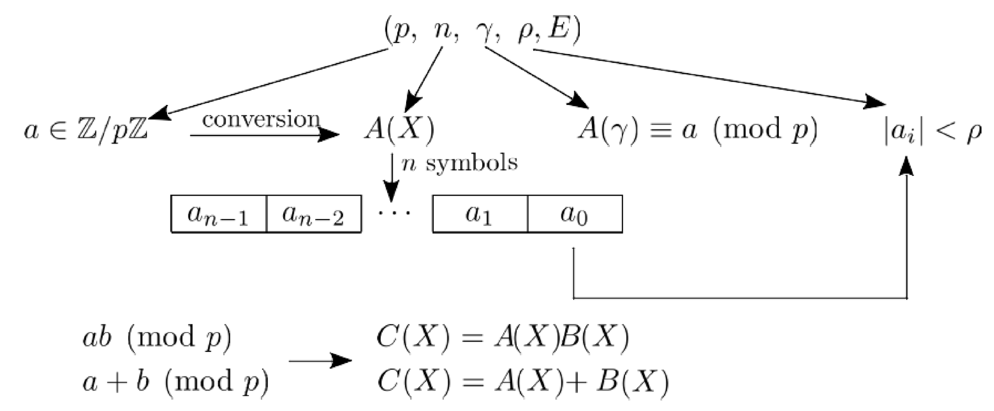
PMNS for Large Integers: Revisiting the internal Montgomery reduction

Nicolas Méloni, François Palma, Pascal Véron
Université de Toulon, Laboratoire IMATH

Definition

Polynomial Modular Number Systems (PMNS)[1] are used for fast and efficient modular integer operations. They improve on generic multiprecision implementations by bypassing the need for any carry propagation. They are thus relevant for any cryptographic system that relies on modular arithmetic such as RSA, ECC, etc. A PMNS is characterized by a tuple $(p,\ n,\ \gamma,\ \rho,\ E)$:



$E \in \mathbb{Z}[X]$ a monic polynomial with $deg(E) = n$ and $E(\gamma) \equiv 0 \pmod p$, used for External Reduction.

Example

A short example using small numbers for clarity

- We take the PMNS $(11, 3, 7, 2, X^3 - 2)$
  - $a = 8 \to A(X) = X^2 - X - 1:\ A(7) \equiv 7^2 - 7 - 1 \equiv 41 \equiv 8 \pmod{11}$
  - $deg(A) < 3$ and $\forall i, |a_i| < 2$
  - $b = 6 \to B(X) = X - 1,\ c = ab \pmod{11} \to C(X) = A(X)B(X)$
    - $C(X) = (X^2 - X - 1)(X - 1) = X^3 - 2X^2 + 1$
- $E(\gamma) \equiv E(7) \equiv 7^3 - 2 \equiv 341 \equiv 0 \pmod{11}$

Problems:

- If $deg(A) = deg(B) = n - 1$, for $C(X) = A(X)B(X)$ we have $deg(C) = 2n - 2 > n - 1$
- $c_i = a_i b_i \implies |c_i| < n\rho^2$ and not ρ so some coefficients might be too big.

- $deg(C) = 3 > 2$
- $c_2 = -2 \implies |c_2| = 2$ and we want $\forall i, |c_i| < 2$

External Reduction

- We compute $C'(X) = A(X)B(X) \pmod{E(X)}$
- $E(\gamma) \equiv 0 \pmod p \implies C'(\gamma) \equiv A(\gamma)B(\gamma) \equiv ab \pmod p$
- $deg(E) = n \implies deg(C') < n$

Since E can be as sparse as we want it to be, we can choose $E(X) = X^n - \lambda$ with $\lambda \in \mathbb{Z}/p\mathbb{Z}$ for fast calculations.
We therefore have a new bound of $|c'_i| < (|\lambda|(n-1)+1)\rho^2$
$C'(X) = C(X) - E(X) = X^3 - 2X^2 + 1 - (X^3 - 2) = -2X^2 + 3$

