Représentations adaptées à l'arithmétique modulaire et à la résolution de systèmes flous

Soutenance de thèse de Jérémy Marrez

Dirigée par Jean-Claude Bajard
et co-encadrée par Lokmane Abbas-Turki

EDITE de Paris (ED 130)
Sorbonne Université

6 décembre 2019

Présentée devant le jury composé de

Marine Minier, Université de Lorraine (rapporteur)
Clément Pernet, Université Grenoble Alpes (rapporteur)
Louis Goubin, UVSQ
Annick Valibouze, Sorbonne Université
Jean-Claude Bajard, Sorbonne Université
Lokmane Abbas-Turki, Sorbonne Université

Contexte : arithmétique modulaire en cryptographie asymétrique

- Exponentiations modulaires présentes dans la plupart des cryptosystèmes asymétriques : calcul du reste dans la division euclidienne de g^e par un *modulo* p connu

$$g^e \mod p$$

- ➥ multiplications modulaires : calcul du produit ab, suivi d'une *réduction modulaire*

$$r = ab + qp, \quad \text{avec } r < p.$$

Modulo dans les cryptosystèmes asymétriques

- Modulo fixé par le protocole
  - RSA introduit par Rivest, Shamir et Adelman en 1978 : modulo de taille 1024 bits au moins
- Modulo premier, standardisé dans les applications cryptographiques
  - Protocoles d'échange des clés Diffie Hellman en 1976 et chiffrement ElGamal en 1984 : modulo de taille 1024 bits au moins
  - Cryptographie sur les courbes elliptiques en 1985 (Koblitz) : modulo de taille 160 bits au moins.
    L'exponentiation devient une multiplication : k scalaire, P générateur du groupe des points à coordonnées dans $\mathbb{Z}/p\mathbb{Z}$: kP

Premières avancées pour les implantations des protocoles d'ECC qui imposent la taille du modulo p, non son type

➤ Classes de nombres adaptées à l'arithmétique modulaire proposées dès 1981

$$\boxed{\begin{array}{c}\text{Mersenne}\\ p = 2^n - 1\end{array}}$$

Knuth, 1981

→ division par une puissance de 2 réduite à un décalage de bits

✗ mersenne premiers rares : pas de premiers pour les tailles cryptographiques qui nous intéressent

Premières avancées pour les implantations des protocoles d'ECC qui imposent la taille du modulo p, non son type

- ➤ Classes de nombres adaptées à l'arithmétique modulaire proposées dès 1981

Mersenne
$p = 2^n - 1$

$\swarrow$ Knuth, 1981

Pseudo Mersenne
$p = 2^n - c$

Crandall, 1992

→ division par une puissance de 2 réduite à un décalage de bits

✗ mersenne premiers rares : pas de premiers pour les tailles cryptographiques qui nous intéressent

→ conseillée pour de petits moduli

✗ plusieurs multiplications ajoutées avec un surcoût non négligeable

Premières avancées pour les implantations des protocoles d'ECC qui imposent la taille du modulo p, non son type

➤ Classes de nombres adaptées à l'arithmétique modulaire proposées dès 1981

Mersenne
$p = 2^n - 1$
Knuth, 1981

- ➞ division par une puissance de 2 réduite à un décalage de bits
- ✗ mersenne premiers rares : pas de premiers pour les tailles cryptographiques qui nous intéressent

Pseudo Mersenne
$p = 2^n - c$
Crandall, 1992

- ➞ conseillée pour de petits moduli
- ✗ plusieurs multiplications ajoutées avec un surcoût non négligeable

Mersenne Généralisé
$p = P(2^k),\ \|P\|_\infty = 1$
Solinas, 1999

- ➞ pas de multiplication supplémentaire sur la forme polynomiale

formes plus adaptées au calcul sur les grands nombres

Premières avancées pour les implantations des protocoles d'ECC qui imposent la taille du modulo p, non son type

- Classes de nombres adaptées à l'arithmétique modulaire proposées dès 1981

Mersenne
$p = 2^n - 1$
Knuth, 1981

→ division par une puissance de 2 réduite à un décalage de bits

✗ mersenne premiers rares : pas de premiers pour les tailles cryptographiques qui nous intéressent

Pseudo Mersenne
$p = 2^n - c$
Crandall, 1992

→ conseillée pour de petits moduli

✗ plusieurs multiplications ajoutées avec un surcoût non négligeable

Mersenne Généralisé
$p = P(2^k),\ \|P\|_\infty = 1$
Solinas, 1999

→ pas de multiplication supplémentaire sur la forme polynomiale

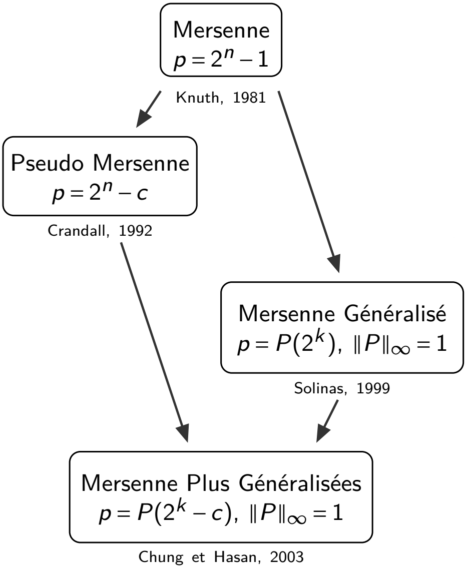
formes plus adaptées au calcul sur les grands nombres

Mersenne Plus Généralisées
$p = P(2^k - c),\ \|P\|_\infty = 1$
Chung et Hasan, 2003

→ fusion des deux classes précédentes

Premières avancées pour les implantations des protocoles d'ECC qui imposent la taille du modulo p, non son type

- Classes de nombres adaptées à l'arithmétique modulaire proposées dès 1981



Système RNS

- ➥ implémentations de l'addition et de la multiplication plus efficaces
- ✗ nature non positionnelle : réductions modulaires (exigées par ECC) plus coûteuses

Premières avancées pour les implantations des protocoles d'ECC qui imposent la taille du modulo p, non son type

- ➤ Classes de nombres adaptées à l'arithmétique modulaire proposées dès 1981

Mersenne
$p = 2^n - 1$
Knuth, 1981

Pseudo Mersenne
$p = 2^n - c$
Crandall, 1992

Mersenne Généralisé
$p = P(2^k), \ \|P\|_\infty = 1$
Solinas, 1999

Mersenne Plus Généralisées
$p = P(2^k - c), \ \|P\|_\infty = 1$
Chung et Hasan, 2003

Système RNS

➥ implémentations de l'addition et de la multiplication plus efficaces

✗ nature non positionnelle : réductions modulaires (exigées par ECC) plus coûteuses

Système HPR
Représentation positionnelle
+ coefficients représentés dans RNS
Bigou et Tisserand, 2016

➥ bases RNS plus petites : réductions modulaires moins coûteuses. Indépendance arithmétique des canaux RNS.

✗ uniquement pour des premiers $P = B_1^n - \beta$: pas extensible aux opérations des courbes elliptiques actuellement standardisées.

Premières avancées pour les implantations des protocoles d'ECC qui imposent la taille du modulo p, non son type

➤ Classes de nombres adaptées à l'arithmétique modulaire proposées dès 1981

Mersenne
$p = 2^n - 1$

Knuth, 1981

Pseudo Mersenne
$p = 2^n - c$

Crandall, 1992

Mersenne Généralisé
$p = P(2^k), \ \|P\|_\infty = 1$

Solinas, 1999

Mersenne Plus Généralisées
$p = P(2^k - c), \ \|P\|_\infty = 1$

Chung et Hasan, 2003

Système RNS

- ➥ implémentations de l'addition et de la multiplication plus efficaces
- ✗ nature non positionnelle : réductions modulaires (exigées par ECC) plus coûteuses

Système HPR
Représentation positionnelle
+ coefficients représentés dans RNS

Bigou et Tisserand, 2016

- ➥ bases RNS plus petites : réductions modulaires moins coûteuses. Indépendance arithmétique des canaux RNS.
- ✗ uniquement pour des premiers $P = B_1^n - \beta$: pas extensible aux opérations des courbes elliptiques actuellement standardisées.

Principal inconvénient de ces approches

Ne garantissent une réduction efficace que sur une classe restreinte de moduli avec des propriétés spécifiques.

➤ Le choix du modulo n'est pas possible.

