Hybrid Position-Residues Number System

Karim Bigou and Arnaud Tisserand

CNRS, IRISA, INRIA Centre Rennes - Bretagne Atlantique and Univ. Rennes 1

ARITH 23, July 10 – 13

Context

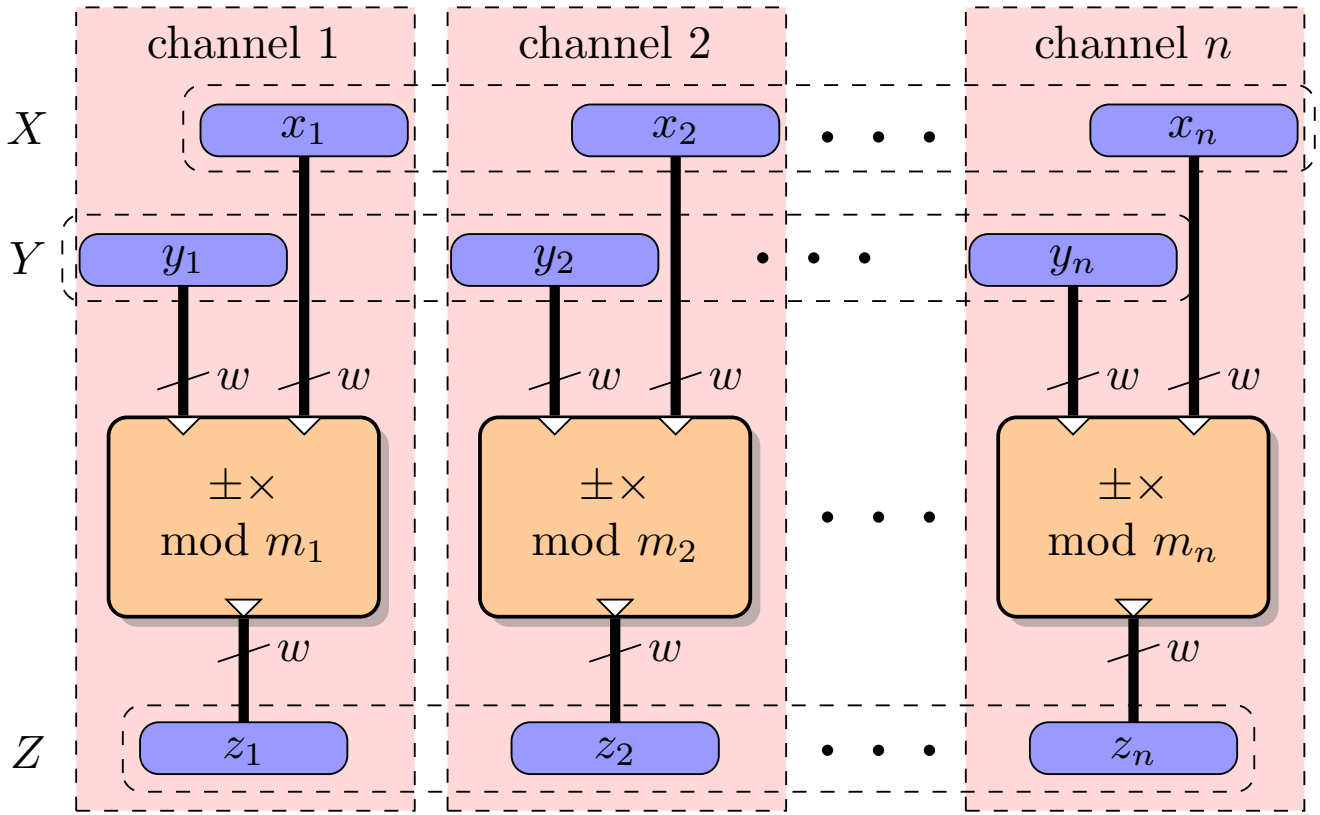
Work on the design of efficient hardware implementations of asymmetric cryptosystems using advanced arithmetic techniques:

- RSA [RSA78]
- Discrete Logarithm Cryptosystems: Diffie-Hellman [DH76] (DH), ElGamal [Elg85]
- Elliptic Curve Cryptography (ECC) [Mil85] [Kob87]

The **residue number system** (RNS) is a representation which enables fast computations for cryptosystems requiring large integers or $\mathbb{F}_P$ elements through internal parallelism

Residue Number System (RNS) [SV55] [Gar59]

