# ON COMPLEXIIY OF FFI OVER FINIIE FIELD

V. Afanasyev

Institute for Problem of Information Transmission
Russian Akademy of Science
Ermolovoy 19,GSP-4, MOSCOW, 101447 RUSSIA
e-mail: afanv@ippi.msk.su

**Abstract.** The new algorithms and complexity estimates are considered.

## FFI AS RESIDUES CALCULATION

This structure of FFI ensues from definition of the Fourier Transform. If we write a given input vector $(a_0, a_1, \ldots, a_{n-1})$ as a polynomial $a(x) = \sum_{i=0}^{n-1} a_i x^i$ then

$$A_j = a(\alpha^j) = a(x) \bmod (x - \alpha^j).$$

Let all elements of the field be written in a random order $b_0, b_1, \ldots, b_{q-1}$, where $b_i \neq b_j$ for $i \neq j$, $b_i \in GF(q)$. Then define $b_i$ as the root of polynomial $f_i^{(0)} = (x - b_i)$ at the lowest level $j = 0$ of a polynomial tree. Then for any $j$ we calculate all polynomials $f_k^{(j+1)}$ of the next level of a dichotomic tree where $f_k^{(j+1)} = f_{2k}^{(j)} * f_{2k+1}^{(j)}$, and $k = 0, 1, \ldots, q*2^{-j-1}$. So at the level $j$ we have a mutually prime polynomials of degree $2^j$ expandable over GF(q).

**Procedure FFI:** $R_0^{(m)} = a(x)$;

For $j := m$ to 1 do begin For $k=0$ to $(2^{m-j}-1)$ do
begin $R_{2k}^{(j-1)} = R_k^{(j)} \bmod f_{2k}^{(j-1)}$, $R_{2k+1}^{(j-1)} = R_k^{(j)} \bmod f_{2k+1}^{(j-1)}$
end end;

It is clear that $R_k^{(0)} = a(b_k) = A_k$. □

Asymptotic estimates of the complexity of this Procedure FFT over any finite field had been given in [1] where had been used the general idea of residues computation from [2] and fast

method of multiplication of long integers from [3] and idea of fast polynomial multiplication over a surrogate field from [4]. The complexity of general procedure FFT in binary operations and over any finite field of characteristic 2 and for the case when $m = \log q$ is $C_b = O(q\ (\log q)^4)$

To make the procedure FFT of residue type practicable for a finite fields of reasonable size it is desirable to find a tree of modules witch satisfy to limitations on weights or values of coefficients.

## THE CASE OF BINARY POLYNOMIAL TREE

If we take cyclotomic classes of the field as roots of polynomials then the first level of a tree is formed by only binary irreducible polynomials. Now it is clear how to design any dichotomic tree of binary polynomials for higher levels. At zero level of the tree we use nonbinary modulii of degree 1.

### The procedure of residue calculation :

Part 1: Recurrent calculation of residues by binary modulii from level $m$ to 1.

Part 2: Polynomial evaluation over the points of a cyclotomic classes.

To implement calculations for binary modulii in Part 1 we can split all residues $R_k^{(j)}$ on bit slices. Now we can calculate next residues in each bit slice separately by using a fast convolutions of binary polynomials [1-4].

Let $u(x)$ and $v(x)$ be binary polynomials of degree $n$ or less. Define the associate integer $\mathcal{U}$ over the base $2^m$ as $\mathcal{U}=\sum_{i=0}^{n-1} 2^{mi} u_i$ where $u_i$ are binary coefficients of $u(x)$ and the same for $\mathcal{V}$. Then $\mathcal{Z}=[\mathcal{U}\cdot\mathcal{V}]_{mod2} = \sum_{i=0}^{n-1} 2^{mi} (z_i \bmod 2)$ is the associate integer to product $[u(x)\cdot v(x)]$ when $m \geq 2\log(n)$. For large $n$ we can use the method [3] of multiplying integers of length $\geq 2n\log(n)$ with the binary complexity $O(n\log^2(n)\log(\log(n)))$.