Protocoles imposent certaines propriétés sur le modulo

➤ utiliser des algorithmes fonctionnant pour tout les moduli

Algorithmes généralistes

Réduction intégrée à la multiplication	Réduction indépendante de la multiplication
Taylor 1981	Montgomery 1986
Blakley 1983	Barrett 1987
Takagi 1986	PMNS 2004
➥ adapté aux petits moduli	➥ + efficace sur de grands moduli

Protocoles imposent certaines propriétés sur le modulo

➤ utiliser des algorithmes fonctionnant pour tout les moduli

Algorithmes généralistes

Réduction intégrée à la multiplication

- Taylor — 1981
- Blakley — 1983
- Takagi — 1986

➥ adapté aux petits moduli

Réduction indépendante de la multiplication

- Montgomery — 1986
- Barrett — 1987
- PMNS — 2004

➥ + efficace sur de grands moduli

→ réduction réduite à une division par une puissance de la base : décalage de bits

✗ représentation particulière des nombres, nécessite des conversions

Protocoles imposent certaines propriétés sur le modulo

➤ utiliser des algorithmes fonctionnant pour tout les moduli

Algorithmes généralistes

Réduction intégrée à la multiplication

- Taylor — 1981
- Blakley — 1983
- Takagi — 1986

➥ adapté aux petits moduli

Réduction indépendante de la multiplication

- Montgomery — 1986
- Barrett — 1987
- PMNS — 2004

➥ + efficace sur de grands moduli

→ réduction réduite à une division par une puissance de la base : décalage de bits

✗ représentation particulière des nombres, nécessite des conversions

→ approche le calcul du quotient avec une représentation classique

✗ coût proche de celui de l'algorithme de Montgomery

Protocoles imposent certaines propriétés sur le modulo

➤ utiliser des algorithmes fonctionnant pour tout les moduli

Algorithmes généralistes

Réduction intégrée à la multiplication

- Taylor — 1981
- Blakley — 1983
- Takagi — 1986

➥ adapté aux petits moduli

Réduction indépendante de la multiplication

- Montgomery — 1986
  - → réduction réduite à une division par une puissance de la base : décalage de bits
  - ✗ représentation particulière des nombres, nécessite des conversions
- Barrett — 1987
  - → approche le calcul du quotient avec une représentation classique
  - ✗ coût proche de celui de l'algorithme de Montgomery
- PMNS — 2004
  - → algorithmes plus efficaces que ceux de Montgomery et Barrett
  - ✗ pour un modulo donné, peu de systèmes construits en pratique

➥ + efficace sur de grands moduli

Approche

- Proposer une arithmétique modulaire efficace pour le plus grande nombre de moduli premiers possible
- La prémunir contre certains types d'attaques commes les attaques par canaux cachés et de Goubin

Trois objectifs majeurs

- Fournir de nouvelles bases de systèmes de représentation modulaires, en garantissant une arithmétique engendrée efficace

- Exploiter la redondance intrinsèque aux systèmes pour effectuer des changements de représentation des données au cours du calcul

- En parallèle de cette recherche en cryptographie, établir de nouvelles méthodes pour optimiser la résolution réelle des sytèmes flous

Plan

- Construction de systèmes PMNS pour un premier donné
- Randomisation de l'arithmétique sur le PMNS
- Présentation d'un sytème de représentation hybride
- Résolution réelle des systèmes polynomiaux flous
- Conclusion

Plan

- Construction de systèmes PMNS pour un premier donné
- Randomisation de l'arithmétique sur le PMNS
- Présentation d'un sytème de représentation hybride
- Résolution réelle des systèmes polynomiaux flous
- Conclusion

Bornes et existence des PMNS sur $\mathbb{Z}/p\mathbb{Z}$

Opérations modulaires présentes dans les algorithmes de cryptographie à clé publique actuels : RSA, l'échange de clés Diffie-Hellman et ECC.

Système de représentation adapté polynomial (PMNS) introduit en 2004

- une implémentation d'une arithmétique modulaire efficace impliquant uniquement des additions et des multiplications
- une arithmétique polynomiale rapide et parallélisation facile pour un modulo arbitraire
- des algorithmes plus efficaces que les méthodes sans division connues telles que Montgomery et Barrett

$\mathfrak{B} = (p, n, \gamma, \rho)_{E(X)}$

$E(X) \in \mathbb{Z}[X]$, *polynôme de réduction* ; unitaire de degré n

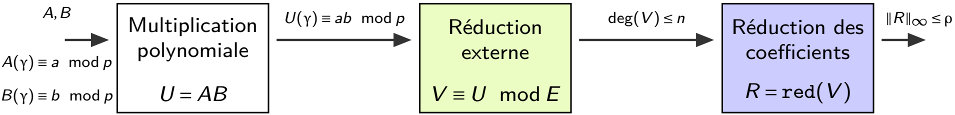
et $E(\gamma) \equiv 0 \pmod{p}$

$a \mod p \xrightarrow{\text{dans } \mathfrak{B}} A = (a_{n-1}, \ldots, a_0)$ avec

$$a \mod p \equiv \sum_{i=0}^{n-1} a_i \gamma^i \mod p,$$

où $a_i \in \mathbb{N}$, $|a_i| < \rho$, $1 < \gamma < p$.

Opérations sur les représentations d'un PMNS



Existence d'un PMNS

Restrictions du théorème d'existence

- Théorème existant prouve l'existence d'au moins un PMNS pour un entier p, pour $E(X) = X^n + aX + b$ satisfaisant certaines hypothèses.
- Construire de tels systèmes à partir d'un p n'est pas trivial :
  - chercher un $E(X)$ satisfaisant les conditions du théorème
  - et l'une de ses racines dans $\mathbb{Z}/p\mathbb{Z}$

Approche

Fournir autant de bases PMNS que possible pour un modulo fixé, avec une arithmétique efficace, pour

- obtenir des bases PMNS avec leurs propres propriétés de calcul
- Utiliser les différentes représentations pour masquer les calculs

Lien entre l'existence d'un PMNS et son réseau associé

Nous considérons le réseau associé au PMNS :

$$\mathfrak{L} = \{P(X) \in \mathbb{Z}[X], \text{ tel que : } \deg(P(X)) < n \text{ et } P(\gamma) \equiv 0 \mod p\}.$$

Theorem

Soit p premier, $n > 1$, $E(X) \in \mathbb{Z}[X]$ de degré n et γ une de ses racines dans $\mathbb{Z}/p\mathbb{Z}$. Soit r le rayon de recouvrement de $\mathfrak{L}$, si $\rho > r$, alors $\mathfrak{B} = (p, n, \gamma, \rho)_{E(X)}$ est un PMNS.

Introduction d'une borne sur ρ à partir d'une base de $\mathfrak{L}$

Nous considérons $\mathrm{B} = \{B_0, \ldots, B_{n-1}\}$ une base de $\mathfrak{L}$ et **B** sa matrice associée.

Démonstration.

Idée : borner la distance entre un point de $\mathbb{R}^n$ et son point le plus proche dans $\mathfrak{L}$ en utilisant une approche « round-off » de Babaï. □

Theorem

Si $\rho \geq \frac{1}{2} \|\mathbf{B}\|_1$, alors $\mathfrak{B} = (p, n, \gamma, \rho)_E$ est un PMNS.

Méthodes pour construire une base de $\mathfrak{L}$

- Stratégie 1 : calculer une base réduite de $\mathfrak{L}$ via LLL, BKZ ou HKZ

Les stratégie suivantes lorsque $E(X)$ est irréductible :

Soit $\mathbf{C}$ la matrice compagnon de $E(X)$
$V.\mathbf{C}^i$ correspond à $X^i.V(X) \mod E(X)$, pour $0 \leq i < n$
$\mathbf{A}$ une base de $\mathfrak{L}$

- Stratégie 2 :
  - choisir un vecteur court non nul $V \in \mathfrak{L}$
  - construire $\mathbf{B}$ la matrice $n \times n$ dont la i-ième ligne est le vecteur $V.\mathbf{C}^i$

  → $\mathbf{B}$ base d'un sous-réseau $\mathfrak{L}' \subseteq \mathfrak{L}$ de rang n, $V \in \mathfrak{L}'$ (Corollaire I.2.1)

- Stratégie 3 :
  - choisir un vecteur court non nul $(V_0|V_1|\cdots|V_{n-1})$ de $\mathfrak{L}_D$, le réseau de rang n dans $\mathbb{Z}^{n^2}$ défini par $\mathbf{D} = (\mathbf{A}|\mathbf{A}.\mathbf{C}^1|\cdots|\mathbf{A}.\mathbf{C}^{n-1})$
  - à construire la famille $B = (V_0, V_1, ..., V_{n-1})$ de $\mathfrak{L}$

  → B est une base de $\mathfrak{L}' \subseteq \mathfrak{L}$, $V_0 \in \mathfrak{L}$. (Corollaire I.2.2)

Exemples : Calcul de $\|\mathbf{B}\|_1$ pour chaque approche de base réduite

$p = 112848483075082590657416923680536930196574208889254960005437791530871071177777$
$n = 8,\ E(X) = X^8 + X^2 + X + 1,$
$\gamma = 1491636446523688584141872655968711774145114474053838625484298666226554558877 4$

LLL :	$\|\mathbf{B}\|_1 = 16940155314$	BKZ :	$\|\mathbf{B}\|_1 = 15289909984$
HKZ :	$\|\mathbf{B}\|_1 = 15289909984$		
Cor. I.2.1 :	$\|\mathbf{B}\|_1 = 13881325101$	Cor. I.2.2 :	$\|\mathbf{B}\|_1 = 12883199915$

$p = 967773291385464184116060378506706919162789802490357968454873914621632628778 31$
$n = 8,\ E(X) = X^8 - X^4 - 1,$
$\gamma = 663781196091410433177282902170533852564491454075567270041323732701464555754 61$

LLL :	$\|\mathbf{B}\|_1 = 17955608045$	BKZ :	$\|\mathbf{B}\|_1 = 17955608045$
HKZ :	$\|\mathbf{B}\|_1 = 17955608045$		
Cor. I.2.1 :	$\|\mathbf{B}\|_1 = 11628752571$	Cor. I.2.2 :	$\|\mathbf{B}\|_1 = 10489321362$

➥ les deux dernières approches offrent les meilleurs résultats pour $E(X)$ avec de petits coefficients.