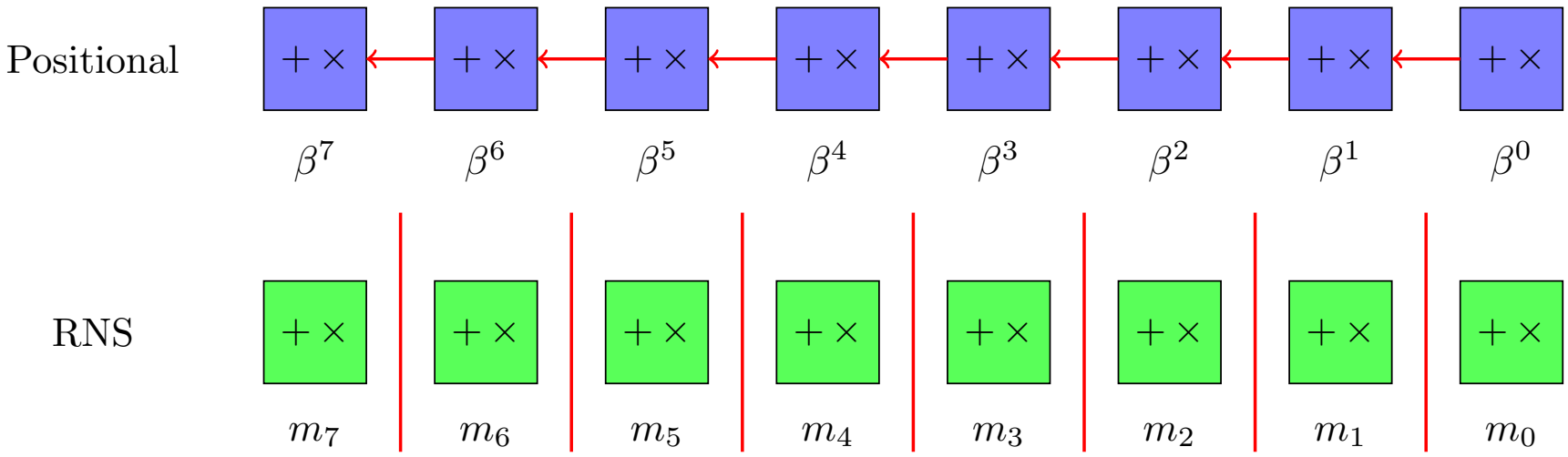
X a large integer of ℓ bits ($\ell > 200$) is represented by:
$\langle X \rangle = (x_1, \ldots, x_n) = (X \bmod m_1, \ldots, X \bmod m_n)$
RNS base $\mathcal{B} = (m_1, \ldots, m_n)$, n pairwise co-prime of w bits, $n \times w \geqslant \ell$



RNS relies on the Chinese remainder theorem (CRT)