Internal Reduction

- We define $\mathfrak{L} = \{A(X) \text{ such that } \deg A(X) \leqslant n-1 \text{ and } A(\gamma) = 0 \bmod p\}$
- We choose $T(X)$ in $\mathfrak{L}$ and compute $C''(X) = C'(X) - T(X)$
  - We still have $C''(\gamma) \equiv C'(\gamma) \equiv ab \pmod p$ because $T(\gamma) \equiv 0 \pmod p$

We choose T to guarantee $|c''_i| < \rho$. Thus the resulting C'' is within our PMNS.
$T(X) = -2X^2 + X + 3.\ T(7) \equiv -2 \times 49 + 7 + 3 \equiv -88 \equiv 0 \pmod{11}$
$C''(X) = C'(X) - T(X) = -2X^2 + 3 - (-2X^2 + X + 3) = -X$
$C''(7) \equiv -7 \equiv 4 \pmod{11}$ and $8 \times 6 \equiv 48 \equiv 4 \pmod{11}$ and $|c''_i| < 2$

The Internal Reduction is one of the most important parts of PMNS calculations. In 2008, C. Negre and T. Plantard proposed an algorithm based on the Montgomery modular reduction.[3]

Montgomery Polynomial Reduction

Algorithm 1 Coefficients reduction
Require: $\mathcal{B} = (p, n, \gamma, \rho, E)$ a PMNS, $C' \in \mathbb{Z}[X]$ such that $deg(C') < n$, $M \in \mathfrak{L} = \{A(X) \text{ such that } \deg A(X) \leqslant n-1 \text{ and } A(\gamma) = 0 \bmod p\}$, $\phi \in \mathbb{N} \setminus \{0\}$ and $M' = M^{-1} \bmod (E, \phi)$.
Ensure: $C''(\gamma) = C'(\gamma)\phi^{-1} \pmod p$, with $C'' \in \mathbb{Z}[X]$ such that $deg(C'') < n$
$Q \leftarrow C' \times M' \bmod (E,\ \phi)$
$T \leftarrow Q \times M \bmod E$
$C'' \leftarrow (C' - T)/\phi$
return C''

Notes:

- We generally choose $\phi = 2^{64}$ for fast division operations on 64-bit hardware.
- For $E(X) = X^n - \lambda$, we choose ρ such that $\phi > (|\lambda|(n-1)+1)\rho$
  - As noted earlier, we have $|c'_i| < (|\lambda|(n-1)+1)\rho^2$ therefore $|c''_i| < \frac{(|\lambda|(n-1)+1)\rho^2}{\phi} \implies |c''_i| < \rho$
- We get $c'' = ab\phi^{-1} \pmod p$ and not ab which has to be accounted for.

Problem:

- Finding M may involve an exhaustive search (2^n iterations [2, Algorithm 8])
- $n_{min} > log_2(\frac{p}{\phi})$ so for 8192-bit integers (ex: RSA) we get $n > 128$
  - More than 2^{128} iterations isn't reasonable

Novel Montgomery Lattice Reduction

Algorithm 2 Coefficients reduction, new version
Require: $\mathcal{B} = (p, n, \gamma, \rho, E)$ a PMNS, $C' \in \mathbb{Z}[X]$ such that $deg(C') < n$, $\mathcal{L}$ a reduced basis of $\mathfrak{L} = \{A(X) \text{ such that } \deg A(X) \leqslant n-1 \text{ and } A(\gamma) = 0 \bmod p\}$, $\phi \in \mathbb{N} \setminus \{0\}$ and $\mathcal{L}^{-1} \bmod \phi$.
Ensure: $C''(\gamma) \equiv C'(\gamma)\phi^{-1} \bmod p$, with $C'' \in \mathbb{Z}[X]$ such that $deg(C'') < n$
1: $q \leftarrow C'(\mathcal{L}^{-1}) \bmod \phi$
2: $C'' \leftarrow (C' - q\mathcal{L})/\phi$
3: return C''