Proposer des polynômes de réduction adaptés au PMNS

Definition

Un polynôme $E(X)$ est *un polynôme de réduction adapté au PMNS*, si :

- $E(X)$ est irréductible dans $\mathbb{Z}[X]$,
  - ➤ rendre applicables l'ensemble des stratégies précédentes
- $E(X) = X^n + a_k X^k + \cdots + a_1 X + a_0 \in \mathbb{Z}[X]$, avec $n \geq 2$ et $k \leq \frac{n}{2}$,
  - ➤ réduction polynomiale de $T(X)$ modulo $E(X)$ réalisée en deux étapes quand $\deg(T(X)) \leq 2n$
- la plupart des coefficients sont nuls, et les autres sont très petits (si possible égaux à ± 1) par rapport à $p^{1/n}$.
  - ➤ donner une borne sur les coefficients de $T(X) \bmod E(X)$

ClassCyclo(n)

Proposition

ClassCyclo(n) $\neq \emptyset$ si et seulement si $n = 2^i 3^j$ avec $i \geq 0,\ j \geq 0$.

Plus précisement, ClassCyclo(n) se compose des cyclotomiques

- $\Phi_{2^i}(X) = X^{2^{i-1}} + 1$, quand $n = 2^{i-1}$ avec $i \in \mathbb{N}^*$,
- $\Phi_{3^j}(X) = X^{2.3^{j-1}} + X^{3^{j-1}} + 1$, quand $n = 2.3^{j-1}$ avec $j \in \mathbb{N}^*$,
- $\Phi_{2^i.3^j}(X) = X^{2^i.3^{j-1}} - X^{2^{i-1}.3^{j-1}} + 1$, quand $n = 2^i.3^{j-1}$ pour $i,j \in \mathbb{N}^*$.

➥ Preuve : propriétés relatives aux cyclotomiques, connus comme étant irréductibles.

ClassBinomial(n,c) : binôme de réduction adapté $\{X^n + c\}$

TrinomialClass(n) : trinômes de réduction adaptés de degré n

QuadrinomialClass(n) : quadrinômes de réduction adaptés de degré n

ClassPrimeCst(n,μ) : coefficients dans {-1,0,1} + un coefficient constant μ premier

ClassPerron(n,a_1) : coefficients dans {-1,0,1} + un coefficient a_1 en X entier

Compter le nombre de PMNS constructibles pour un modulo premier

Proposition (Cas $E(X)$ est cyclotomique)

Soit $p > 2$ un premier, et un entier $m \geq 3$ tel que $m \mid (p-1)$ alors $\Phi_m(X)$ a $\varphi(m)$ racines dans $\mathbb{Z}/p\mathbb{Z}$ (φ l'indicatrice d'Euler).

Bornes sur les chiffres des PMNS obtenus à partir de `ClassCyclo(4)` pour $p = 22777$

$E(X)$	γ	ρ_{min}	$\frac{1}{2}\Vert\mathbf{B}\Vert_1$	$\frac{1}{2}\Vert\mathbf{D}\Vert_1$	$\frac{1}{2}\Vert\mathrm{LLL}(\mathbf{A})\Vert_1$
X^4+1	19189	9	11.5	11.5	17.5
	11890	9	11.5	11.5	17.5
	10887	9	11.5	11.5	17.5
	3588	9	11.5	11.5	17.5
X^4-X^2+1	22191	9	13.5	13.5	16.5
	17452	9	13.5	13.5	16.5
	5325	9	13.5	13.5	16.5
	586	9	13.5	13.5	16.5

Proposition (Cas $E(X)$ est un binôme)

Soit $E(X) = X^n + c$ un élément de `ClassBinomial(n,c)`*. Soit g un générateur de $(\mathbb{Z}/p\mathbb{Z})^\times$ et y tel que $g^y \equiv -c \mod p$.*
Si $\mathrm{pgcd}(n, p-1)$ divise y, alors, $E(X) = X^n + c$ a $\mathrm{pgcd}(n, p-1)$ racines distinctes dans $\mathbb{Z}/p\mathbb{Z}$.

Proposition (Cas général $E(X)$ irréductible)

Soit p premier, $n > 2$, $E(X)$ de degré n et irréductible dans $\mathbb{Z}[X]$, et $D(X) = \mathrm{pgcd}(X^p - X, E(X)) \mod p$.
Il existe $\deg(D(X))$ systèmes PMNS $(p, n, \gamma_i, \rho)_{E(X)}$.

Exemple : calcul du nombre de PMNS constructible à partir d'un premier

On choisit

- un premier p sur 256 bits,
  $p = 57896044618658097711785492504343953926634992332820282019728792003956566811073$
- un nombre de chiffres $n = 8$, i.e. $\rho \sim 2^{32}$,
- Les classes de polynômes adaptées au PMNS, telles que $\|E\|_\infty \leq 7$.

`ClassCyclo(n)` : 8 systems	`TrinomialClass(n)` : 24 systems
`QuadrinomialClass(n)` : no system	`ClassBinomials(n,3)` : no system
`ClassBinomials(n,4)` : no system	`ClassBinomials(n,5)` : no system
`ClassBinomials(n,6)` : 16 systems	`ClassBinomials(n,7)` : no system
`ClassPrimeCst(n,3)` : 6 systems	`ClassPrimeCst(n,5)` : 158 systems
`ClassPrimeCst(n,7)` : 190 systems	`ClassPerron(n,3)` : 8 systems
`ClassPerron(n,4)` : 38 systems	`ClassPerron(n,5)` : 78 systems
`ClassPerron(n,6)` : 104 systems	`ClassPerron(n,7)` : 112 systems

➤ Pour ce premier p et $n = 8$, on trouve 742 systèmes.

Synthèse

- Formalisation du lien entre l'existence d'un PMNS et son réseau euclidien
  - ✓ nouvelle borne sur la taille des chiffres à partir de la norme 1 du réseau
- Méthodes pour construire des bases d'un sous-réseau
- Introduction de classes de polynômes de réduction spécifiques
  - ✓ Réductions efficaces au sein du système
  - ✓ Les racines fournissent les bases de ces systèmes
- Méthodes pour compter le nombre de systèmes obtenus en fonction du polynôme de réduction
  - ✓ Offrir pour un modulo premier donné une grande variété de PMNS

Perspectives

- Recherche de nouvelles classes de polynômes de réduction
- Étude des PMNS définis par un polynôme réductible

Plan

- Construction de systèmes PMNS pour un premier donné
- Randomisation de l'arithmétique sur le PMNS
- Présentation d'un sytème de représentation hybride
- Résolution réelle des systèmes polynomiaux flous
- Conclusion

Attaques en ECC

Résistance aux attaques SCA

- Attaques par canaux cachés (SCA) profitent d'une fuite d'information durant l'exécution d'un protocole pour récupérer totalement ou partiellement le secret.
- SCA se sont avérés efficaces en ECC : contre-mesures doivent être inclus dans la mise en œuvre de la multiplication scalaire en ECC
- La méthode classique double et add n'est pas résistante au SCA. L'échelle Montgomery et sa variante sont plus résistantes mais peuvent être attaquées.

Idée

- Introduire la randomisation au niveau arithmétique

Utiliser une représentation aléatoire du point P chaque fois que ce point est utilisé pendant l'algorithme de multiplication scalaire

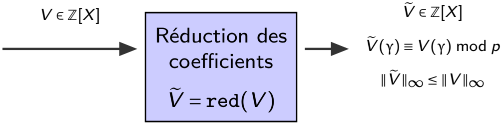
- assurer la résistance aux SCA et à des attaques de points spécifiques

- Les coordonnées de P sont représentées en PMNS
  - ✓ Nous utilisons la redondance du PMNS pour randomiser les coordonnées

Randomisation des entrées et de la multiplication

Approche

- Randomiser la multiplication scalaire de deux façons :
  - Randomisation de chaque coordonnée initiale de P en utilisant une procédure de conversion appropriée, assurant la résistance aux SCA et aux attaques de points spécifiques
  - Randomisation de chaque multiplication entre deux éléments dans la représentation PMNS pour être plus résistant aux SCA



- Réduction utilisée dans les primitives PMNS : multiplication et conversion.
- Deux méthodes pour effectuer cette procédure.

Réduction des coefficients de type Montgomery

Require: $V \in \mathbb{Z}[X]$ tel que $\deg(V) < n$;
$\mathscr{B} = (p, n, \gamma, \rho)_E$;
un entier r non nul
$M \in \mathscr{B}$, tel que $M(\gamma) \equiv 0 \bmod p$ et
$M' = -M^{-1} \bmod (E, r)$.

Ensure: $R(\gamma) \equiv V(\gamma) r^{-1} \bmod p$

1: $Q \leftarrow V \times M' \bmod (E, r)$

2: $R' \leftarrow V + (Q \times M) \bmod E$

3: $R \leftarrow R'/r$

4: return R

.

- r est souvent une puissance de 2 pour garantir une division exacte par r rapide
- M' existe si et seulement si $\mathrm{pgcd}(\mathrm{resultant}(E, M), r) = 1$

➤ 2 : $R'(\gamma) \equiv V(\gamma) \bmod p$,
coefficients de R' divisibles par r

➤ 4 : $R(\gamma) \equiv V(\gamma) r^{-1} \bmod p$.