EMM = w-bit elementary modular multiplication in one channel

RNS vs Positional Number Systems



$\longleftarrow$ involves data dependencies

| involves hard access to positional information

Remark: here, one assumes a high radix positional representation of w bits

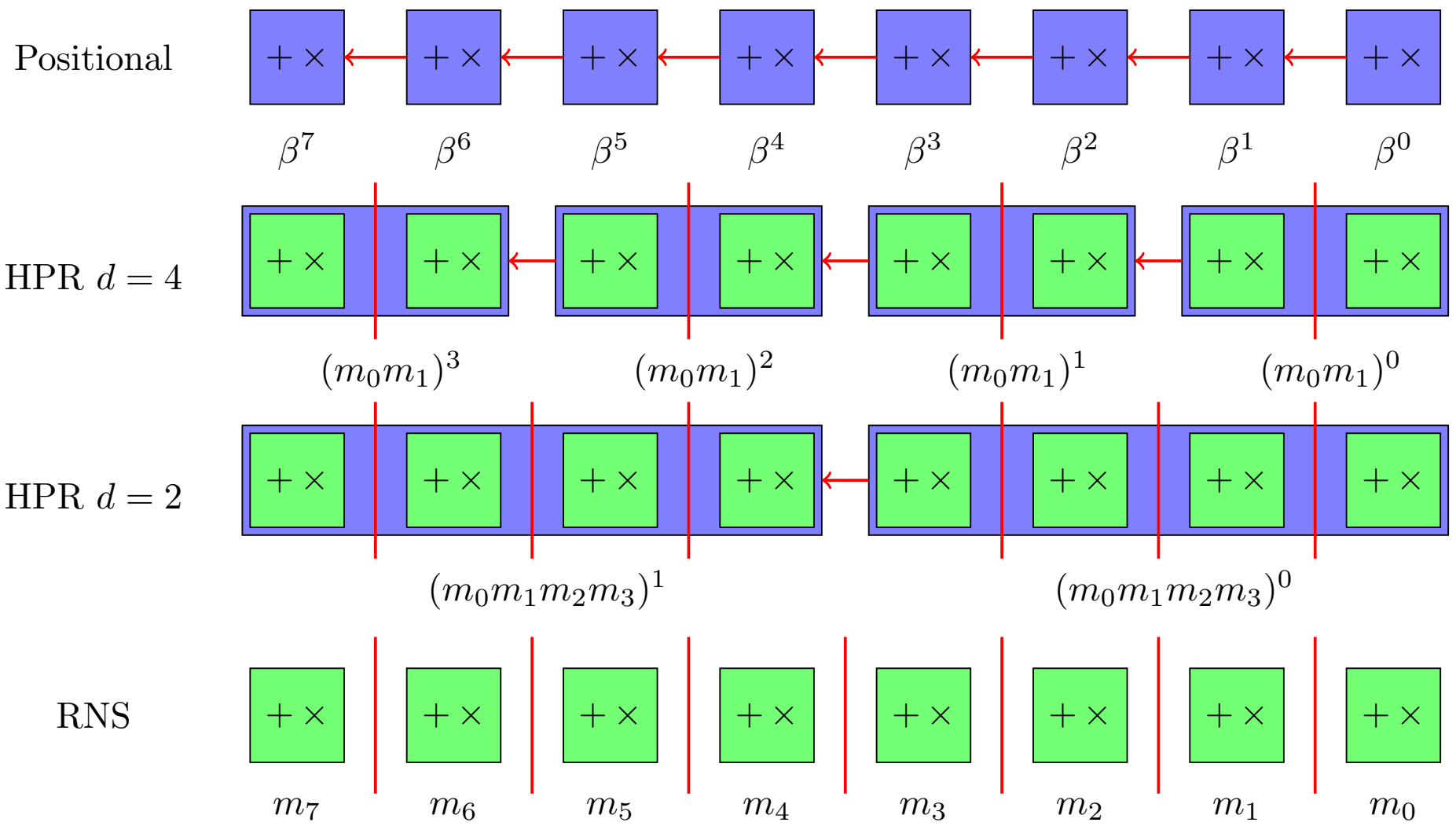
RNS vs Positional Number Systems

operation/feature	RNS	Positional Representation
multiplication	easier	harder
modular reduction	harder	easier
modular multiplication	equivalent	equivalent
expansion of values	harder	easier
comparisons	harder	easier
parallelism	easier	harder
flexibility	easier	harder
internal randomization	easier	harder

Proposed Representation:

Hybrid Position-Residues HPR

Main principle of HPR (finite field case)



Hybrid Position-Residues Representation HPR

Formally:

$$X_{\mathrm{HPR}} = \left(\langle X_{d-1}\rangle_{a|b}, \ldots, \langle X_0\rangle_{a|b}\right)_{\mathrm{HPR}} \quad \text{with} \quad X = \sum_{i=0}^{d-1} X_i \, M_a^i$$

where

$$\mathcal{B}_a = (m_{a,0}, \ldots, m_{a,\frac{n}{d}-1}) \text{ and } \mathcal{B}_b = (m_{b,0}, \ldots, m_{b,\frac{n}{d}-1}),$$

$M_a = \prod_{i=0}^{\frac{n}{d}-1} m_{a,i}$ and $\beta_{min} M_a \leqslant X_i \leqslant \beta_{max} M_a$ $(\beta_{max} - \beta_{min} > 1)$

2 RNS bases are required to contain temporary sub-products of HPR words during a full multiplication

Remark 1: conversions are made using classical methods (radix conversions and RNS conversions)

Remark 2: internal conversions between both RNS bases are made using state-of-the-art base extension methods (*e.g* using CRT)

Example for 1 HPR-word Multiplication (1/2)

Parameters: $\mathcal{B}_a = (2, 7, 13)$, $\mathcal{B}_b = (3, 5, 11)$, $M_a = 182$, $M_b = 165$

Inputs: $X = 141$, $Y = 101$

$X_{\mathrm{HPR}} = \left(\langle 1, 1, 11, 0, 1, 9 \rangle_{a|b}\right)$

$Y_{\mathrm{HPR}} = \left(\langle 1, 3, 10, 2, 1, 2 \rangle_{a|b}\right)$

$X \times Y = 14241$

$$\begin{aligned} X_{\mathrm{HPR}} \times Y_{\mathrm{HPR}} &= \left(\langle 1 \times 1, 1 \times 3, 11 \times 10, 0 \times 2, 1 \times 1, 9 \times 2 \rangle_{a|b}\right) \\ &= \left(\langle 1, 3, 6, 0, 1, 7 \rangle_{a|b}\right) \end{aligned}$$

The high part of the product must be propagated:

$$14241 = 78 \times 182 + 45 = 78 \times M_a + 45$$

Decomposition algorithm (`Split`)

`Split` decomposes a double word value into 2 HPR-words (*i.e* radix M_a)

Input: $\langle X \rangle_{a|b}$ with $X < (\beta_{max} M_a)^2$ and $M_b > \beta_{max}^2 M_a$
Precomp.: $\langle M_a^{-1} \rangle_b$
Output: $(\langle Q \rangle_{a|b}, \langle R \rangle_{a|b})$

$\langle R \rangle_a \leftarrow \langle X \rangle_a$ *(virtual operation)*
$\langle R \rangle_b \leftarrow \mathrm{BE}\,(\langle R \rangle_a, \mathcal{B}_a, \mathcal{B}_b)$ $(n/d) \times (n/d)$ `EMMs`
$\langle Q \rangle_b \leftarrow (\langle X \rangle_b - \langle R \rangle_b) \times \langle M_a^{-1} \rangle_b$
if $\langle Q \rangle_b = \langle -1 \rangle_b$ **then**
  $\langle Q \rangle_b \leftarrow \langle 0 \rangle_b$ */*using Kawamura BE [KKSS00] */*
  $\langle R \rangle_b \leftarrow \langle R \rangle_b - \langle M_a \rangle_b$
$\langle Q \rangle_a \leftarrow \mathrm{BE}\,(\langle Q \rangle_b, \mathcal{B}_b, \mathcal{B}_a)$ $(n/d) \times (n/d)$ `EMMs`
return $\langle Q \rangle_{a|b}$, $\langle R \rangle_{a|b}$

