Modular Number Systems: Beyond the Mersenne family

Jean-Claude Bajard
Laurent Imbert
Thomas Plantard
-
SAC 2004

LIRMM - Montpellier - France
ATIPS - Calgary - Canada

Contents

Context

- Many cryptographic protocols use modular arithmetic
  - ECC: uses a prime number P, $160 < |P| < 500$
  - RSA: uses composite number N, $1024 < |N|$
- We need:
  - fast modular algorithm ...
  - ... for a large class of moduli.
- Remark: Any modular operation can be decomposed in the equivalent classical operation, followed by a modular reduction.

Modular Reduction

Input

- Constant: P with $n = |P|$,the length of P
- Variable: X, the result of a multiplication: $0 \leq X < P^2$

Output

- Variable: R with $R = X \bmod P$

Modular Reduction

Input

- Constant: P with $n = |P|$,the length of P
- Variable: X, the result of a multiplication: $0 \leq X < P^2$

Output

- Variable: R with $R = X \bmod P$

Example

- $P = 31$ and $n = 5$
- $X = 21 \times 13 = 273$
- $R = 25$
  $X = 25 + 8 \times 31 = 273$

Some interesting classes of moduli

Mersenne's number

- $P = 2^n - 1$

Algorithm

- $2^n \equiv 1 \pmod{P}$
- $X = X_1 2^n + X_0$
- $X \equiv X_1 + X_0 \pmod{P}$

- Advantage: cost = one addition
- Drawback: Prime Mersenne's number class is too small

Some interesting classes of moduli

Mersenne's number

- $P = 2^n - 1$

Algorithm

- $2^n \equiv 1 \pmod{P}$
- $X = X_1 2^n + X_0$
- $X \equiv X_1 + X_0 \pmod{P}$

Example: $P = 31$, $X = 273$

- $2^5 \equiv 1 \pmod{31}$
- $X = 8 \times 2^5 + 17$
- $R = 8 + 17 = 25$

- Advantage: cost = one addition
- Drawback: Prime Mersenne's number class is too small

Pseudo Mersenne

- Introduced by Crandall in 1992.
- $P = 2^n - c$ with c small integer
- Example: $n = 10, c = 3 \longrightarrow P = 1021$

Pseudo Mersenne

- Introduced by Crandall in 1992.
- $P = 2^n - c$ with c small integer
- Example: $n = 10, c = 3 \rightarrow P = 1021$

Generalized Mersenne

- Introduced by Solinas in 1999.
- $P = f(2^t)$ where f is a polynomial with coefficients in $\{0, 1\}$
- Example: $t = 3, f(x) = x^3 - x - 1 \rightarrow P = 8^3 - 8 - 1 = 503$

Pseudo Mersenne

- Introduced by Crandall in 1992.
- $P = 2^n - c$ with c small integer
- Example: $n = 10, c = 3 \rightarrow P = 1021$

Generalized Mersenne

- Introduced by Solinas in 1999.
- $P = f(2^t)$ where f is a polynomial with coefficients in $\{0, 1\}$
- Example: $t = 3, f(x) = x^3 - x - 1 \rightarrow P = 8^3 - 8 - 1 = 503$

More generalized Mersenne

- Introduced by Chung and Hassan in SAC 2003.
- $P = f(2^t - c)$ with $f_i = \{0, 1\}$
- Example: $f(x) = x^4 - x^3 - 1 \rightarrow P = f(2^4 - 2) = 35671$

Number system

Classical number system with radix β

$$X = \sum_{i=0}^{n-1} x_i \beta^i \text{ with } x_i \in \{0, ...\beta - 1\}$$

Example: $X = 1315 = [3, 4, 4, 2]_8$ $X = 3 + 4 \times 8 + 4 \times 8^2 + 2 \times 8^3$

Number system

Classical number system with radix β

$$X = \sum_{i=0}^{n-1} x_i \beta^i \text{ with } x_i \in \{0, ...\beta - 1\}$$

Example: $X = 1315 = [3, 4, 4, 2]_8 \; X = 3 + 4 \times 8 + 4 \times 8^2 + 2 \times 8^3$