Introduction d'un aléa dans la primitive de réduction des coefficients du PMNS

$z \in \mathbb{N}$ $\longrightarrow$ randomPoly $\longrightarrow$ $Z \in \mathbb{Z}[X]$

choisi lors du processus de génération de PMNS

$\|Z\|_\infty \leq z$

$\deg Z < n$

- randomPoly est considérée comme sûre.
- les coefficients de Z sont générés en utilisant la distribution uniforme.

➤ exactement $(2z+1)^n$ sorties possibles Z

Randomisation de la conversion via Montgomery

Trois conditions pour la randomisation

- $M(\gamma) \equiv 0 \bmod p$
  - ➤ $A(\gamma) \bmod p$ n'est pas modifié
- Calculs modulo E et r permettent de donner borne sur ρ .
  - ➤ Résultat dans $\mathscr{B}$
- $\mathrm{pgcd}(E, M) = 1$ dans $\mathbb{Q}[X]$
  - ➤ Si $Z \neq Z'$ alors les sorties sont distinctes

Algorithme 1 Conversion randomisée vers le PMNS via Montgomery

Require: $a \in \mathbb{Z}/p\mathbb{Z}$ et $\mathscr{B} = (p, n, \gamma, \rho)_E$
Ensure: $A(\gamma) \equiv a\,r \bmod p$
Data : $P_i \equiv (\rho^i)_{\mathscr{B}}$, pour $i = 0, \ldots, n-1$, $z \in \mathbb{N}$, $r = 2^j$, $j \geq 1$ et $M \in \mathscr{B}$ avec $M(\gamma) \equiv 0 \bmod p$ et $\mathrm{pgcd}(r, \mathrm{resultant}(E, M)) = 1$.

1: $Z \leftarrow$ **randPoly**(z)
2: $a' \leftarrow a\,r^2 \bmod p$
3: $b \leftarrow (a'_{n-1}, \ldots, a'_0)_\rho$
4: $U \leftarrow \sum_{i=0}^{n-1} a'_i P_i$
5: $V \leftarrow U + ((r+1) Z \times M) \bmod E$
6: $A \leftarrow$ **RedCoeff**(V)
7: **return** A

Theorem

Nous considérons :

- $\mathscr{B} = (p, n, \gamma, \rho, E)$ *un PMNS*
- $a \in \mathbb{Z}/p\mathbb{Z}$
- $r = 2^j$, $j \geq 1$
- $M \in \mathscr{B}$ *tel que* $M(\gamma) \equiv 0 \bmod p$ *et* $\mathrm{pgcd}(r, \mathrm{resultant}(E, M)) = 1$, $m = \|M\|_\infty$
- z *l'entrée de la procédure* `randPoly`

comme entrées et données de l'Algorithme. Si ρ *et* r *satisfont*

$$\rho \geq 2nsm\left(1 + z + \frac{z}{r}\right) \quad \text{et} \quad r \geq 2ns\rho,$$

alors l'Algorithme 1 peut générer $(2z+1)^n$ *sorties distinctes uniformément distribuées, toutes représentant* a *dans* $\mathscr{B}$.

RedCoeff - Babaï like

$$\mathbf{A} = \begin{pmatrix} p & 0 & \dots & \dots & 0 & 0 \\ -\gamma & 1 & \dots & \dots & 0 & 0 \\ \vdots & \ddots & \ddots & & & \vdots \\ 0 & \dots & -\gamma & 1 & \dots & 0 \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & 0 & \dots & \dots & -\gamma & 1 \end{pmatrix}$$

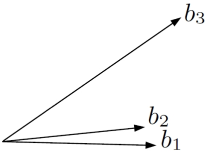


FIGURE – Réduction via Babaï en dimension 3. Les hyperplans sélectionnés sont verts.

Data : $B = \{b_i, 1 \leq i \leq n\}$ la base LLL réduite de $\mathfrak{L}_{\mathscr{B}}$

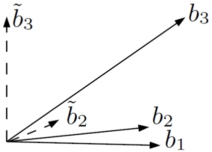


FIGURE – Réduction via Babaï en dimension 3. Les hyperplans sélectionnés sont verts.

Data : $B = \{b_i, 1 \le i \le n\}$ la base LLL réduite de $\mathfrak{L}_{\mathscr{B}}$
$\widetilde{B} = \{\widetilde{b_i}, 1 \le i \le n\}$ la base Gram-Schmidt obtenue à partir de B.

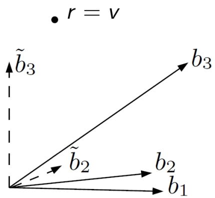


FIGURE – Réduction via Babaï en dimension 3. Les hyperplans sélectionnés sont verts.

Require: $V \in \mathbb{Z}[X]$ with $\deg(V) < n$, $\mathscr{B} = (p, n, \gamma, \rho, E)$

Data : $B = \{b_i, 1 \le i \le n\}$ la base LLL réduite de $\mathfrak{L}_{\mathscr{B}}$

$\widetilde{B} = \{\widetilde{b_i}, 1 \le i \le n\}$ la base Gram-Schmidt obtenue à partir de B.

1: $r \leftarrow v$

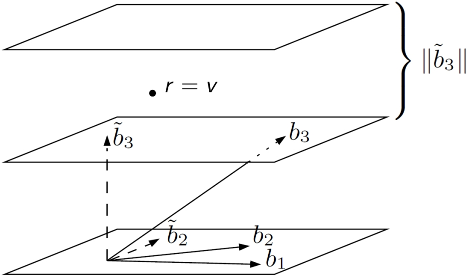


FIGURE – Réduction via Babaï en dimension 3. Les hyperplans sélectionnés sont verts.

Require: $V \in \mathbb{Z}[X]$ with $\deg(V) < n$, $\mathscr{B} = (p, n, \gamma, \rho, E)$

Data : $B = \{b_i, 1 \leq i \leq n\}$ la base LLL réduite de $\mathfrak{L}_{\mathscr{B}}$

$\widetilde{B} = \{\widetilde{b_i}, 1 \leq i \leq n\}$ la base Gram-Schmidt obtenue à partir de B.

1: $r \leftarrow v$

2: **for** $i = 1$ **to** n **do**

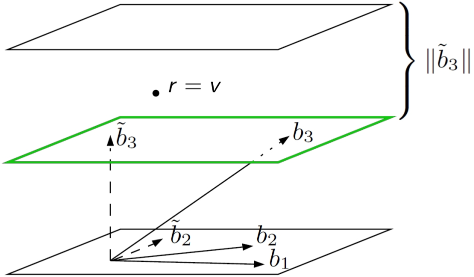


FIGURE – Réduction via Babaï en dimension 3. Les hyperplans sélectionnés sont verts.

Require: $V \in \mathbb{Z}[X]$ with $\deg(V) < n$, $\mathscr{B} = (p, n, \gamma, \rho, E)$

Data : $B = \{b_i, 1 \le i \le n\}$ la base LLL réduite de $\mathfrak{L}_{\mathscr{B}}$

$\widetilde{B} = \{\widetilde{b}_i, 1 \le i \le n\}$ la base Gram-Schmidt obtenue à partir de B.

1: $r \leftarrow v$

2: **for** $i = 1$ **to** n **do**

3: $\quad c \leftarrow \lfloor < r, \widetilde{b}_{n-i+1} > / \|\widetilde{b}_{n-i+1}\|^2 \rceil$

RedCoeff - Babaï like

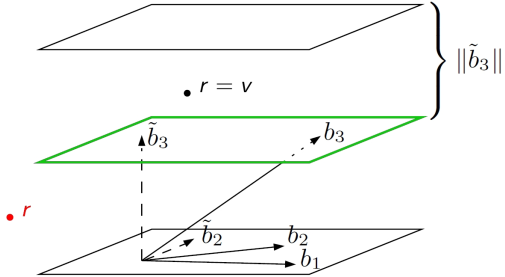


FIGURE – Réduction via Babaï en dimension 3. Les hyperplans sélectionnés sont verts.

Require: $V \in \mathbb{Z}[X]$ with $\deg(V) < n$, $\mathscr{B} = (p, n, \gamma, \rho, E)$

Data : $B = \{b_i, 1 \leq i \leq n\}$ la base LLL réduite de $\mathfrak{L}_{\mathscr{B}}$

$\widetilde{B} = \{\widetilde{b}_i, 1 \leq i \leq n\}$ la base Gram-Schmidt obtenue à partir de B.

$r \leftarrow v$

for $i = 1$ **to** n **do**

  $c \leftarrow \lfloor < r, \widetilde{b}_{n-i+1} > / \|\widetilde{b}_{n-i+1}\|^2 \rceil$

  $r \leftarrow r - c \times b_{n-i+1}$

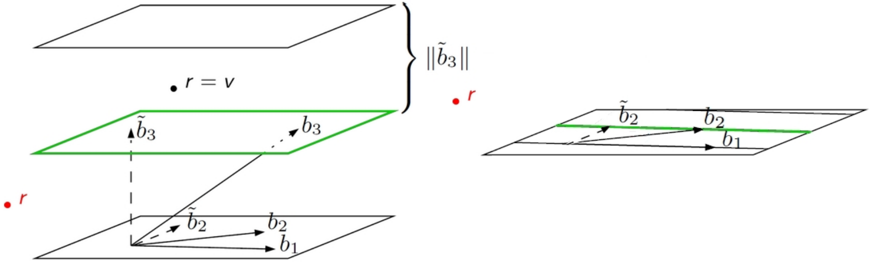


FIGURE – Réduction via Babaï en dimension 3. Les hyperplans sélectionnés sont verts.

