A Residue Approach of the Finite Field Arithmetics

JC Bajard

LIRMM, CNRS UM2
161 rue Ada, 34392 Montpellier cedex 5, France
Contents

Introduction to Residue Systems

Modular reduction in Residue Systems

Applications to Cryptography

Conclusion
Introduction to Residue Systems
Introduction to Residue Systems

▶ In some applications, like cryptography, we use finite field arithmetics on huge numbers or large polynomials.
▶ Residue systems are a way to distribute the calculus on small cells.
▶ Are these systems available for finite fields?
Residue Number Systems in \( GF(p) \), \( p \) prime

- Modular arithmetic mod \( p \), elements are considered as integers.
- **Residue Number System**
  - RNS base: a set of coprime numbers \((m_1, \ldots, m_k)\)
  - RNS representation: \((a_1, \ldots, a_k)\) with \(a_i = |A|_{m_i}\)
  - Full parallel operations mod \( M \) with \( M = \prod_{i=1}^{k} m_i \)
    \(|a_1 \otimes b_1|_{m_1}, \ldots, |a_n \otimes b_n|_{m_n}\) \(\rightarrow\) \(A \otimes B \) (mod \( M \))
- Very fast product, but an extension of the base could be necessary and a reduction modulo \( p \) is needed.
Lagrange representations in $GF(p^k)$ with $p > 2k$

- Arithmetic modulo $I(X)$, an irreducible $GF(p)$ polynomial of degree $k$. Elements of $GF(p^k)$ are considered as $GF(p)$ polynomials of degree lower than $k$.

- Lagrange representation
  - is defined by $k$ different points $e_1, ... e_k$ in $GF(p)$. ($k \leq p$.)
  - A polynomial $A(X) = \alpha_0 + \alpha_1 X + ... + \alpha_{k-1} X^{k-1}$ over $GF(p)$ is given in Lagrange representation by:
    $$ (a_1 = A(e_1), ..., a_k = A(e_k)). $$
  - Remark: $a_i = A(e_i) = A(X) \mod (X - e_i)$. If we note $m_i(X) = (X - e_i)$, we obtain a similar representation as RNS.

- Operations are made independently on each $A(e_i)$ (like in FFT or Tom-Cook approaches). We need to extend to $2k$ points for the product.
Trinomial residue in $GF(2^n)$

- Arithmetic modulo $I(X)$, an irreducible $GF(2)$ polynomial of degree $n$. Elements of $GF(2^n)$ are considered as $GF(2)$ polynomials of degree lower than $n$.

- Trinomial representation
  - is defined by a set of $k$ coprime trinomials $m_i(X) = X^d + X^{t_i} + 1$, with $k \times d \geq n$,
  - an element $A(X)$ is represented by $(a_1(X),...a_k(X))$ with $a_i(X) = A(X) \mod m_i(X)$.
  - This representation is equivalent to RNS.

- Operations are made independently on each $a_i(X)$
Residue Systems

- Residue systems could be an issue for computing efficiently the product.
- The main operation is now the modular reduction for constructing the finite field elements.
- The choice of the residue system base is important, it gives the complexity of the basic operations.
Modular reduction in Residue Systems
Reduction of Montgomery

- The most used reduction algorithm is due to Montgomery (1985)[9]
- For reducing $A$ modulo $p$, it evaluates $q = -(Ap^{-1}) \mod 2^s$, then it constructs $R = (A + qp)/2^s$. The obtained value satisfies: $R \equiv A \times 2^{-s} \mod p$ and $R < 2p$ if $A < p2^s$. We note $\text{Montg}(A, 2^s, p) = R$.
- Montgomery notation: $A' = A \times 2^s \mod p$
  $\text{Montg}(A' \times B', 2^s, p) \equiv (A \times B) \times 2^s \mod p$
Residue version of Montgomery Reduction

- The residue base is such that $p < M$
  (or $\deg M(X) \geq \deg I(X)$)
- We use an auxiliary base such that $p < M'$
  (or $\deg M'(X) \geq \deg I(X)$), $M'$ and $M$ coprime.
  (Exact product, and existence of $M^{-1}$)
- Steps of the algorithm
  1. $q = -(Ap^{-1}) \mod M$ (calculus in base $M$)
  2. Extension of the representation of $q$ to the base $M'$
  3. $R = (A + qp) \times M^{-1}$ (calculus in base $M'$)
  4. Extension of the representation of $R$ to the base $M$
- The values are represented in the two bases.
Extension of Residue System Bases (from $M$ to $M'$)