As this new method works directly with a basis matrix, no further generation is needed and the step in $O(2^n)$ is completely bypassed. To represent all the elements of $\mathbb{Z}/p\mathbb{Z}$ we have the following restriction on $\mathcal{L}$:

$$\|\mathcal{L}\|_1 \geqslant p^{1/n}$$

Furthermore we have the following bound on $\mathcal{L}$ to guarantee the coefficient reduction:

$$\frac{\phi}{4(|\lambda|(n-1)+1)} \geqslant \|\mathcal{L}\|_1$$

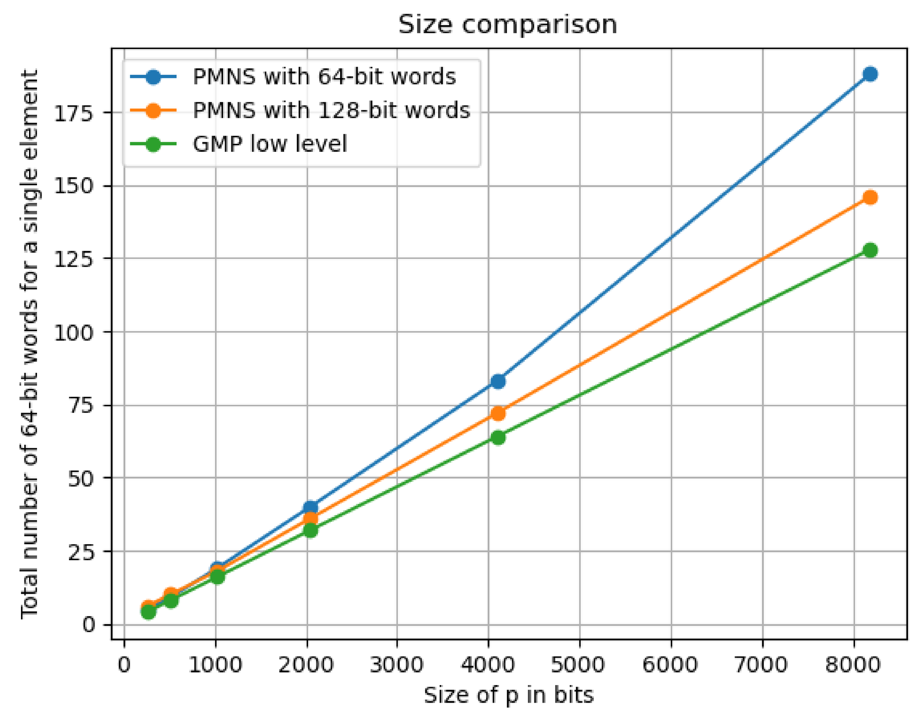
Note that the bounds for the new method are slightly better than the old one. However, even with the new bounds, for 8192-bit integers if we pick a minimal n of 129 with $\phi = 2^{64}$ and $E(X) = X^n - 2$ we get:

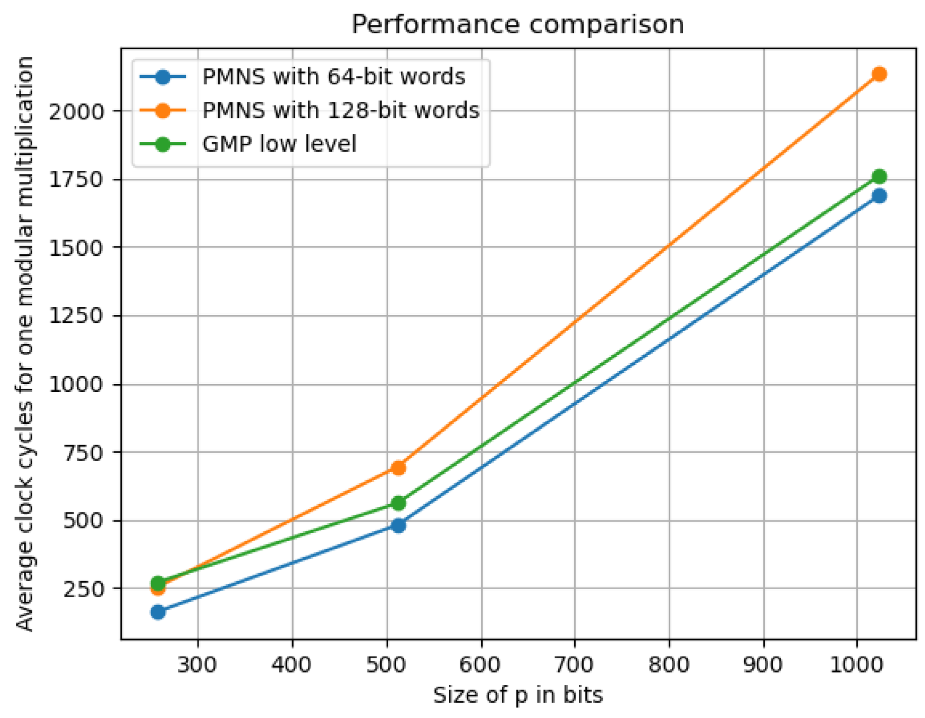
$$\frac{2^{64}}{4(2(129-1)+1)} \approx 2^{55} \geqslant \|\mathcal{L}\|_1 \geqslant p^{1/129} \approx 2^{63}$$