Require: $V \in \mathbb{Z}[X]$ with $\deg(V) < n$, $\mathscr{B} = (p, n, \gamma, \rho, E)$

Data : $B = \{b_i, 1 \leq i \leq n\}$ la base LLL réduite de $\mathfrak{L}_{\mathscr{B}}$

$\widetilde{B} = \{\widetilde{b}_i, 1 \leq i \leq n\}$ la base Gram-Schmidt obtenue à partir de B.

Ensure: $r \in \mathscr{B}$ **such that** $\|R\|_\infty \leq \|V\|_\infty$ **and** $r(\gamma) = V(\gamma) \mod p$

$r \leftarrow v$

for $i = 1$ **to** n **do**

$\quad c \leftarrow \lfloor < r, \widetilde{b}_{n-i+1} > / \|\widetilde{b}_{n-i+1}\|^2 \rceil$

$\quad r \leftarrow r - c \times b_{n-i+1}$

end for

return r

Trois conditions pour la randomisation

- V vecteur ***masque*** dans $\mathfrak{L}_{\mathscr{B}}$, Z vecteur ***translation*** dont les coeffs sont multipliés par des éléments de la base
  - ➤ $A(\gamma) \bmod p$ n'est pas modifié
- La sortie se trouve se trouve dans un rectangle connu
  - ➤ Résultat dans $\mathscr{B}$
- $\mathrm{pgcd}(E, H) = 1$ dans $\mathbb{Q}[X]$ et chaque Z définit une suite distincte d'hyperplans pour une base donnée
  - ➤ Si $Z \neq Z'$ alors les sorties sont distinctes

Algorithme 2 Conversion randomisée vers le PMNS via Babaï

Require: $a \in \mathbb{Z}/p\mathbb{Z}$ et $\mathscr{B} = (p, n, \gamma, \rho)_E$
Ensure: $A(\gamma) \equiv a \bmod p$
Data : $P_i \equiv (\rho^i)_{\mathscr{B}}$, **pour** $i = 0, \dots, n-1$, $D, \widetilde{D}$, $v, z \in \mathbb{N}$, $M \in \mathscr{B}$ **et** $H \in \mathbb{Z}[X]$.

1: $V_0 \leftarrow$ **randPoly**(v), $Z_0 \leftarrow$ **randPoly**(z)
2: $V \leftarrow V_0 \times M \bmod E$, $Z \leftarrow Z_0 \times H \bmod E$
3: $b \leftarrow (a_{n-1}, \dots, a_0)_\rho$
4: $T \leftarrow \sum_{i=0}^{n-1} a_i P_i$, $A \leftarrow T + V$
5: **for** $i = 1$ **to** n **do**
6: $\quad c \leftarrow \lfloor < A, \widetilde{D}_{n-i+1} > / \|\widetilde{D}_{n-i+1}\|^2 \rceil + z_{n-i}$
7: $\quad A \leftarrow A - c \times D_{n-i+1}$
8: **end for**
9: **return** A

Theorem

Nous considérons :

- $\mathscr{B} = (p, n, \gamma, \rho, E)$ *un PMNS.*
- $a \in \mathbb{Z}/p\mathbb{Z}$
- v *et* z *les entrées de la procédure* `randPoly`
- $M \in \mathscr{B}$ *tel que* $M(\gamma) \equiv 0 \bmod p$ *et* $\mathrm{pgcd}(E, M) = 1$ *dans* $\mathbb{Q}[X]$
- $H \in \mathbb{Z}[X]$ *tel que* $\mathrm{pgcd}(E, H) = 1$ *dans* $\mathbb{Q}[X]$, $h = \|H\|_\infty$.

comme entrées et données de l'Algorithme. Si ρ *satisfait*

$$\rho > \left(\frac{1}{2} + nszh\right)\left(2^{\frac{3n-1}{2}} p^{1/n}\right),$$

alors l'Algorithme 3 peut générer $(2z+1)^n$ *sorties distinctes uniformément distribuées, toutes représentant* a *dans* $\mathscr{B}$.

Deux exécutions de l'algorithme de type Babai

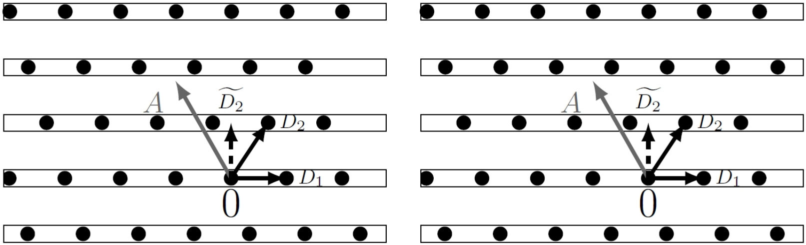


FIGURE – Réduction via Babaï en dimension 2. Les hyperplans sélectionnés sont grisés ou entourés.

Non randomisé

Randomisé

$z = 1 \rightarrow$ translation $Z = (-1, 1)$

Deux exécutions de l'algorithme de type Babai

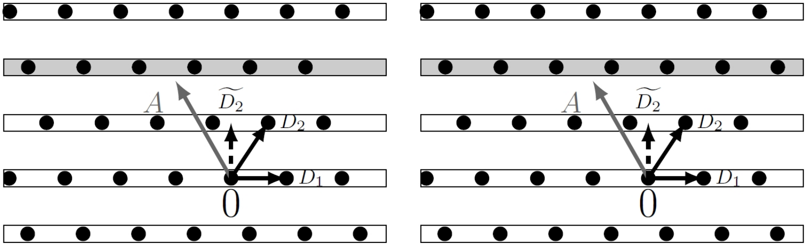


FIGURE – Réduction via Babaï en dimension 2. Les hyperplans sélectionnés sont grisés ou entourés.

Non randomisé

Randomisé

$z = 1 \rightarrow$ translation $Z = (-1, 1)$

$$c \leftarrow \lfloor < A, \widetilde{D}_2 > / \|\widetilde{D}_2\|^2 \rceil$$

Deux exécutions de l'algorithme de type Babai

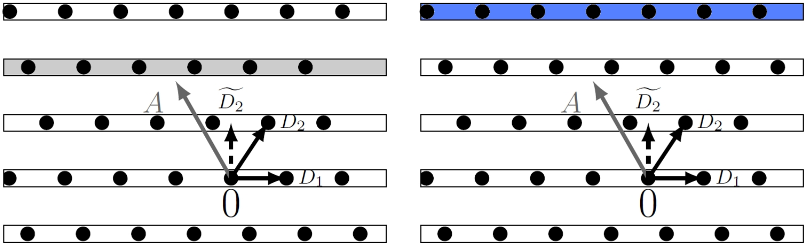


FIGURE – Réduction via Babaï en dimension 2. Les hyperplans sélectionnés sont grisés ou entourés.

Non randomisé

Randomisé

$z = 1 \rightarrow$ translation $Z = (-1, \underline{1})$

$$c \leftarrow \lfloor < A, \widetilde{D}_2 > / \| \widetilde{D}_2 \|^2 \rceil + Z_2$$

Deux exécutions de l'algorithme de type Babai

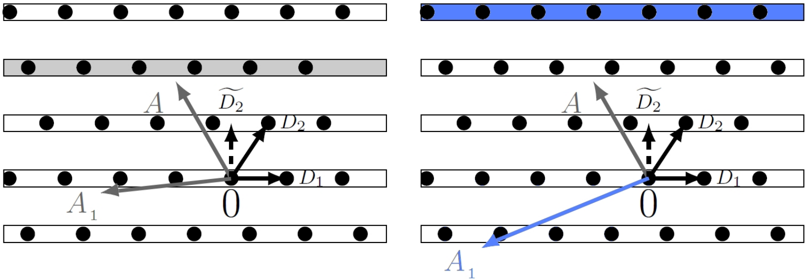


FIGURE – Réduction via Babaï en dimension 2. Les hyperplans sélectionnés sont grisés ou entourés.

Non randomisé

Randomisé

$z = 1 \rightarrow$ translation $Z = (-1, \underline{1})$

$$A_1 \leftarrow A - c \times D_2$$

Deux exécutions de l'algorithme de type Babai

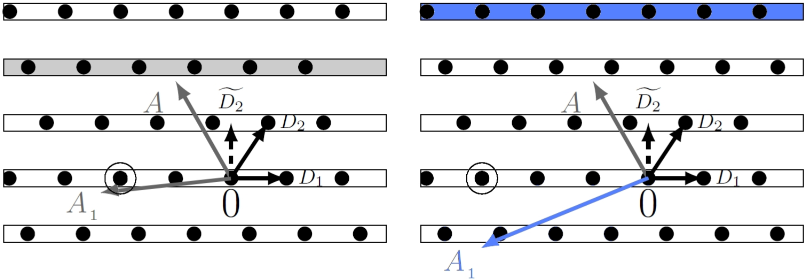


FIGURE – Réduction via Babaï en dimension 2. Les hyperplans sélectionnés sont grisés ou entourés.