Modular number system (γ, ρ, n, P)

$$X = \sum_{i=0}^{n-1} x_i \gamma^i \bmod P \text{ with } x_i \in \{0, \ldots, \rho - 1\}$$

Example

MNS

- $(\gamma = 7, \rho = 3, n = 3, P = 17)$
- $X = \sum_{i=0}^{2} x_i 7^i \bmod 17$ with $x_i \in \{0, 1, 2\}$

Table with $7^0 = 1, 7^1 = 7, 7^2 \bmod 17 = 15$

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	

Example

MNS

- $(\gamma = 7, \rho = 3, n = 3, P = 17)$
- $X = \sum_{i=0}^{2} x_i 7^i \bmod 17$ with $x_i \in \{0, 1, 2\}$

Table with $7^0 = 1, 7^1 = 7, 7^2 \bmod 17 = 15$

0	1	2	3	4	5
$[0, 0, 0]$	$[1, 0, 0]$	$[2, 0, 0]$			

6	7	8	9	10	11

12	13	14	15	16

Example

MNS

- $(\gamma = 7, \rho = 3, n = 3, P = 17)$
- $X = \sum_{i=0}^{2} x_i 7^i \bmod 17$ with $x_i \in \{0, 1, 2\}$

Table with $7^0 = 1, 7^1 = 7, 7^2 \bmod 17 = 15$

0	1	2	3	4	5
$[0,0,0]$	$[1,0,0]$	$[2,0,0]$			
6	7	8	9	10	11
	$[0,1,0]$	$[1,1,0]$	$[2,1,0]$		
12	13	14	15	16	

Example

MNS

- $(\gamma = 7, \rho = 3, n = 3, P = 17)$
- $X = \sum_{i=0}^{2} x_i 7^i \bmod 17$ with $x_i \in \{0, 1, 2\}$

Table with $7^0 = 1, 7^1 = 7, 7^2 \bmod 17 = 15$

0	1	2	3	4	5
[0, 0, 0]	[1, 0, 0]	[2, 0, 0]			
6	7	8	9	10	11
	[0, 1, 0]	[1, 1, 0]	[2, 1, 0]		
12	13	14	15	16	
		[0, 2, 0]	[1, 2, 0]	[2, 2, 0]	

Example

MNS

- $(\gamma = 7, \rho = 3, n = 3, P = 17)$
- $X = \sum_{i=0}^{2} x_i 7^i \bmod 17$ with $x_i \in \{0, 1, 2\}$

Table with $7^0 = 1, 7^1 = 7, 7^2 \bmod 17 = 15$

0	1	2	3	4	5
$[0,0,0]$	$[1,0,0]$	$[2,0,0]$			$[0,1,1]$
6	7	8	9	10	11
$[1,1,1]$	$[0,1,0]$	$[1,1,0]$	$[2,1,0]$		
12	13	14	15	16	
$[0,2,1]$	$[1,2,1]$	$[0,2,0]$	$[1,2,0]$	$[2,2,0]$	

Example

MNS

- $(\gamma = 7, \rho = 3, n = 3, P = 17)$
- $X = \sum_{i=0}^{2} x_i 7^i$ mod 17 with $x_i \in \{0, 1, 2\}$

Table with $7^0 = 1, 7^1 = 7, 7^2$ mod $17 = 15$

0	1	2	3	4	5
$[0, 0, 0]$	$[1, 0, 0]$	$[2, 0, 0]$	$[1, 1, 2]$	$[2, 1, 2]$	$[0, 1, 1]$
6	7	8	9	10	11
$[1, 1, 1]$	$[0, 1, 0]$	$[1, 1, 0]$	$[2, 1, 0]$	$[0, 2, 2]$	$[1, 2, 2]$
12	13	14	15	16	
$[0, 2, 1]$	$[1, 2, 1]$	$[0, 2, 0]$	$[1, 2, 0]$	$[2, 2, 0]$	

How find a “good” Modular Number System?