`Split` becomes faster when d increases (but it reduces the parallelism)

"High" part propagation in HPR

This algorithm uses `Split` to propagate the high parts ("MSBs" in radix M_a) of subproducts

Input: $X_{\mathrm{HPR}} = (\langle X_{d-1}\rangle, \ldots, \langle X_0\rangle)$ with $X_i < (\beta_{max} M_a)^2$
Output: $X_{\mathrm{HPR}} = (\langle X_d\rangle, \ldots, \langle X_0\rangle)$ with $X_i < (\beta_{max}^2 + 1) M_a$
$\langle C_{-1}\rangle \leftarrow \langle 0\rangle$, $\langle X_{d-1}\rangle \leftarrow \langle 0\rangle$
for i **from** 0 **to** $d-1$ **parallel do**
  $(\langle C_i\rangle, \langle X_i\rangle) \leftarrow \mathtt{Split}(\langle X_i\rangle)$
for i **from** 0 **to** d **parallel do**
  $\langle X_i\rangle \leftarrow \langle X_i\rangle + \langle C_{i-1}\rangle$
return $(\langle X_d\rangle, \ldots, \langle X_0\rangle)$

Remark: to propagate a carry after an addition, we use a *small carry propagation* algorithm (details in the paper)

Example for 1 HPR-Word Multiplication (2/2)

$$X \times Y = \mathbf{14241} = \mathbf{78} \times 182 + \mathbf{45} = \mathbf{78} \times M_a + \mathbf{45}$$

$$\begin{aligned} X_{\mathrm{HPR}} \times Y_{\mathrm{HPR}} &= \left(\langle 1 \times 1, 1 \times 3, 11 \times 10, 0 \times 2, 1 \times 1, 9 \times 2 \rangle_{a|b}\right) \\ &= \left(\langle \mathbf{1,3,6,0,1,7} \rangle_{a|b}\right) \\ &= \left(\langle \mathbf{0,1,0,0,3,1} \rangle_{a|b}, \langle \mathbf{1,3,6,0,0,1} \rangle_{a|b}\right) \end{aligned}$$

High part propagation

Using BE, convert 45 from $\mathcal{B}_a$ to $\mathcal{B}_b$: $\langle \mathbf{1,3,6} \rangle_a \longrightarrow \langle \mathbf{0,0,1} \rangle_b$

In $\mathcal{B}_b$ perform the division by M_a:

$$\begin{aligned} \frac{\langle XY \rangle_b - \langle |XY|_{M_a} \rangle_b}{\langle M_a \rangle_b} &= \left(\langle \mathbf{0,1,7} \rangle_b - \langle \mathbf{0,0,1} \rangle_b\right) \times \langle 2, 3, 2 \rangle_b \\ &= \left(\langle 0, 1, 6 \rangle_b\right) \times \langle 2, 3, 2 \rangle_b \\ &= \langle \mathbf{0,3,1} \rangle_b \end{aligned}$$

Finally one performs another BE from $\mathcal{B}_b$ to $\mathcal{B}_a$: $\langle \mathbf{0,3,1} \rangle_b \longrightarrow \langle \mathbf{0,1,0} \rangle_a$

Application 1:
A New Modular Multiplication Algorithm

Principle

Proposition:well-chosen finite fields $\mathbb{F}_P$ for fast modular multiplications

- example of application: finite field for ECC

P prime with $P = Q(M_a)$ and $Q(X) = X^d - Q'(X)$ where Q' is sparse

- $\mathbb{F}_P$ is a $d \times (n/d) \times w = nw$ bits finite field
- $M_a^d \equiv Q'(M_a) \bmod P$
- toy example 1: $P_1 = (2 \times 7 \times 13)^2 - 5 = M_a^2 - 5 = 33119$ is prime
- toy example 2: $P_2 = (3 \times 5 \times 11)^3 - 2 = M_b^3 - 2 = 27225$ is prime

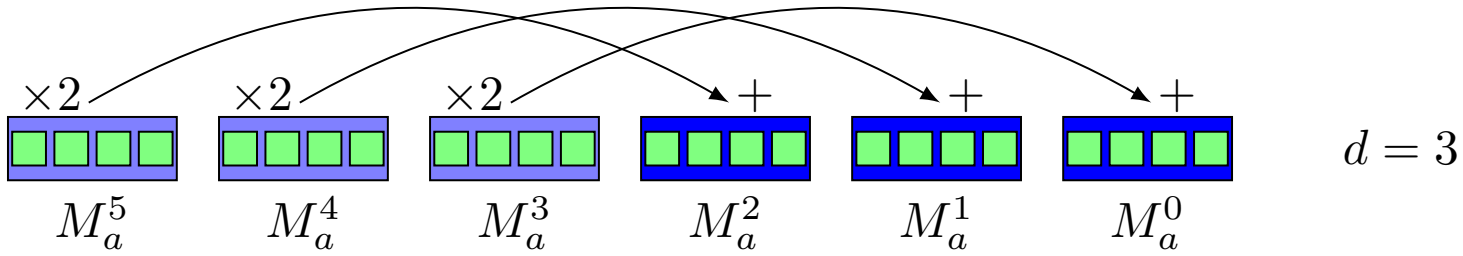
Main Idea:

