D34: Méthodes de calcul efficaces et sécurisées

Nicolas Méloni

Master 2: 1er semestre
(2014/2015)

Représentation des nombres

Le corps $\mathbb{F}_p$

- Ensemble d'entiers: $\{0, 1, 2, \ldots, p-1\}$
- On y effectue les opérations classiques: $+, -, \times, \div$
- Les résultats opérations multiprécisions sont réduits modulo p

Approches

- Algorithmes de réduction modulaire généralistes
- Algorithmes de réduction spécifiques

Dans la suite on considère un entier m de $2n$ bits à réduire modulo p un premier de n bits.

Division euclidienne

- On utilise l'algorithme de division multiprécision pour calculer $m = pq + r$ où $r \equiv m \mod p$
- Compléxité: $O(n^2)$

Algorithme de Barrett (1986)

- On cherche à évaluer le quotient $\left\lfloor \frac{m}{p} \right\rfloor$ pour pouvoir calculer:

$$m \mod p = m - p \times \left\lfloor \frac{m}{p} \right\rfloor.$$

- On remarque que:

$$2^{n+1} \frac{m}{p} = \frac{2^{2n}}{p} \times \frac{m}{2^{n-1}}.$$

- On évalue le quotient approché:

$$Q = \left\lfloor \frac{\left\lfloor \frac{2^{2n}}{p} \right\rfloor \left\lfloor \frac{m}{2^{n-1}} \right\rfloor}{2^{n+1}} \right\rfloor, \text{ tel que } m - pQ \leq 2p < 3p.$$

Algorithme de Barrett (1986)

Algorithm 1 Réduction de Barrett

Require: $m < 2^{2n}$ et $2^{n-1} \leq p < 2^n$, $q' = \left\lfloor \frac{2^{2n}}{p} \right\rfloor$

Ensure: $r \equiv m \mod p$

1: $m' \leftarrow \left\lfloor \frac{m}{2^{n-1}} \right\rfloor$
2: $Q \leftarrow \frac{q' \times m'}{2^{n+1}}$
3: $r \leftarrow m - pQ$
4: **while** $r > p$ **do**
5: $\quad r \leftarrow r - p$
6: **end while**
7: **return** r

Algorithme de Montgomery

- L'algortihme de Barrett cherche Q tel que les parties hautes de m et $Q \times p$ correspondent:

$$\begin{array}{cc|c} & 10101\ldots10110|10011\ldots10111 & (m) \\ - & 10101\ldots10110|00111\ldots00001 & (Qp) \\ = & 00000\ldots00000|01100\ldots10110 & (m-Qp) \end{array}$$

Algorithme de Montgomery

- L'algorithme de Montgomery repose sur une approche duale en recherchant Q tels que les parties basses de m et $Q \times p$ correspondent:

$$\begin{array}{cc|c} & 10101\ldots10110|10011\ldots10111 & (m) \\ \hline - & 01101\ldots10001|10011\ldots10111 & (Qp) \\ \hline = & 01000\ldots00101|00000\ldots00000 & (m-Qp) \end{array}$$

- Pour cela on calcule $Q = m \times p^{-1} \mod 2^n$
- On calcule alors $R = \frac{m-Qp}{2^n} \equiv m\left(2^n\right)^{-1} \mod p.$

Algorithme de Montgomery

- Récupérer la véritable valeur de $m \mod p$ est couteux
- On introduit alors la représentation de Montgomery dans laquelle un A est représenté par $A' \equiv A2^n \mod p$.
- La représentation est stable par multiplication et réduction de Montgomery:

$$A'B' \equiv AB2^n \equiv (AB)' \mod p.$$

On peut donc accomplir tout une exponentiation modulaire dans cette représentation et revenir en représentation standard uniquement à la fin.

Multiplication de Montgomery

Algorithm 2 Multiplication de Montgomery

Require: $2^{n-1} \leq P = \sum_{i=0}^{n-1} p_i 2^i < 2^n$ et $A = \sum_{i=0}^{n-1} a_i 2^i, B = \sum_{i=0}^{n-1} b_i 2^i < p$

Ensure: $R \equiv AB2^{-n} \mod P$

1: $R \leftarrow 0$
2: **for** $i = 0..n$ **do**
3: $\quad q \leftarrow r_0 + a_i b_0 \mod 2$
4: $\quad R \leftarrow (R + a_i B + qp) \div 2$
5: **end for**
6: **if** $R > p$ **then**
7: $\quad R \leftarrow R - p$
8: **end if**
9: **return** R

Moduli spécifiques

- Les algorithmes précédents sont des algorithmes généraux de réduction modulaire.
- En pratique, les corps $\mathbb{F}_p$ sont fixés une fois pour toute.
- On peut alors chercher des nombres premiers p avec une forme particulière permettant une réduction modulaire efficace.

Moduli spécifiques

Nombres de Mersennes

- Ce sont les nombres de la forme $2^n - 1$
- La réduction modulo un tel nombre est très efficace en remarquant que $2^n \equiv 1 \mod 2^n - 1$

Algorithm 3 Réduction modulaire avec nombre de Mersenne

Require: $p = 2^n - 1, m < p^2$
Ensure: $r = m \mod p$
1. $m_1 \leftarrow m/2^n$
2. $m_0 \leftarrow m \mod 2^n$
3. $r \leftarrow m_1 + m_0$
4. **if** $\leq p$ **then**
5. $\quad r \leftarrow r - p$
6. **end if**
7. **return** r

Moduli Spécifiques

Mersenne généralisés

- Les nombres de Mersenne premiers sont rares: aucun entre 2^{128} et 2^{512}
- On recherche des nombres premiers avec une forme plus générale

Nombre de Crandall

- On considère des entiers de la forme $2^n - c$ avec $c < 2^{n/2}$

Algorithm 4 Réduction modulaire avec nombre de Mersenne

Require: $p = 2^n - c, m < p^2 m, c < 2^{n/2}$
Ensure: $r = m \mod p$
1. $m_1 \leftarrow m/2^n, m_0 \leftarrow m \mod 2^n$
2. $r \leftarrow m_1 c + m_0$
3. **if** $r \geq p$ **then**
4. $\quad r_1 \leftarrow r/2^n, r_0 \leftarrow r \mod 2^n$
5. $\quad r \leftarrow r_1 c + r_0$
6. **end if**
7. **if** $r \geq p$ **then**
8. $\quad r \leftarrow r - p$
9. **end if**
10. **return** r

Moduli Spécifiques

Nombre de Solinas

- On considère des entiers de la forme $f(2^t)$ où $f(X)$ est un polynome creux.
- Ex: $f(X) = X^3 - X - 1$ et $t = 64$ donne $p_{192} = 2^{192} - 2^{64} - 1$ qui est bien premier

Algorithm 5 Réduction modulo $p_{192} = 2^{192} - 2^{64} - 1$

Require: $m = (m_5, m_4, m_3, m_2, m_1, m_0)$ en base 2^{64} avec $0 < m < p_{192}^2$
Ensure: $r \equiv m \mod p_{192}$
1. $r_1 = (m_2, m_1, m_1)$
2. $r_2 = (0, c_3, c_3)$
3. $r_3 = (c_4, c_4, 0)$
4. $r_4 = (c_5, c_5, c_5)$
5. **return** $r = r_1 + r_2 + r_3 + r_4 \mod p_{192}$