Definition: AMNS

A modular number system $\mathcal{B} = MNS(\gamma, \rho, n, P)$ is called Adapted Modular Number System (AMNS) if $\gamma^n \bmod P = c$ with c small integer.

How find a “good” Modular Number System?

Definition: AMNS

A modular number system $\mathcal{B} = MNS(\gamma, \rho, n, P)$ is called Adapted Modular Number System (AMNS) if $\gamma^n \bmod P = c$ with c small integer.

Modular Multiplication in AMNS

1. Polynomial multiplication in $\mathbb{Z}[X]$: $U(X) \leftarrow A(X)\,B(X)$
2. Polynomial reduction: $V(X) \leftarrow U(X) \bmod (X^n - c)$
3. Coefficient reduction: $S \leftarrow CR(V)$, gives $S \equiv V(\gamma) \pmod{P}$

AMNS

- $P = 250043 \Rightarrow |P| = 18$
- $n = 3,\ \rho = 2^7$
- $\gamma = 127006$ such that $c = 2 = \gamma^3 \bmod P$

Input

- $A = 7 + 30X + 100X^2 \Rightarrow A = 65842$
- $B = 59 + 2X + 76X^2 \Rightarrow B = 8816$

Algorithm

1. $U(X) = A(X) \times B(X)$
   $U(X) = 413 + 1784X + 6492X^2 + 2480X^3 + 7600X^4$
2. $V(X) = U(X) \bmod (X^3 - 2) \leftarrow 5373 + 16984X + 6492X^2$
3. $S(X) = ?$

Input

- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$

Input

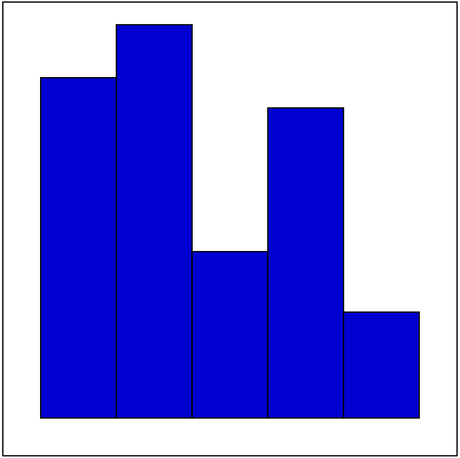
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k + 1$ DO
    1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
    2. $\overline{S} \leftarrow RED(\overline{S})$
    3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
    4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

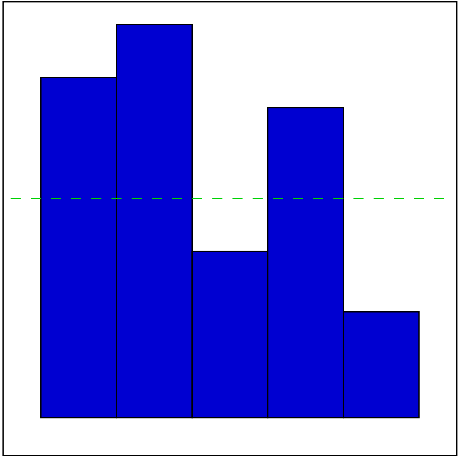
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k + 1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

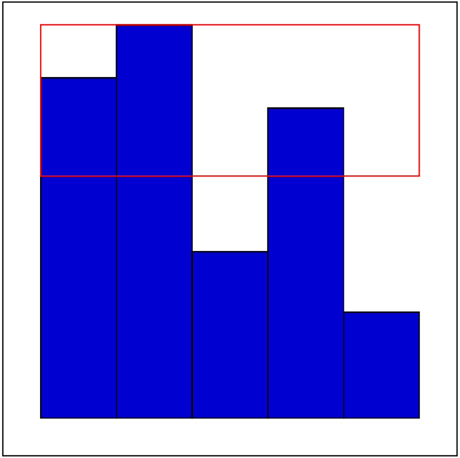
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