Adapt pseudo-Mersenne modular multiplication for HPR representation

HPR Modular Multiplication

Positional Modular Reduction: reduction using $M_a^d \equiv Q'(M_a) \bmod P$

- example: $Q' = 2$ then $Z_i = Z_i + 2Z_{i+d}$ for $i \in [0, d-1]$



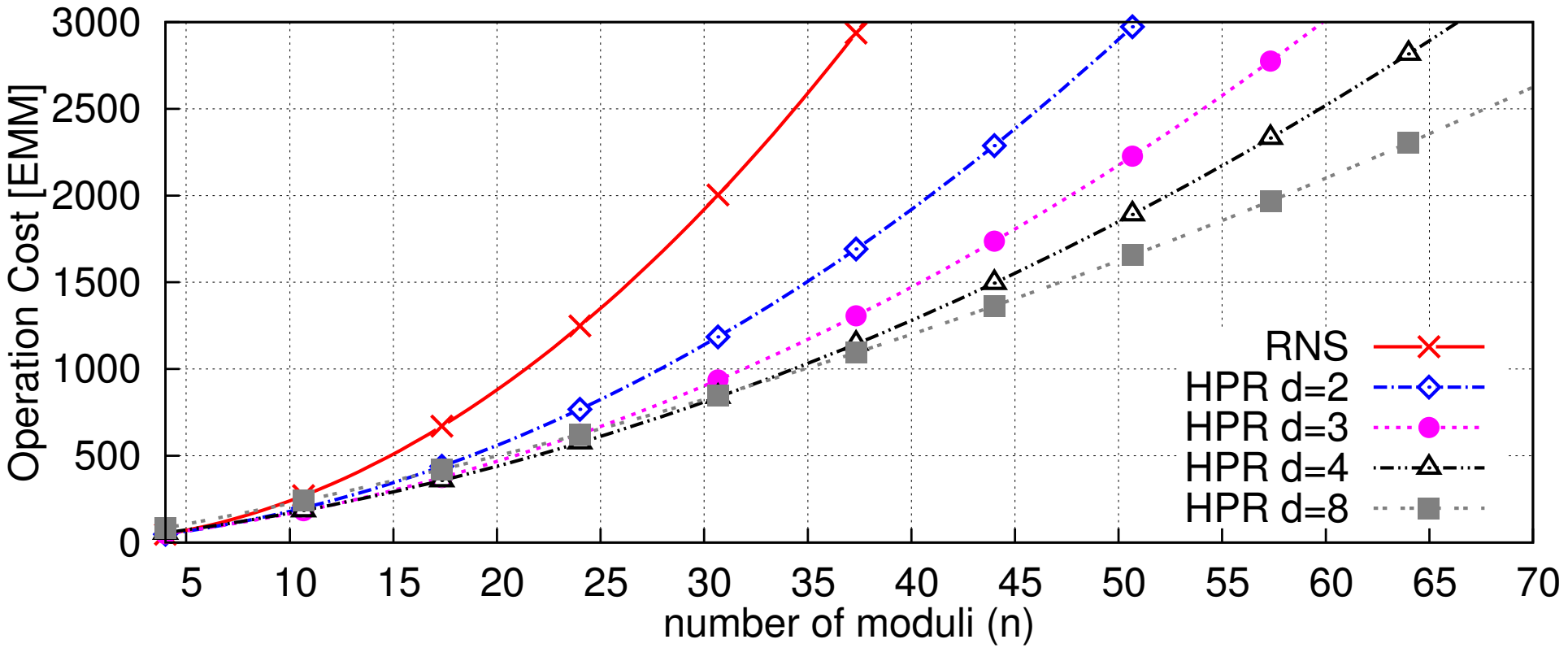
Parameters: $\mathcal{B}_a$ with $P = Q(M_a)$ and Q of degree d
Input: X_{HPR}, Y_{HPR}
Output: Z_{HPR} with $Z = XY \bmod P$

$Z_{\mathrm{HPR}} \leftarrow$ **HPR Product**$(X_{\mathrm{HPR}}, Y_{\mathrm{HPR}})$ $\quad d^2(n/d) =$ $2nd$ EMMs
$Z_{\mathrm{HPR}} \leftarrow$ Positional Modular Reduction$(Z_{\mathrm{HPR}},\ Q)$ $\quad (n$ EMAs$)$
$Z_{\mathrm{HPR}} \leftarrow$ **HPR "High" Part Propagation** (Z_{HPR}) $\quad 2\frac{n^2}{d} + 2n$ EMMs
$Z_{\mathrm{HPR}} \leftarrow$ Positional Modular Reduction$(Z_{\mathrm{HPR}},\ Q)$ $\quad (n/d$ EMAs$)$
$Z_{\mathrm{HPR}} \leftarrow$ HPR Small Carry Propagation (Z_{HPR}) $\quad 2n$ EMMs
return Z_{HPR}