Non randomisé

Randomisé

$z = 1 \rightarrow$ translation $Z = (-1, \underline{1})$

$$c \leftarrow \lfloor < A_1, \widetilde{D}_1 > / \| \widetilde{D}_1 \|^2 \rceil$$

Deux exécutions de l'algorithme de type Babai

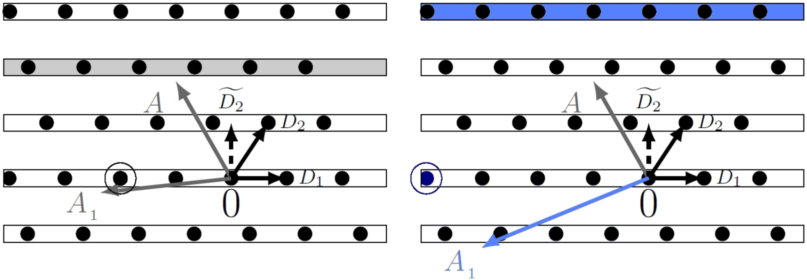


FIGURE – Réduction via Babaï en dimension 2. Les hyperplans sélectionnés sont grisés ou entourés.

Non randomisé

Randomisé

$z = 1 \rightarrow$ translation $Z = (\underline{-1}, 1)$

$$c \leftarrow \lfloor < A_1, \widetilde{D}_1 > / \| \widetilde{D}_1 \|^2 \rceil + Z_1$$

Deux exécutions de l'algorithme de type Babai

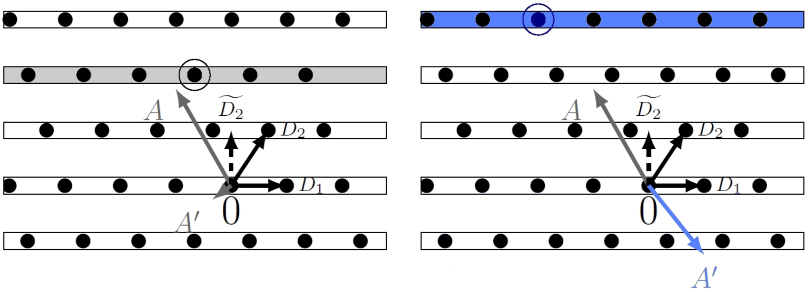


FIGURE – Réduction via Babaï en dimension 2. Les hyperplans sélectionnés sont grisés ou entourés.

Non randomisé

Randomisé

$z = 1 \rightarrow$ translation $Z = (\underline{-1}, 1)$

$$A' \leftarrow A_1 - c \times D_1$$

Nous présentons deux algorithmes pour randomiser la multiplication

- une entrée est randomisée afin de randomiser tous les résultats intermédiaires

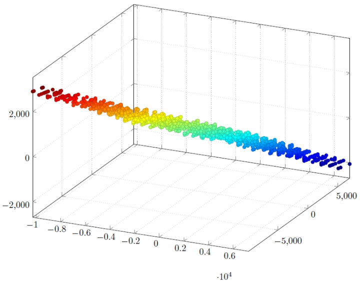


Fig. II.2 *Vecteurs retournés pour 1000 exécutions de la multiplication randomisée via Babaï pour les mêmes entrées* $\mathcal{B} = (p, n, \gamma, \rho)_E$ *et* $A, B \in \mathcal{B}$, *avec* $p = 45667$, $\gamma = 15943$, $E(X) = X^3 - X + 1$, $R(\gamma) = 44981 \mod p$ *et* $z = 6$.

Avantages

Attaque de Goubin : sur des courbes qui ont au moins un point avec une coordonnée égale à zéro.

- quel que soit le type de courbe, cette attaque ne peut être effectuée
  - ✓ au moins $(2z+1)^n$ représentants distincts de $0 \in \mathbb{Z}/p\mathbb{Z}$ qui n'ont pas de formes particulières

  Pour z assez grand, pas d'information exploitable pour effectuer cette attaque.
  - ✓ Randomiser la conversion via Montgomery ou Babaï suffit à contrer l'attaque de Goubin
- elle opère au niveau arithmétique.
  - ✓ peut être combinée avec d'autres contre-mesures classiques (point blinding, scalar blinding)

$\mathcal{M}$ et $\mathcal{A}$: multiplication et addition de deux entiers de w-bits

$\mathcal{I}$: division par un entier de $2w + \lfloor \log_2(n) \rfloor$ bits

$\mathcal{R}$: coût d'un appel à la fonction `randPoly`

Nous notons respectivement $\mathcal{S}_l^i$ et $\mathcal{S}_r^i$ un décalage à gauche et un décalage à droite de i bits

Coût théoriques des opérations, quand $E(X) = X^n - \lambda$ avec $\lambda = \pm 2^u$, $r = 2^j$.

Méthode de mult.	type Montgomery	type Babaï
Mult. polynomiale	$n^2\mathscr{M} + (3n^2 - 6n + 3)\mathscr{A}$	$n^2\mathscr{M} + (3n^2 - 6n + 3)\mathscr{A}$
Réduction polynomiale	$3(n-1)\mathscr{A} + (n-1)\mathscr{S}_l^u$	$3(n-1)\mathscr{A} + (n-1)\mathscr{S}_l^u$
Réduction interne	$2n^2\mathscr{M} + (4n^2 - n)\mathscr{A} + n\mathscr{S}_r^j$	$2n^2\mathscr{M} + (6n^2 - 3n)\mathscr{A} + n\mathscr{I}$
Total	$3n^2\mathscr{M} + (7n^2 - 4n)\mathscr{A} + (n-1)\mathscr{S}_l^u + n\mathscr{S}_r^j$	$3n^2\mathscr{M} + (9n^2 - 6n)\mathscr{A} + n\mathscr{I} + (n-1)\mathscr{S}_l^u$
Méthode de mult.	Montgomery randomisé (Alg. 8)	Babaï randomisé (Alg. 9)
Mult. polynomiale	$2n^2\mathscr{M} + (4n^2 - 7n + 3)\mathscr{A} + \mathscr{R}$	$3n^2\mathscr{M} + (5n^2 - 8n + 3)\mathscr{A} + 2\mathscr{R}$
Réduction polynomiale	$3(n-1)\mathscr{A} + (n-1)\mathscr{S}_l^u$	$3(n-1)\mathscr{A} + (n-1)\mathscr{S}_l^u$
Réduction interne	$2n^2\mathscr{M} + (4n^2 + n)\mathscr{A} + n(\mathscr{S}_r^j + \mathscr{S}_l^1)$	$2n^2\mathscr{M} + (6n^2 - 2n)\mathscr{A} + n\mathscr{I}$
Total	$4n^2\mathscr{M} + (8n^2 - 3n)\mathscr{A} + (n-1)\mathscr{S}_l^u + n(\mathscr{S}_l^1 + \mathscr{S}_r^j) + \mathscr{R}$	$5n^2\mathscr{M} + (11n^2 - 7n)\mathscr{A} + n\mathscr{I} + (n-1)\mathscr{S}_l^u + 2\mathscr{R}$

Synthèse

- Exploitation de la redondance du PMNS pour définir des protections arithmétiques contre les attaques DPA
  - ✓ générer une représentation aléatoire suivant une loi uniforme
- Randomisation des entrées lors de la conversion vers le PMNS via deux méthodes
  - ✓ suffit pour se protéger contre l'attaque de Goubin
- Introduction de deux multiplications modulaires randomisées au sein du PMNS
  - ✓ ces méthodes peuvent être employées pour appliquer des contre-mesures classiques en ECC

Perspectives

- Étude plus approfondie sur l'efficacité pratique
- Comparaison exhaustive avec les contre-mesures existantes suivront bientôt afin d'établir quand ces méthodes sont les plus pertinentes

Plan

- Construction de systèmes PMNS pour un premier donné
- Randomisation de l'arithmétique sur le PMNS
- Présentation d'un sytème de représentation hybride
- Résolution réelle des systèmes polynomiaux flous
- Conclusion

Améliorer la réduction modulaire pour tous les premiers

Algorithmes existants

- Pour des premiers particuliers
  - Plantard (2004) avec PMNS : un produit se réécrit $D = D_L + D_H 2^k$ avec $\|D_L\| < 2^k$. puis D mis à jour avec $D_L + D_H \star M$ (temps quadratique)
  - Bigou et Tisserand (2016) avec HPR : système hybride positionnelle-RNS : limité à $p = B_1^n - \beta$. Réduction via une propagations de retenue (temps sous-quadratique)
- Pour tous les premiers
  - Plantard (2005) avec PMNS : un produit se réécrit $D = D_L + D_H 2^{k-1}$ avec $\|D_L\| < 2^{k-1}$, puis D mis à jour avec $D_L + D'_H$, D'_H dans un tableau pré-calculé (temps quadratique)
- le choix de p, standardisé pour la plupart des applications cryptographiques, n'est pas toujours possible.

Idée

- Proposer un nouveau système de représentation hybride polynomial, HyPoRes
- pour permettre des réductions efficaces pour tout premier

- Les éléments sont représentés dans un PMNS, avec les chiffre en RNS

Système HyPoRes

Paramètres

- Un sextuplet $\mathscr{H} = (p, n, \rho, \mathscr{B}_1, \mathscr{B}_2, b_{sk})$,
- β le plus petit entier qui n'est pas une puissance n-ième sur $\mathbb{Z}$,
- mais qui possède une racine n-ième γ modulo p.