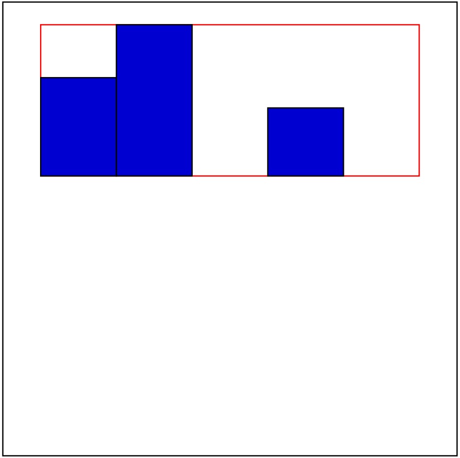
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k + 1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

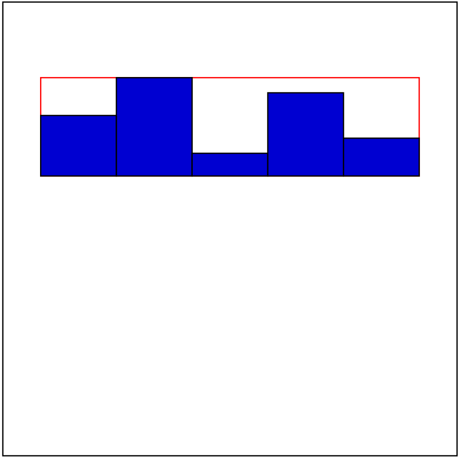
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
    1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
    2. $\overline{S} \leftarrow RED(\overline{S})$
    3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
    4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

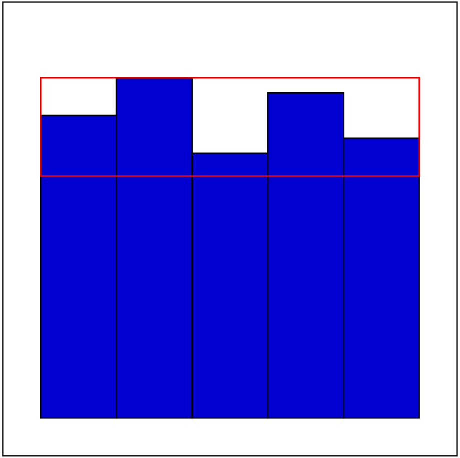
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

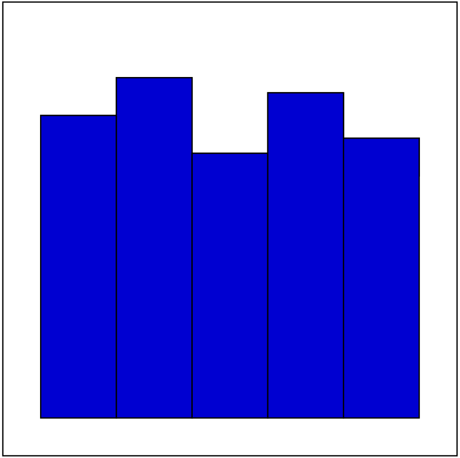
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
    1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
    2. $\overline{S} \leftarrow RED(\overline{S})$
    3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
    4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

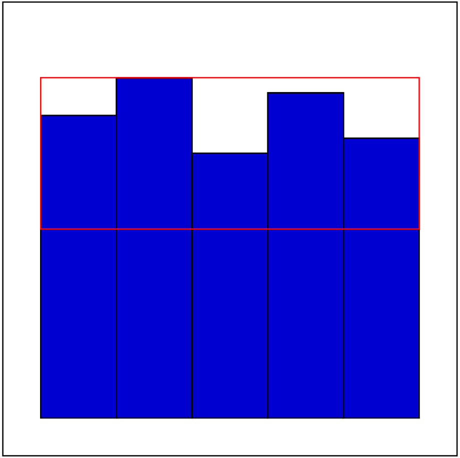
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