The total binary complexity estimate of Part1 of the procedure [2] for $m$ bit slices is

$O(n\log^4(n)\log(\log(n)))$, $n\leq 2^m$.

To estimate the complexity of Part 2 we also use bit splitting. Let $u(x)$ be a polynomial over $GF(2^m)$ of degree less $m$. Then $u(x)=\sum_i \beta_i u_i(x)$ where $\{\beta_0 \ldots \beta_{m-1}\}$ is the finite field basis and $u_i(x)$ are binary polynomials. If we calculated $u_i(\gamma)$ for some $\gamma\in GF(2^m)$ then $u(\gamma)=\sum_i \beta_i u_i(\gamma)$ and $u(\gamma^{2^j})=\sum_i \beta_i (u_i(\gamma))^{2^j}$.

The total binary complexity estimate for Part 2 of the procedure is $O(n\log^3(n))$.

**Theorem:** Over any given finite field $GF(2^m)$ there exists a $FFT(n)$ procedure of binary complexity $C_b=O(nm^4)$ where $n\leq 2^m$.

The considered FFT method is close to [5] but has better asymptotic behavior.

## THE CASE OF AFFINE POLYNOMIAL TREE

One of important examples of sparse polynomials is the class of *affine* polynomials. An affine polynomial over $GF(q^m)$ is defined as the sum of *linearized* polynomial $L(x)$ and field element $\beta$ where $L(x)=\sum_{i=0}^{n} l_i x^{q^i}$, $l_i\in GF(q^m)$. The roots of $L(x)$ form a linear space denoted as $\mathfrak{G}$. When $n<m$ then $\mathfrak{G}$ is a subspace of $GF(q^m)$ of dimension $n$. If $\xi\in GF(q^m)\setminus\mathfrak{G}$ then shift $\mathfrak{G}$ by $\xi$ is an affine subspace of $GF(q^m)$ and its characteristic polynomial is $B(x)=L(x+\xi)=L(x)+L(\xi)$. Now we see that the characteristic polynomials of all cosets of a $\mathfrak{G}$ have $L(x)$ as the linearized part. Now we can construct a tree of affine polynomials over the given finite field.

**The method:** Let us take any *basis* of the given $GF(q)$ and delete any one element from this basis. It is the basis $\mathfrak{B}_1$ of a subspace of $GF(q^m)$. All elements of the subspace are roots of a linearized polynomial $L^{(m-1)}$ at the (m-1)-st level of the tree. Now we can compute this polynomial and its affine shifts.

To bild a nested structure of subspaces we take $\mathfrak{B}_1$ and delete any one element from. Now we have the basis $\mathfrak{B}_2$ and can compute the polynomial $L^{(m-2)}$ and all its affine shifts.

We continue this procedure up to the zero level of the

tree. The similar procedure may be realized from lower levels to the upper levels of the tree.

**Calculation** of $L^{(m-1)}$: Let us denote as $\beta_{ij}$ the elements of the basis $\mathfrak{B}_i$. The dimension of $\mathfrak{B}_i$ is $m-i$. As all $\beta_{ij}$ are roots of $L^{(m-1)}$ by definition then we have the system of linear equations of order $m-i$

$$L_0^{(m-1)}\beta_{ij}^{q^0}+L_1^{(m-1)}\beta_{ij}^{q^1}+\ldots+L_{m-1-1}^{(m-1)}\beta_{ij}^{q^{m-1-1}}+L_{m-1}^{(m-1)}\beta_{ij}^{q^{m-1}}=0,$$

for $j=1,m-i$

with unknown values $L_0^{(m-1)},L_1^{(m-1)},\ldots,L_{m-1-i}^{(m-1)}$ and $L_{m-i}^{(m-1)}=1$ by definition.