The extension comes from the Lagrange interpolation. If $(a_1, ..., a_k)$ is the residue representation in the base $M$, then

$$A = \sum_{i=1}^{k} a_i \times \left[ \frac{M}{m_i} \right]^{-1} \bigg|_{m_i} \times \frac{M}{m_i} - \alpha M$$

The factor $\alpha$ can be in certain cases, neglected or computed. [1] Another approach consists in the Newton interpolation where $A$ is correctly reconstructed. [4]
Extension of Residue System Bases (from $M$ to $M'$)

We first translate in an intermediate representation (MRS):

\[
\begin{align*}
\zeta_1 &= a_1 \\
\zeta_2 &= (a_2 - \zeta_1) m_1^{-1} \mod m_2 \\
\zeta_3 &= ((a_3 - \zeta_1) m_1^{-1} - \zeta_2) m_2^{-1} \mod m_3 \\
&\hspace{1cm} \vdots \\
\zeta_n &= (\ldots ((a_n - \zeta_1) m_1^{-1} - \zeta_2) m_2^{-1} - \cdots - \zeta_{n-1}) m_{n-1}^{-1} \mod m_n.
\end{align*}
\]

We evaluate $A$, with Horner's rule, as

\[
A = (\ldots ((\zeta_n m_{n-1} + \zeta_{n-1}) m_{n-2} + \cdots + \zeta_3) m_2 + \zeta_2) m_1 + \zeta_1.
\]
Features of the residue system

- Efficient multiplication, the cost being the cost of one multiplication on one residue.
- Costly reduction: $O(k^{1.6})$ for trinomials [4], $2k^2 + 3k$ for RNS [1], $O(k^2)$ for Lagrange representation [5].
- If we take into account that most of the operations are multiplications by a constant, the cost can be considerably smaller.
Applications to Cryptography
Elliptic curve cryptography

- The main idea comes from the efficiency of the product and the cost of the reduction in Residue Systems.

- We try to minimize the number of reductions. A reduction is not necessary after each operation. Clearly, for a formula like $A \times B + C \times D$, only one reduction is needed.

- Elliptic Curve Cryptography is based on points addition. We use appropriate forms (Hessian, Jacobi, Montgomery Ladder...) and coordinates: projective, Jacobian or Chudnowski.

- For 512 bits values, Residues Systems, for curves defined over a prime field, are more efficient than classical representations. [2]
Pairings

- Summarizing, we define a pairing as following: $G_1$ and $G_2$ two additive abelian groups of cardinal $n$ and $G_3$ a cyclic (multiplicative) group of cardinal $n$.

- A pairing is a function $e : G_1 \times G_2 \rightarrow G_3$ which verifies the following properties: Bilinearity, Non-degeneracy.

- For pairings defined on an elliptic curve $E$ over a finite field $GF(p)$, we have $G_1 \subset E(GF(p))$, $G_2 \subset E(GF(p^k))$ and $G_3 \subset GF(p^k)$, where $k$ is the smallest integer such that $n$ divides $p^k - 1$, $k$ is called the embedded degree of the curve.
Pairings

- The construction of the pairing implies values over $GF(p)$ and $GF(p^k)$ into the formulas. An approach with Residue Systems, similar to the one made on ECC could be interesting.[3]
- $k$ is most of the time chosen as a small power of 2 and 3 for algorithmic reasons. Residue arithmetics allow to pass over this restriction.
- With pairings, we can also imagine two levels of Residue Systems: one over $GF(p)$ and one over $GF(p^k)$. 
Conclusion on Residue Systems
Conclusions

- We have seen that Residue Systems give some good results for ECC over $GF(p)$.
- We will extend these studies for ECC over $GF(p^k)$ and $GF(2^k)$.
- Residue Systems offer to pairings an opening to a large variety of embedded degrees and finite fields. We remind that the security is given by the one of ECC over $GF(p)$ and by the discrete logarithm over $GF(p^k)$. 


Garner, H.L.: The residue number system. IRE Transactions on Electronic Computers, EL 8:6 (1959) 140–147