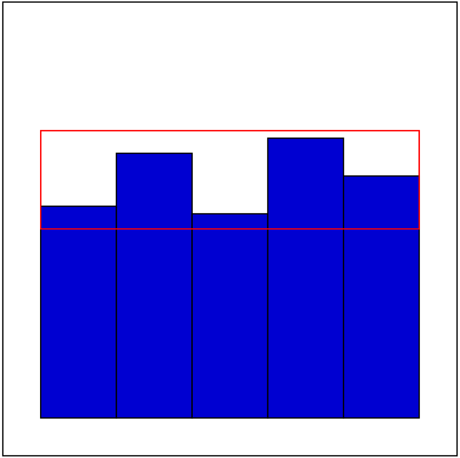
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

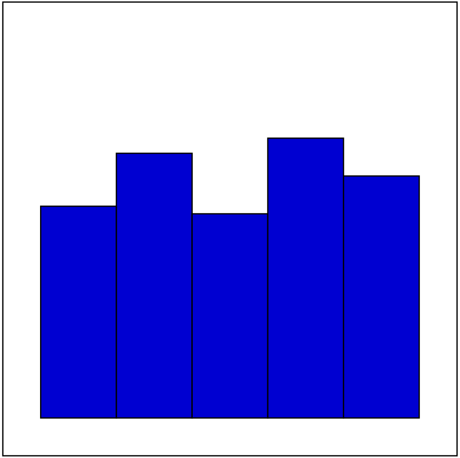
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k + 1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

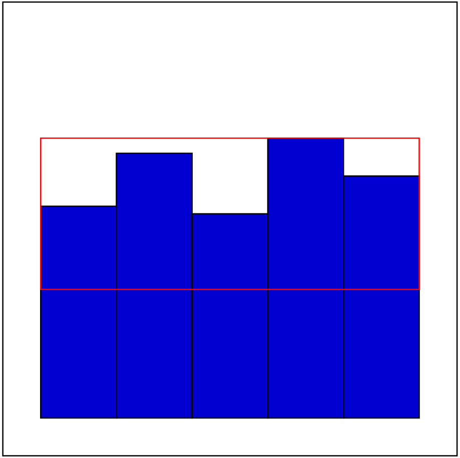
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

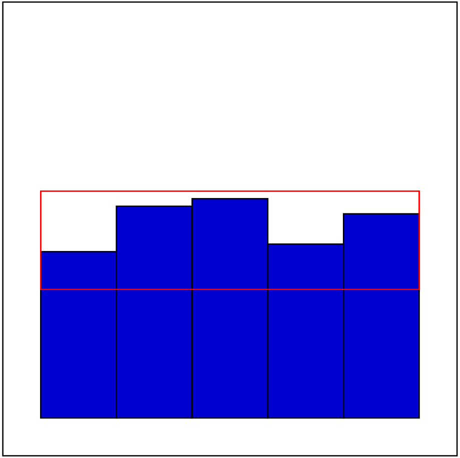
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

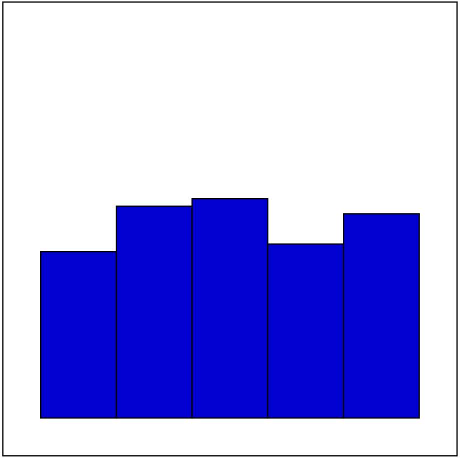
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k+1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