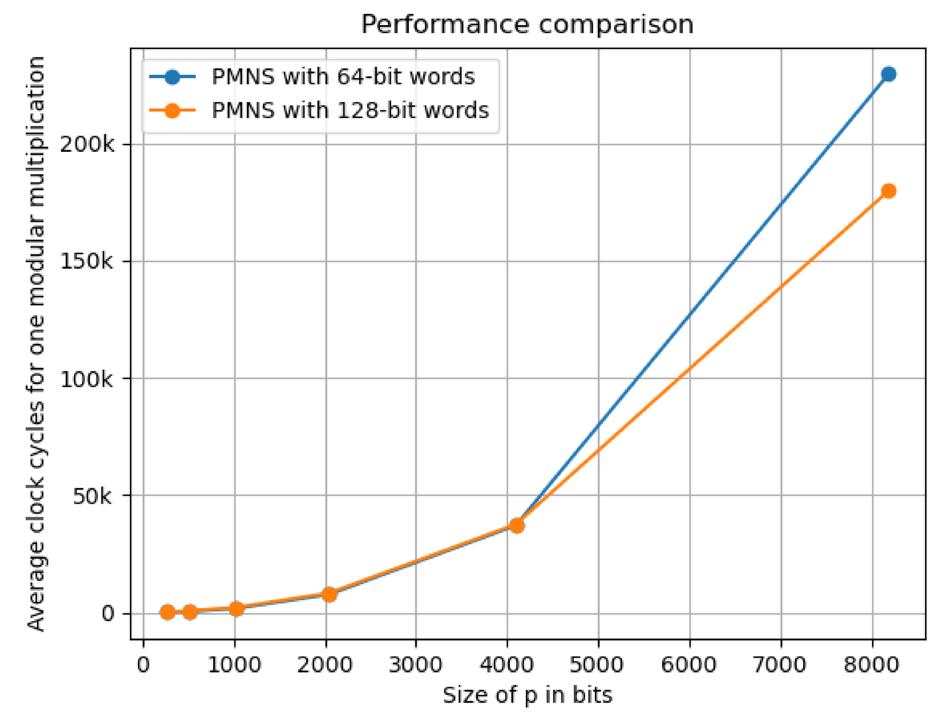
It is easy to see n is too small and we will in practice need $n = 188$ for this integer size which dramatically affects performance.

Optimizing PMNS for large integer use with 128-bit hardware words

Even though current CPUs operate on 64-bit logic, compilers like GCC offer the possibility of using 128-bit variables. As such a possible solution to reduce n is to use $\phi = 2^{128}$. This leads to the following results.







As GMP uses subquadratic algorithms for higher limb counts and our current implementation doesn't, a noticeable gap in favour of GMP appears for greater integer sizes (>2048 bits). Our next step is to adopt such methods as well.

References

[1] J.-C. Bajard, Laurent Imbert, and Thomas Plantard. Modular number systems: Beyond the mersenne family. In *Selected Areas in Cryptography, 11th International Workshop, SAC 2004, Waterloo, Canada*, pages 159–169, 2004.
[2] Fangan Yssouf Dosso, Jean-Marc Robert, and Pascal Veron. PMNS for Efficient Arithmetic and Small Memory Cost. *IEEE Transactions on Emerging Topics in Computing*, 10(3):1263 – 1277, July 2022.
[3] Christophe Negre and Thomas Plantard. Efficient modular arithmetic in adapted modular number system using lagrange representation. In *Information Security and Privacy, 13th Australasian Conference, ACISP 2008, Wollongong, Australia*, pages 463–477, 2008.