Cost of modular multiplication in RNS and HPR for various fixed d

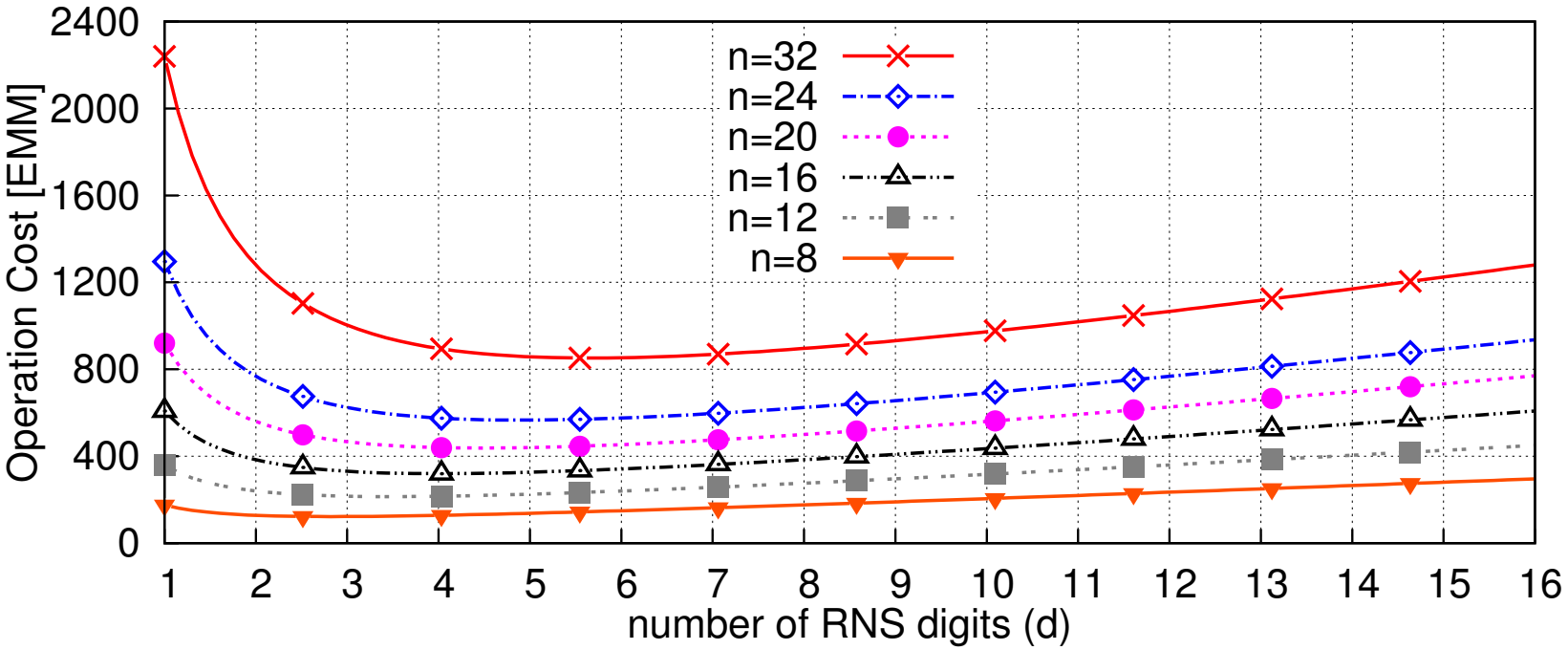
Operation cost:

- trade-off between HPR product and HPR High part propagation



Impact of d for n and the field size fixed

Using schoolbook multiplication, $d = \sqrt{n}$ is the best trade-off



d	2	4	8	16
cost (EMM)	$n^2 + 8n$	$\frac{n^2}{2} + 12n$	$\frac{n^2}{4} + 20n$	$\frac{n^2}{8} + 36n$

Sources of Parallelism

Two main sources of parallelism:

- parallelism due to digits in RNS: decreases while d increases
- parallelism due to HPR algorithms

Assuming n hardware channels (as in usual RNS architectures):

- High Part propagation: $2\frac{n}{d} + 2$ `EMMs` by parallel channel
- HPR multiplication (schoolbook): $2d$ `EMMs` by parallel channel ...
- ... but number additions increases with d, increasing dependencies between the sub-products

Summary:

- when d is small our algorithm is as parallel as RNS algorithms
- in practice, d is small ($\approx \sqrt{n}$)

Application 2:
A New Exponentiation Algorithm

Application 2: Modular Exponentiation

Idea: take benefit of positional information of HPR to accelerate some specific, but usual computation patterns