Input

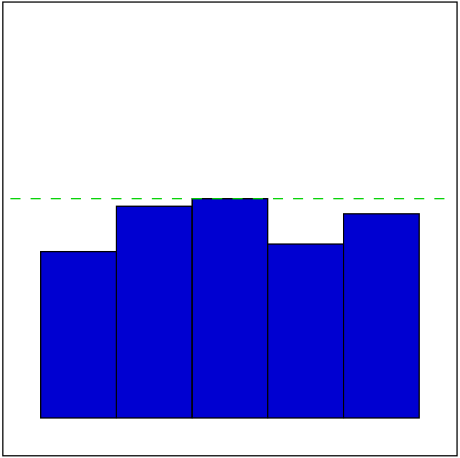
- A vector V

Algorithm

1. $S \leftarrow V, t \leftarrow |S|_2$
2. WHILE $t > k + 1$ DO
   1. $S = \overline{S}2^{t-3k/2} + \underline{S}$
   2. $\overline{S} \leftarrow RED(\overline{S})$
   3. $S \leftarrow \overline{S}2^{t-3k/2} + \underline{S}$
   4. $t \leftarrow t - (k/2 - 1)$

Output

- A vector $S \equiv V$
- With coefficients $S_i < \rho = 2^{k+1}$



The Algorithm *RED*

Input

- A vector V with its coefficients $V_i < 2^{3k/2}$

Algorithm *RED*

1. $V = \overline{V}2^k + \underline{V}$
2. $S \leftarrow \overline{V}M + \underline{V}$, where $M \equiv 2^k Id$

Output

- A vector $S \equiv V$
- With its coefficients $S_i < 2^{k+1}$

How find M with $M \equiv 2^k I$

Condition

- A vector ξ which represent 2^k: $2^k \equiv \xi[\gamma] \pmod{P}$
- With small coefficients: $\sum_{i=0}^{n-1} \xi_i < 2^{\lfloor k/2 \rfloor}/c$

How find M with $M \equiv 2^k I$

Condition

- A vector ξ which represent 2^k: $2^k \equiv \xi[\gamma] \pmod{P}$
- With small coefficients: $\sum_{i=0}^{n-1} \xi_i < 2^{\lfloor k/2 \rfloor}/c$

How to build M

$$\begin{pmatrix} 2^k & 0 & \cdots & 0 & 0 \\ 0 & 2^k & \cdots & 0 & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \cdots & 2^k & 0 \\ 0 & 0 & \cdots & 0 & 2^k \end{pmatrix} \equiv \begin{pmatrix} & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{pmatrix} \qquad (1)$$

How find M with $M \equiv 2^k I$

Condition

- A vector ξ which represent 2^k: $2^k \equiv \xi[\gamma] \pmod{P}$
- With small coefficients: $\sum_{i=0}^{n-1} \xi_i < 2^{\lfloor k/2 \rfloor}/c$

How to build M

$$\begin{pmatrix} 2^k & 0 & \cdots & 0 & 0 \\ & & & & \\ & & & & \\ & & & & \end{pmatrix} \equiv \begin{pmatrix} \xi_0 & \xi_1 & \cdots & \xi_{n-2} & \xi_{n-1} \\ & & & & \\ & & & & \\ & & & & \end{pmatrix} \quad (1)$$

How find M with $M \equiv 2^k I$

Condition

- A vector ξ which represent 2^k: $2^k \equiv \xi[\gamma] \pmod{P}$
- With small coefficients: $\sum_{i=0}^{n-1} \xi_i < 2^{\lfloor k/2 \rfloor}/c$

How to build M

$$\begin{pmatrix} 2^k & 0 & \cdots & 0 & 0 \\ 0 & 2^k & \cdots & 0 & 0 \\ & & & & \\ & & & & \\ & & & & \end{pmatrix} \equiv \begin{pmatrix} \xi_0 & \xi_1 & \cdots & \xi_{n-2} & \xi_{n-1} \\ c\xi_{n-1} & \xi_0 & \cdots & \xi_{n-3} & \xi_{n-2} \\ & & & & \\ & & & & \\ & & & & \end{pmatrix} \tag{1}$$

How find M with $M \equiv 2^k I$

Condition

- A vector ξ which represent 2^k: $2^k \equiv \xi[\gamma] \pmod{P}$
- With small coefficients: $\sum_{i=0}^{n-1} \xi_i < 2^{\lfloor k/2 \rfloor}/c$

How to build M

$$\begin{pmatrix} 2^k & 0 & \cdots & 0 & 0 \\ 0 & 2^k & \cdots & 0 & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & \cdots & 2^k & 0 \\ 0 & 0 & \cdots & 0 & 2^k \end{pmatrix} \equiv \begin{pmatrix} \xi_0 & \xi_1 & \cdots & \xi_{n-2} & \xi_{n-1} \\ c\xi_{n-1} & \xi_0 & \cdots & \xi_{n-3} & \xi_{n-2} \\ \vdots & & & & \vdots \\ c\xi_2 & c\xi_3 & \cdots & \xi_0 & \xi_1 \\ c\xi_1 & c\xi_2 & \cdots & c\xi_{n-1} & \xi_0 \end{pmatrix} \quad (1)$$

Input

- $AMNS(\gamma = 127006, \rho = 128, n = 3, P = 250043)$ with $\gamma^n \equiv 2$
- $\gamma^3 = 2 \bmod P$ and $2^6 = 1 + \gamma^2 \bmod P$

$$\begin{pmatrix} 2^6 & 0 & 0 \\ 0 & 2^6 & 0 \\ 0 & 0 & 2^6 \end{pmatrix} \equiv \begin{pmatrix} 1 & 0 & 1 \\ 2 & 1 & 0 \\ 0 & 2 & 1 \end{pmatrix} \tag{2}$$

- A vector $V = [120, 444, 22]$ with $V_i < 2^{3k/2} = 2^9$

RED

1. $V = [1, 6, 0]2^6 + [56, 60, 22]$
2. $S \leftarrow [1, 6, 0]M + [56, 60, 22] = [1, 8, 12] + [56, 60, 22]$

Output

$S = [57, 68, 34]$ with $S_i < 2^{k+1} = 2^7 = 128$

An Example of Coefficient Reduction

Input

- $AMNS(\gamma = 127006, \rho = 128, n = 3, P = 250043)$ with $\gamma^n \equiv 2$
- $V = [5373, 16984, 6492]$

Step

1. $S = [1853, 984, 2524]$
2. $S = [357, 544, 532]$
3. $S = [121, 56, 32]$

Output

$S = [121, 56, 32]$ with $S_i < 128$

How to find convenient P?

How to make a AMNS?

1. Choose $\rho = 2^{k+1}$ with $k = 15, 31, 63.$
2. Define n such that $|P| \sim kn$
3. Select an integer c and a vector ξ with $\xi_i \in \{0, 1, 2\}$
4. Find P: P divides $det(2^k I - M)$

How to find convenient P?

How to make a AMNS?

1. Choose $\rho = 2^{k+1}$ with $k = 15, 31, 63$.
2. Define n such that $|P| \sim kn$
3. Select an integer c and a vector ξ with $\xi_i \in \{0, 1, 2\}$
4. Find P: P divides $det(2^k I - M)$

Example

1. $\rho = 2^{16} \rightarrow k = 15$
2. $|P| \sim 160 \rightarrow n = 11$
3. $c = 3$ and $2^k = [1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1]_{\mathcal{B}}$
4. $P = 792412797713126686196656160294175215426473063853$

Conclusions and future directions

What we proposed

- A new modular number system which is adapted to modular arithmetic
- A way to find interesting AMNS
- Fast algorithm for make operations on this AMNS

Conclusions and future directions

What we proposed

- A new modular number system which is adapted to modular arithmetic
- A way to find interesting AMNS
- Fast algorithm for make operations on this AMNS

Perspective

- Find a method to determine γ, ρ for a given P
- Try to generalize this algorithm for all moduli