$\mathscr{H} = (p, n, \rho, \mathscr{B}_1, \mathscr{B}_2, b_{sk})$

$E(X) = X^n - \beta \in \mathbb{Z}[X]$, *polynôme de réduction* ; unitaire de degré n

et $E(\gamma) \equiv 0 \pmod{p}$

$a \mod p \xrightarrow{\text{dans } \mathscr{H}} A = (a^{(0)}, \ldots, a^{(n-1)})$, avec

$a^{(i)}$

RNS $\mathscr{B}_1 = \{b_{1,0}, \ldots, b_{1,h_1-1}\}$	RNS $\mathscr{B}_2 = \{b_{2,0}, \ldots, b_{2,h_2-1}\}$	modulo b_{sk}
$A_1^{(i)}$	$A_2^{(i)}$	$a_{sk}^{(i)}$

$$a \equiv \sum_{i=0}^{n-1} \left[\sum_{j=0}^{h_1-1} \xi_{i,1,j} \frac{B_1}{b_{1,j}} \right]_{B_1} \gamma^i \mod p \quad \text{avec } \xi_{i,1,j} = \left[a_{i,1,j} \frac{b_{1,j}}{B_j} \right]_{b_{1,j}}, \quad |a^{(i)}| < \rho$$

Multiplication modulaire

Algorithme 3 Algorithme de multiplication modulaire

Require: $\|A\|_\infty, \|C\|_\infty < k\rho$
Require: $M' = -M^{-1} \bmod \mathscr{B}_1$
Ensure: $\|R\|_\infty < \rho$ avec $r = acB_1^{-1} \bmod p$

$D \leftarrow A \star C \bmod \mathscr{B}_1 \cup \mathscr{B}_2 \cup \{b_{sk}\}$

$A \star C = AC \bmod (X^n - \beta)$

$Q_1 \leftarrow D \star M' \bmod \mathscr{B}_1$

Calcul du quotient q dans $\mathscr{B}_1$

$Q_2 \leftarrow \mathrm{FastBConv}(q, \mathscr{B}_1) \bmod \mathscr{B}_2$

Extension de q dans la base $\mathscr{B}_2$

$q_{sk} \leftarrow \mathrm{FastBConv}(q, \mathscr{B}_1) \bmod b_{sk}$
$R_2 \leftarrow \dfrac{D + Q_2 \star M}{B_1} \bmod \mathscr{B}_2$

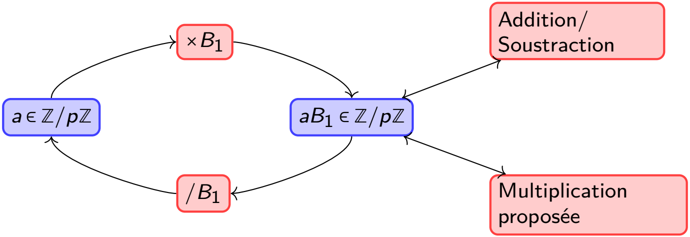
Calcul d'une rep. équivalente à $A \star C$, nulle modulo B_1, division par B_1 dans $\mathscr{B}_2$ pour obtenir R de norme inférieure

$r_{sk} \leftarrow \dfrac{d_{sk} + q_{sk} \star M}{B_1} \bmod b_{sk}$
$\alpha \leftarrow \left[(\mathrm{FastBConv}(r, \mathscr{B}_2) - r_{sk})B_2^{-1}\right]_{b_{sk}}$
$R_1 \leftarrow \mathrm{FastBConvSK}(r, \mathscr{B}_2, \alpha) \bmod \mathscr{B}_1$

Extension de r dans $\mathscr{B}_1$

$R \leftarrow (R_1, R_2)$
return R

- M de norme $\leq p^{1/n}$ existe dans $\mathscr{H}$ (Lemme I.2.1),
- Si $\mathscr{B}_1 = \{b_{1,0}, \ldots, b_{1,h_1-1}\}$ est tel que tous les $b_{1,i}$ sont des premiers ne divisant pas resultant(M, $X^n - \beta$) et $M \not\equiv 0 \bmod B_1$, alors M inversible dans $\mathbb{Z}_{B_1}[X]/(X^n - \beta)$ (Lemme 1.2.2),
- L'agorithme de multiplication est correct (Théorème I.2.3).



Autres opérations : addition, conversion utilisant la mult. et la base des $M \star X^i$, $0 \leq i \leq n$.

Complexité de la mult.

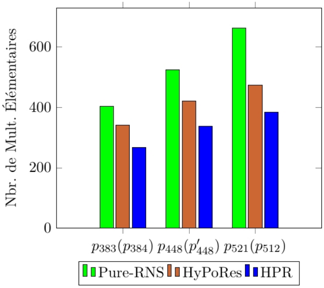
- multiplications par β : décalages et additions, ou au précalcul.
- opération $\star$: n^2 mult. pour chaque module.
- constantes dans le calcul des extensions intégrée aux précalculs.

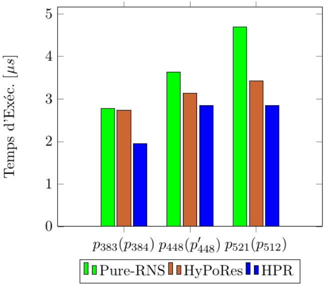
Pour $n \sim h_1 \sim h_2 \sim \log_2^{1/2} p$:

- HyPoRes : $\mathcal{O}(\log_2^{3/2} p)$ SMM.

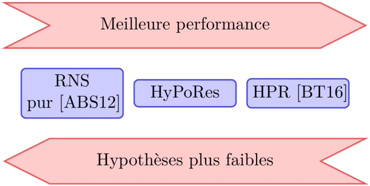
  En comparaison, avec des paramètres équivalents,

- HPR : $\mathcal{O}(\log_2^{3/2} p)$ SMM   RNS-pure : $\mathcal{O}(\log_2^2 p)$





- HyPoRes a une meilleure évolutivité que RNS lorsque la taille des premiers augmente
- accélération maximale $\simeq 1{,}4$ par rapport à RNS
- contrairerement à HPR, HyPoRes peut être utilisé chaque fois que la factorisation du module sous-jacent est connue



Empêcher toute prédiction sur la consommation d'énergie

Généralisation du polynôme de réduction

- γ racine de $E(X) = e^{(0)} + \ldots + e^{(n-1)} X^{n-1} + X^n$ modulo p,

  irréductible dans $\mathbb{Z}[X]$, avec de petits coefficients

- Existence d'une petite représentation non nulle de zéro M (Lemme I.2.1 indépendant de E)

- Mult. implémentée uniquement avec des décalages et des additions

  ➤ complexité de l'Algorithme de mult. est conservée

Deux types de conversion

Paramètres pré-calculés dans plusieurs HyPoRes avec des E différents.

- Résistance aux DPA : randomiserau début de la mult.

  ➤ Sélectionner un E aléatoire, une conversion du binaire vers HyPoRes est nécessaire.

- Résistance à des attaques de points : randomiser au cours de la mult.

  ➤ Conversions entre HyPoRes avec un E différent

$$A' = A[0] + \sum_{i=1}^{n-1} \text{HyPoRes-mul}_{E'}(A[i], T_{E \to E'}[i]),$$

$$\text{avec } T_{E \to E'}[i] = \left[\gamma^i B_1\right]_p \text{ dans } \mathcal{H}_{E'}$$

.

Synthèse

- Introduction d'un système hybride qui supporte n'importe quelle valeur première
  - ✓ compatible avec les courbes elliptiques standardisées
- Réduction de la complexité des extensions de base par rapport à ue approche RNS
  - ✓ accélération pouvant atteindre 1,4 lorsqu'il est comparé aux approches basées sur RNS
- Complexité en temps sous-quadratique similaire à celle d'HPR
- Généralisation du polynôme de réduction
  - ✓ introduction de représentations redondantes, procurant une protection contre les attaques SCA.

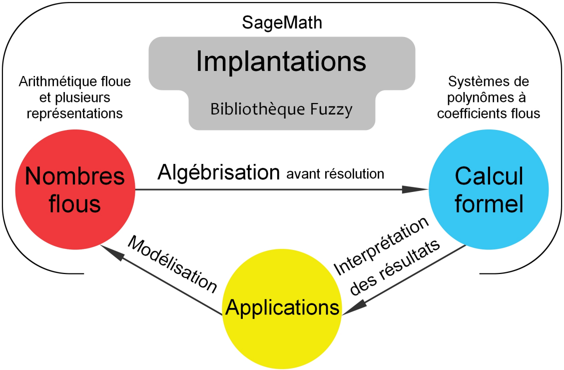
Perspectives

- Nouvelles optimisations : algorithme de Barrett pour la réduction au sein de HyPoRes
- Étude des bases RNS utilisées pour la randomisation

Plan

- Construction de systèmes PMNS pour un premier donné
- Randomisation de l'arithmétique sur le PMNS
- Présentation d'un sytème de représentation hybride
- Résolution réelle des systèmes polynomiaux flous
- Conclusion

Chercher les solutions réelles d'un système polynomial flou : crucial pour l'interpolation des fonctions floues



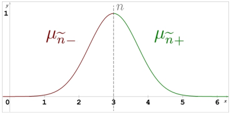
Méthodes connues avec des nombres flous spécifiques

- approches locales et globales se concentrent sur les nombres flous *triangulaires*

Notre approche

- pour tout nombre flou à support borné et de fonctions de dispersion bijectives, représentation tuple transformable en une autre représentation dite "paramétrique", où les coefficients ne sont plus flous mais réels.

Nombres flous : modéliser des problèmes aux données incertaines (Zadeh,1965)



Nombre flou $\tilde{n}$

- Fonction d'appartenance : $\mu_{\tilde{n}}(x) \in [0,1]$ représente le degré de validité de la proposition

  "x est la valeur de $\tilde{n}$"

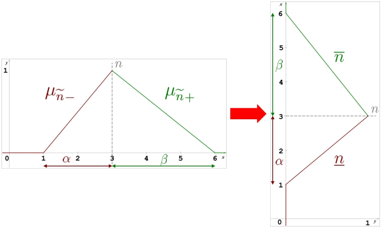
  ➥ n est le *mode*
- Avantages : capturer l'incertitude autour d'une valeur donnée

$$Supp(\tilde{n}) = Supp(\mu_{\tilde{n}}(x)) = \{a \in \mathbb{R} \mid \mu_{\tilde{n}}(a) \neq 0\}$$

Deux représentations

tuple

$(n,\alpha,\beta) \in \mathfrak{F}(L,R)$



paramétrique

paire ordonnée de fonctions $[\underline{n}, \overline{n}]$

Calculer les solutions positives de

$$(E) : \sum_{\boldsymbol{d} \in \mathrm{Expon}(E)} \widetilde{n_{\boldsymbol{d}}}\, \boldsymbol{x}^{\boldsymbol{d}} = \widetilde{m}$$

Approche connue : coefficients flous triangulaires

① Calculer $\mathscr{C}(E)$ la forme tranchée de (E) : passer les coefficients en paramétrique

② Calculer $\mathscr{C}\mathscr{C}(E)$ la forme tranchée collectée de (E) :

[BBRV2016] : Collecter les coefficients de $\mathscr{C}(E)$ en la variable r et constants

$$\mathscr{C}(E) : \begin{cases} \sum_{\boldsymbol{d}} \alpha_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}} & = \quad \alpha \\ \sum_{\boldsymbol{d}} (n_{\boldsymbol{d}} - \alpha_{\boldsymbol{d}})\, \boldsymbol{x}^{\boldsymbol{d}} & = \quad m - \alpha \\ \sum_{\boldsymbol{d}} \beta_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}} & = \quad \beta \\ \sum_{\boldsymbol{d}} (n_{\boldsymbol{d}} + \beta_{\boldsymbol{d}})\, \boldsymbol{x}^{\boldsymbol{d}} & = \quad m + \beta\ . \end{cases}$$

$\hookrightarrow$ $\mathscr{C}\mathscr{C}(E)$ a 4 équations à coefficients réels et k indéterminés.

$$\mathrm{Sol}^+(E) = \mathrm{Sol}^+(\mathscr{C}\mathscr{C}(E))$$

Peut être résolu avec des techniques algébriques

Proposer un nouveau système tranché général

Theorem (1)

Le système tranché général $\mathscr{C}(E)$ *de* (E) *est donné par :*

$$\mathscr{C}(E) : \begin{cases} \sum_{\boldsymbol{d}} n_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}} - m + (\alpha - \sum_{\boldsymbol{d}} \alpha_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}})\, u & = & 0 \\ \sum_{\boldsymbol{d}} n_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}} - m + (-\beta + \sum_{\boldsymbol{d}} \beta_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}})\, v & = & 0\ , \end{cases}$$

et $\mathrm{Sol}^+(E) = \mathrm{Sol}_k^+(\mathscr{C}(E))$

Fonctionne pour L et R bijectives, avec $u = L^{-1}(r)$ and $v = R^{-1}(r)$.

Déduire un nouveau système tranché collecté

Theorem (2)

La transformation réelle $\mathscr{T}(E)$ *de* (E) *est le système polynomial suivant sur* $\mathbb{R}$ *:*

$$\mathscr{T}(E) \begin{cases} \sum_{\boldsymbol{d} \in Expon(E)} n_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}} & = & m \\ \sum_{\boldsymbol{d} \in Expon(E)} \alpha_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}} & = & \alpha \\ \sum_{\boldsymbol{d} \in Expon(E)} \beta_{\boldsymbol{d}}\, \boldsymbol{x}^{\boldsymbol{d}} & = & \beta\ . \end{cases}$$

et $\mathrm{Sol}^+(E) = \mathrm{Sol}^+(\mathscr{T}(E))$

$$F_1 : \begin{cases} (2,1,1)xy + (3,1,1)x^2y^2 + (2,1,1)x^3y^3 = (7,3,3) \\ (5,1,1)xy + (2,3,1)x^2y^2 + (2,2,1)x^3y^3 = (9,6,3) \end{cases}$$

$$\mathscr{T}(F_1) : \begin{cases} 2xy + 3x^2y^2 + 2x^3y^3 = 7 \\ xy + x^2y^2 + x^3y^3 = 3 \\ xy + x^2y^2 + x^3y^3 = 3 \\ 5xy + 2x^2y^2 + 2x^3y^3 = 9 \\ xy + 3x^2y^2 + 2x^3y^3 = 6 \\ xy + x^2y^2 + x^3y^3 = 3 \end{cases}$$

Solutions de $F_1 = \{(x, 1/x) \mid x \geq 0\}$

Rappelons $\mathscr{C}\mathscr{C}(F_1)$ est

$$\begin{cases} xy + x^2y^2 + x^3y^3 - 3 = 0 & xy + 3x^2y^2 + 2x^3y^3 - 6 = 0 \\ xy + 2x^2y^2 + x^3y^3 - 4 = 0 & 4xy - x^2y^2 - 3 = 0 \\ xy + x^2y^2 + x^3y^3 - 3 = 0 & xy + x^2y^2 + x^3y^3 - 3 = 0 \\ 3xy + 4x^2y^2 + 3x^3y^3 - 10 = 0 & 6xy + 3x^2y^2 + 3x^3y^3 - 12 = 0 \end{cases}$$

Obtenir les solutions réelles de E

Solutions réelles

Impossible de passer $\widetilde{n_{\boldsymbol{d}}} x^{\boldsymbol{d}}$ en paramétrique sans connaître le signe de $x^{\boldsymbol{d}}$

➥ la forme paramétrique dépend du signe du scalaire :

$$q \cdot \tilde{n} = \begin{cases} [q \cdot \underline{n},\ q \cdot \overline{n}] & \text{si } q \geq 0, \\ [q \cdot \overline{n},\ q \cdot \underline{n}] & \text{si } q \leq 0 \end{cases}$$

Idée

Introduire un k-uplet artificiel $I \in \{-1, 1\}^k$ pour les signes des indéterminés.

➤ $I^{\boldsymbol{d}}$ multiplié par le coefficient $\widetilde{n_{\boldsymbol{d}}}$ et $\boldsymbol{x}$ supposé positif

Theorem

Soit (S) un système flou à coefficients dans une même famille $\mathfrak{F}(L, L)$.

$$\mathrm{Sol}(S) = \bigcup_{I \in \{-1,1\}^k} I \otimes \mathrm{Sol}^{+}(\mathscr{T}(S(I))).$$

Implémenter : calcul des solutions réelles

Version naïve de l'algorithme : 2^k systèmes à résoudre. Certaines sont identiques.

Automatiser la reconnaissance de systèmes induits identiques

Optimisation de l'algorithme

- identifier les systèmes avec des coefficients identiques (matrice de signe)
- détection rapide de systèmes induits identiques à une permutation des équations près

j	I	lb[j]	SPos[j]	returned solutions
0	[-1,-1]	0	set([(1/y, 'R+')])	set([(1/y, 'R-')])
1	[-1,1]	1	set([])	set([])
2	[1,-1]	1	SPos[1]	set([])
3	[1,1]	0	SPos[0]	set([(1/y, 'R+')])

Les solutions non vides sont ajoutées dans la variable `sol` contenant les solutions réelles du système flou F_1 :

```
sol =  set([(1/y, 'R+'), (1/y, 'R-')])
```

Nous trouvons bien la variété $V(F_1) = \{(x = \frac{1}{y}, y) \mid y \in \mathbb{R}\backslash\{0\}\}$.

Synthèse

- Renforcer l'approche de résolution : de nouveaux résultats indépendants des fonctions de dispersion L-R.
  - ✓ pas nécessaire de développer des calculs intermédiaires sur les représentations paramétriques
- Introduction de la transformation réelle T(S)
  - ✓ système tranché collecté à coefficients réels avec seulement 3s équations au lieu de 4s.
- Extension des résultats aux familles de nombres flous non triangulaires
- Trouver les solutions réelles de systèmes à partir des solutions positives de systèmes induits
  - ✓ introduction d'un algorithme qui réduit le nombre de systèmes à résoudre

Perspectives

- Une parallélisation de l'algorithme est possible

Plan

- Construction de systèmes PMNS pour un premier donné
- Randomisation de l'arithmétique sur le PMNS
- Présentation d'un sytème de représentation hybride
- Résolution réelle des systèmes polynomiaux flous
- Conclusion

Conclusion

- Introduction de nouveaux systèmes PMNS, avec une arithmétique efficace
  - ✓ systèmes avec leurs propres propriétés de calcul, méthodes pour déterminer le nombre de systèmes constructibles
- Exploitation de la redondance intrinsèque des PMNS
  - ✓ randomiser les représentations des donneés au cours des calculs, résistance à des attaques SCA et à des attaques spécifiques de points pour ECC
- Introduction d'un système hybride, avec une arithmétique plus efficace
  - ✓ fonctionne pour tous les premiers, extensible au calculs en ECC, possibilité d'introduire des représentations redondantes
- Méthode de résolution réelle des systèmes flous avec moins d'équations
  - ✓ résultats étendus à tous les nombres flous de type L-R, automatisation de la recherche des solutions, optimisation de l'algorithme

Merci !