The affine shifts of $L^{(m-1)}$ are $L^{(m-1)}+\alpha*L^{(m-1)}(\beta_i)$ where $\beta_i$ is an element of expansion $\mathfrak{B}_i$ to $\mathfrak{B}_{i-1}$ and $\alpha\in GF(q)$. The affine shifts of $L^{(m-1)}$ in other subtrees are

$$L^{(m-1)}+\sum_{j=1}^{i}\alpha_j*L^{(m-1)}(\beta_j)$$

In the most important case of $GF(2^m)$ when $m$ is any integer we have a dichotomic tree of affine polynomials.

**Theorem**. For any $GF(2^m)$ there exist a dichotomic tree of affine polinomials of the height $m$ with identical linearized part of degree $2^i$ at $i$-th level, $i=0,\ldots,m-1$. The upper bound of the number additions and multiplications over $GF(2^m)$ in FFT procedure generated by this tree is $2^{m-2}(m^2+m-4)+1$.

Note that this estimate is close to the estimate in [6] but in [6] there is no constructive procedure of a affine tree design.

Now consider the special case of $m=2^k$. In this case all linearized parts are binary polynomials. The design of the linearized part is connected very close with squaring of the field as expanded field generation procedure. For $GF(2)$ we have $L^{(1)}=x^2+x$ and $L^{(0)}=x$. For $GF(2^2)$ we have $L^{(2)}=x^4+x$. For $GF(2^4)$ we have $L^{(4)}=L^{(2)}(L^{(2)})=x^{16}+x$, $L^{(3)}=x^8+x^4+x^2+x$. Let us denote as $\{L^{(j)}\}$ the sequence $L^{(j)},\ldots,L^{(0)}$. Now we see that $L^{(2^{i+1})}=L^{(2^i)}(L^{(2^i)})$ and $\{L^{(2^{i+1})}\}=L^{(2^i)}(\{L^{(2^i)}\})$. Let us name this procedure as *tree generation by squaring*. To complete the tree design we calculate all affine shifts as in the

general procedure.

At $i$-th level of the tree the affine polinomials adjacent to one of them with free member $\xi$ at the $(i-1)$-th level have a free members as the roots of the equation $x^2+x+\xi$. Because sum of this roots is equal 1 only one residue has to be calculated with multiplications over finite field from the couple of residues connected with this roots. Note that the total set of free members of the tree are roots of $L^{(1)}(x)=\sum_{i=0}^{m-1} x^{2^i}$ and therefore has zero trace. So the number of multiplications over the field for this case is less than $2^{m-1}(m-1)$. This estimate is also very close to the estimate in [6]

The number of additions is proportional to the weight of linearized part of the tree. As it follows from the tree design the weight is equal $m^{\log(3)}$. So number of additions is less than $2^m\ m^{\log(3)}$.

**Theorem** : For $m=p\cdot 2^k$ there exist a dichotomic tree expanded from affine tree over $GF(2^p)$ by squaring procedure. The FFT procedure over this tree has multiplicative complexity less than $2^m(m+M_p\cdot 3^k)$ and additive complexity less than $2^m(m+M_a\cdot 3^k)$ where $M_p$ and $M_a$ are multiplicative and additive complexities over $GF(2^p)$.

## REFERENCES.


[1] Afanasyev V.B.,"Bystroje decodirovanie BCH kodov", *Voprosy kibernetiki. Kodirovanie i peredacha informatsii v vychislitelnykh setyah.*-M.,Sov.Radio, 1978, vyp. 42.

[2] Aho, Hopcroft and Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, 1974.

[3] Schonhage,A., and V.Strassen, "Schnelle Multiplikation Grosser Zahlen", *Computing* 7, 281-292, 1971.

[4] Preparata F.P. and Sarvate D.V., "Computational Complexity of Fourier Transforms over Finite Fields", *Math.Comp.* 1977,v.31,N 139.

[5] Zakharova T.G.,"Calculation of Fourier Transform in Fields of Characteristic 2", *Problemy Peredachi Informatsii*, vol.28, no 2, 1992, M.: Nauka.

[6] Yao Wang and Xuelong Zhu, "A Fast Algorithm for the Fourier Transform Over Finite Fields and its VLSI Implementation", *IEEE Journal on Selected Areas in Communications*, Vol.6, No.3, April 1988.