Input: $k = (k_{\ell-1}, \ldots, k_1, k_0)_2$, $G \in \mathbb{Z}/N\mathbb{Z}$
Output: $G^k \bmod N$
$Z \leftarrow 1$
for i **from** $\ell - 1$ **to** 0 **do**
  $Z \leftarrow Z^2 \bmod N$
  if $k_i = 1$ **then** $S \leftarrow Z \cdot G \bmod N$
return Z

One can observe:

$$\begin{aligned} Z^2 G &\equiv \left(Z_1^2 M_a^2 + 2 Z_1 Z_0 M_a + Z_0^2\right) G \bmod N \\ &\equiv Z_1^2 |M_a^2 G|_N + Z_1 Z_0 |2 M_a G|_N + Z_0^2 |G|_N \bmod N \\ &\equiv Z_1 \left(Z_1 |M_a^2 G|_N + Z_0 |2 M_a G|_N\right) + Z_0^2 |G|_N \bmod N \end{aligned}$$

Proposed Regular Modular Exponentiation

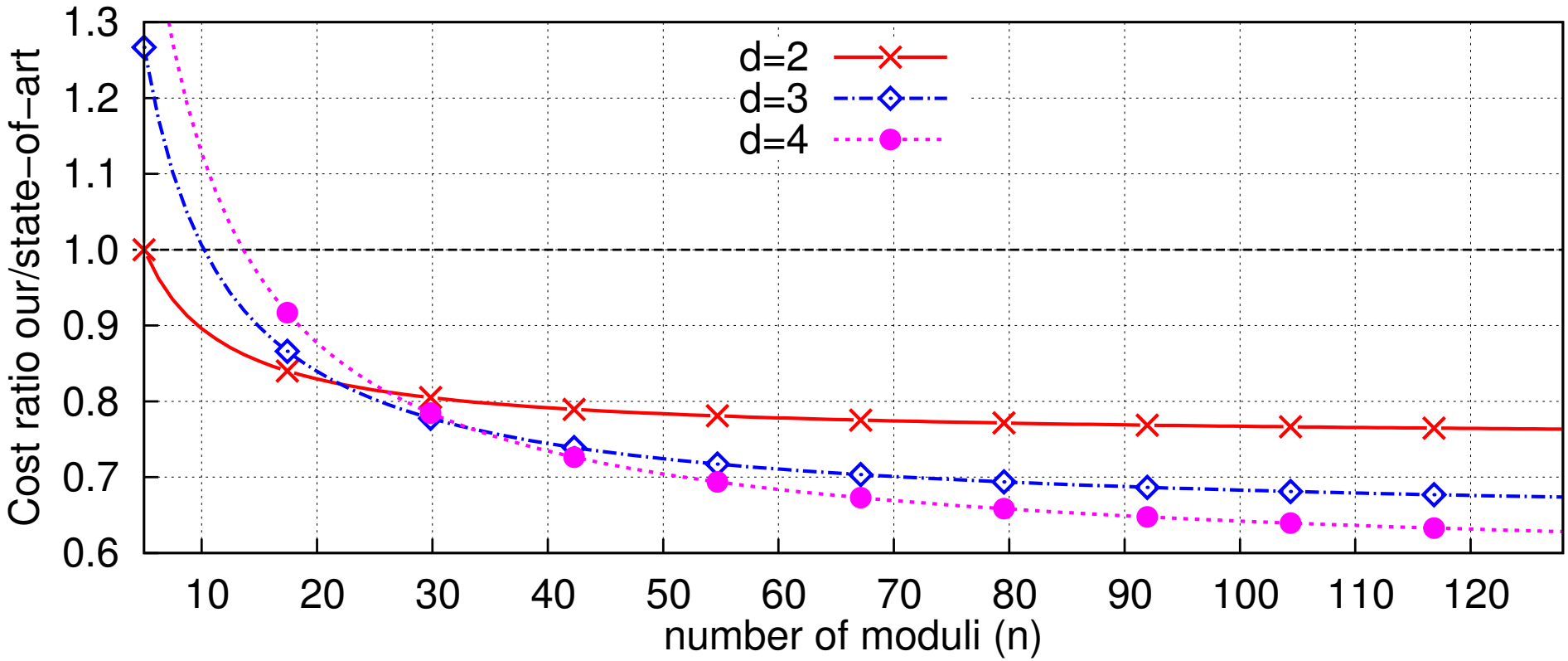
For a general d : $|Z^2 G|_N \equiv \sum_{k=0}^{2(d-1)} \sum_{i=0}^{d-1} Z_i Z_{k-i} |M_a^k G|_N$

Parameters: $\mathcal{B}_a, \mathcal{B}_b, \mathcal{B}_c$ with $n_a = n_b = n/d$ and $n_c = n$
Input: $\langle G \rangle_{a|b|c}$, e the exponent
Output: $\langle Z \rangle_{a|b|c}$ with $Z = G^e \bmod N$, $Z < 3P$

$\langle Z \rangle_{a|b|c} \leftarrow \langle |M_{a|b}|_P \rangle_{a|b|c}$
for i **from** $\ell - 1$ **to** 0 **do**
  $Z_{\mathrm{HPR}} \leftarrow$ **RNStoHPR**$(\langle Z \rangle_{a|b|c}, \mathcal{B}_a, \mathcal{B}_b, \mathcal{B}_c)$ $O(n^2)$
  if $e_i = 0$ **then**
    $\langle Z \rangle_{a|b|c} \leftarrow$ **SubProducts**$(Z_{\mathrm{HPR}}, \langle |M_{a|b}|_P \rangle)$ $O(d^2)$
    $\langle Z \rangle_{a|b|c} \leftarrow$ **RNS-MR**$(\langle Z \rangle_{a|b|c}, \mathcal{B}_{a|b}, \mathcal{B}_c)$ $O(n^2)$
  else
    $\langle Z \rangle_{a|b|c} \leftarrow$ **SubProducts**$(Z_{\mathrm{HPR}}, \langle G \rangle)$ $O(d^2)$
    $\langle Z \rangle_{a|b|c} \leftarrow$ **RNS-MR**$(\langle Z \rangle_{a|b|c}, \mathcal{B}_{a|b}, \mathcal{B}_c)$ $O(n^2)$
return $\langle Z \rangle_{a|b|c}$

Remark: conversions HPR to RNS are implicits

Comparison with state-of-the-art RNS exponentiation



Conclusion

HPR Conclusion and Further Work

The representation HPR

- reduces the cost of some RNS modular arithmetic algorithms with a high level of parallelism (for small d)
- enables to use positional properties (as the extensibility of the representation) or tricks (as pseudo-Mersenne like numbers)
- provides more flexibility with a lot of new trade-off possibilities

Examples of applications:

- modular multiplication: HPR offers a reduction of computation cost of 40 to 60% reduction (for ECC 256 – 512)
- modular multiplication: HPR offers a reduction of computation cost of 20 to 40% (for RSA 2048 – 4096)

Further works:

- adaptation of other usual arithmetic algorithms (e.g. Montgomery or Barrett Reduction Algorithms)
- application to very large values (e.g. homomorphic encryption)

Thank you for your attention

References I

[DH76] W. Diffie and M. E. Hellman.
New directions in cryptography.
IEEE Transactions on Information Theory, 22(6):644–654, November 1976.

[Elg85] T. Elgamal.
A public key cryptosystem and a signature scheme based on discrete logarithms.
IEEE Transactions on Information Theory, 31(4):469–472, July 1985.

[Gar59] H. L. Garner.
The residue number system.
IRE Transactions on Electronic Computers, EC-8(2):140–147, June 1959.

[KKSS00] S. Kawamura, M. Koike, F. Sano, and A. Shimbo.
Cox-Rower architecture for fast parallel Montgomery multiplication.
In *Proc. 19th International Conference on the Theory and Application of Cryptographic (EUROCRYPT)*, volume 1807 of *LNCS*, pages 523–538. Springer, May 2000.

[Kob87] N. Koblitz.
Elliptic curve cryptosystems.
Mathematics of computation, 48(177):203–209, 1987.

References II

[Mil85] V. Miller.
Use of elliptic curves in cryptography.
In *Proc. 5th International Cryptology Conference (CRYPTO)*, volume 218 of *LNCS*, pages 417–426. Springer, 1985.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman.
A method for obtaining digital signatures and public-key cryptosystems.
Communications of the ACM, 21(2):120–126, February 1978.

[SV55] A. Svoboda and M. Valach.
Operátorové obvody (operator circuits in czech).
Stroje na Zpracování Informací (Information Processing Machines), 3:247–296, 1955.

Small Carry Propagation

Input: X_{HPR} with $X_i < (m_\gamma - 2)M_a \ \forall i \in [0, d-1]$
Parameters: Q' such that $P = Q(M_a)$ with $Q = X^d - Q'$
Precomp.: $\left|M_a^{-1}\right|_{m_\gamma}$
Output: X_{HPR}, $X_i < 2M_a + (m_\gamma - 2) \quad \forall i \in [0, d-1]$

for i **from** 0 **to** $d-1$ **do**
  $|R_i|_{m_\gamma} \leftarrow \mathrm{BE}\left(\langle X_i \rangle_a, \mathcal{B}_a, m_\gamma\right)$
  $|C_i|_{m_\gamma} \leftarrow \left|(X_i - R_i)M_a^{-1}\right|_{m_\gamma}$
  if $|C_i|_{m_\gamma} = m_\gamma - 1$ **then** $|C_i|_{m_\gamma} = 0$
  $\langle X_i \rangle_b \leftarrow \langle X_i \rangle_b - |C_{i,H}|_{m_\gamma} \times \langle M_a \rangle_b$
for i **from** 1 **to** $d-1$ **parallel do**
  $\langle X_i \rangle_{a|b} \leftarrow \langle X_i \rangle_{a|b} + \langle C_{i-1} \rangle_{a|b}$
return X_{HPR}