վրա անվերածելի բազմանդամների բացահայտ տեսքով կառուցման եղանակ կլաստերային հաշվողական համակարգում

Օ. Մանուկյան և Մ. Կյուրեղյան

Ամփոփում

Աշխատանքում նկարագրված է F_2 վերջավոր դաշտի վրա անվերածելի բազմանդամների բացահայտ տեսքով կառուցման մի եղանակ, որը հնարավորինս էֆեկտիվորեն մշակվել և իրականացվել է կլաստերային հաշվողական համակարգում: Հիշողության ռեսուրսների բաշխումը՝ հիշողության հատկացումն ու ազատումը կատարվել են խնայողաբար, քանի որ հաշվարկների ընթացքում տվյալների երկարությունները շատ արագ աճում են: Կլաստերային հաշվողական համակարգում ծրագրի զուգահեռացումը կատարվել է ըստ տվյալների, այսինքն տվյալները բաշխվում են պրոցեսորների միջև, բոլոր պրոցեսորները աշխատում են միևնույն ծրագրով և յուրաքանչյուրը կառուցում է հերթական անվերածելի բազմանդամը: Արդյունքում ստանում ենք անվերածելի բազմանդամների հաջորդականություն և բացի այդ փնտրում ենք միևնույն աստիճանի հնարավորինս փոքր կշիռ ունեցող անվերածելի բազմանդամը: