

Algorithmes pour l'arithmétique II

Cours 9

Julien Lavauzelle

Université Paris 8

Master 2 ACC et CSSD – Algorithmes pour l'arithmétique

02/02/2020

Questions ?

La **semaine dernière** : Logarithme discret dans \mathbb{G} générique

- algorithme pas de bébé, pas de géant
- algorithme de Pohlig-Hellman
 1. $n = pq$: restes chinois
 2. $n = p^k$: relèvement de Hensel
 3. analyse de complexité
- méthodes ρ et λ (=kangourous) de Pollard
- théorème de Shoup sur la complexité minimale de la résolution

Aujourd'hui :

Objectif. Résolution du problème du logarithme discret dans le groupe \mathbb{F}_q^\times .

Application. Analyse de systèmes cryptographiques à clef publique basés sur la difficulté de ce problème. Exemples :

- ElGamal pour du chiffrement
- DSA pour de la signature

1. Logarithme discret dans \mathbb{F}_q^\times

Calcul d'indice
État de l'art

2. Exercice

3. Résumé de l'année

1. Logarithme discret dans \mathbb{F}_q^\times

Calcul d'indice

État de l'art

2. Exercice

3. Résumé de l'année

Rappels.

- Un entier x est **B -lisse** / **B -friable** si tous ses facteurs premiers sont plus petits que B .
- La **proportion** d'entiers B -lisses inférieurs à N est d'environ u^{-u} où $u = \frac{\log N}{\log B}$ (théorème de Canfield, Erdős, Pomerance, très simplifié).
- Quand B est de taille moyenne, on peut trouver efficacement tous les facteurs $\leq B$ d'un entier x en temps moyen

$$O(L_{\log B}(\frac{1}{2}, \sqrt{2})^{1+o(1)} \log^d x), \text{ où } d = 2 \text{ en pratique.}$$

Méthode utilisée : **ECM de Lenstra**.

Dans ce cours, on se place dans le cas plus simple où $q = p$ est **premier**.

Problème. Étant donné g un générateur de \mathbb{F}_p^\times et $y \in \mathbb{F}_p^\times$, on cherche $x \in \{0, \dots, p-2\}$ tel que $y = g^x$.

Idée. Exprimer l'entier $x = \log_g y$ recherché en fonction de certains entiers u_i et certains $\log_g b_j$ où les b_j sont « petits ».

Comment faire ? Voici une stratégie (à raffiner) :

1. **[base de facteurs]** On se fixe une base de facteurs $\mathcal{B} = \{b_1, \dots, b_m\}$ où $b_j \in \mathbb{F}_p^\times$ est « petit »
2. **[collection de relations]** On cherche des entiers u_i tels que g^{u_i} se factorise dans \mathcal{B} :

$$g^{u_i} = b_1^{e_1} \dots b_m^{e_m} \pmod{p}$$

On a alors :

$$u_i = e_1 \log_g(b_1) + \dots + e_m \log_g(b_m)$$

3. **[algèbre linéaire]** On résout le système linéaire pour trouver tous les $\log_g(b_j)$
4. **[résolution individuelle]** On cherche v tel que yg^v s'écrit dans \mathcal{B} comme $yg^v = b_1^{f_1} \dots b_m^{f_m}$. Puis on obtient

$$\log_g y = -v + f_1 \log_g(b_1) + \dots + f_m \log_g(b_m)$$

Exemple : effritement

Présentation de l'algorithme par l'exemple.

Soit $p = 160499$. L'élément $g = 19$ est un générateur de \mathbb{F}_p^\times .

On se fixe la base de facteurs $\mathcal{B} = \{2, 3, 5, 7, 11\}$.

On calcule alors

i	g^i	factorisation	décomp. $\in \mathcal{B}$?
1	19	19	×
2	361	19^2	×
3	6859	19^3	×
4	130321	19^4	×
5	68614	$2 \times 7 \times 13^2 \times 29$	×
6	19674	$2 \times 3^2 \times 1093$	×
7	52808	$2^3 \times 7 \times 23 \times 41$	×
	...		
30	100800	$2^6 \times 3^2 \times 5^2 \times 7$	✓
	...		
174	108900	$2^2 \times 3^2 \times 5^2 \times 11^2$	✓
	...		
239	39366	2×3^9	✓
	...		
297	704	$2^6 \times 11$	✓
	...		
357	17820	$2^2 \times 3^4 \times 5 \times 11$	✓

Remarque : pas besoin de factoriser complètement en pratique.

Exemple : mise en place du système

On peut retranscrire les relations obtenues de la manière suivante.

i	g^i	factorisation
30	100800	$2^6 \times 3^2 \times 5^2 \times 7$
174	108900	$2^2 \times 3^2 \times 5^2 \times 11^2$
239	39366	2×3^9
297	704	$2^6 \times 11$
357	17820	$2^2 \times 3^4 \times 5 \times 11$

$$\Rightarrow \begin{cases} 30 = & 6\log_g 2 + 2\log_g 3 + 2\log_g 5 + \log_g 7 \\ 174 = & 2\log_g 2 + 2\log_g 3 + 2\log_g 5 + 2\log_g 11 \\ 239 = & \log_g 2 + 9\log_g 3 \\ 297 = & 6\log_g 2 + \log_g 11 \\ 357 = & 2\log_g 2 + 4\log_g 3 + \log_g 5 + \log_g 11 \end{cases}$$

On obtient donc une équation matricielle :

$$\begin{pmatrix} 6 & 2 & 2 & 1 & 0 \\ 2 & 2 & 2 & 0 & 2 \\ 1 & 9 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 1 \\ 2 & 4 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \log_g 2 \\ \log_g 3 \\ \log_g 5 \\ \log_g 7 \\ \log_g 11 \end{pmatrix} = \begin{pmatrix} 30 \\ 174 \\ 239 \\ 297 \\ 357 \end{pmatrix}$$

et on veut maintenant retrouver $\log_g 2, \log_g 3, \dots, \log_g 11$. On peut résoudre le système linéaire...

Question. Mais dans quoi (anneau) vit ce système ?

\Rightarrow Ici, on est dans $\mathbb{Z}/(p-1)\mathbb{Z}$.

Exemple : algèbre linéaire

Question : comment résoudre un système linéaire dans $\mathbb{Z}/(p-1)\mathbb{Z}$?

Comme d'habitude : restes chinois + relèvement de Hensel.

Par exemple, ici $\ell = 13$ divise $p - 1$. On résout

$$\begin{pmatrix} 6 & 2 & 2 & 1 & 0 \\ 2 & 2 & 2 & 0 & 2 \\ 1 & 9 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 1 \\ 2 & 4 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \log_g 2 \\ \log_g 3 \\ \log_g 5 \\ \log_g 7 \\ \log_g 11 \end{pmatrix} = \begin{pmatrix} 30 \\ 174 \\ 239 \\ 297 \\ 357 \end{pmatrix} \equiv \begin{pmatrix} 4 \\ 5 \\ 5 \\ 11 \\ 6 \end{pmatrix} \pmod{13}$$

Cela donne

Après résolution modulo les autres facteurs de $p - 1$, puis reconstruction, on obtient la solution

$$\begin{pmatrix} \log_g 2 \\ \log_g 3 \\ \log_g 5 \\ \log_g 7 \\ \log_g 11 \end{pmatrix} \equiv \begin{pmatrix} 6 \\ 10 \\ 5 \\ 3 \\ 1 \end{pmatrix} \pmod{13} \qquad \begin{pmatrix} \log_g 2 \\ \log_g 3 \\ \log_g 5 \\ \log_g 7 \\ \log_g 11 \end{pmatrix} = \begin{pmatrix} 120659 \\ 147118 \\ 54722 \\ 156380 \\ 78833 \end{pmatrix}$$

Remarque. $\ell = 2$ est toujours facteur de $p - 1$; pour éviter de résoudre des systèmes modulo 2 qui sont très probablement de rang faible, on ajoute artificiellement -1 dans la base de facteurs.

Exemple : résolution individuelle

Résolution individuelle. Supposons maintenant que l'on souhaite retrouver le logarithme de $y = 29750$ en base $g = 19$.

v	$y \times g^v$	factorisation	décomp. $\in \mathcal{B}$?
0	29570	$2 \times 5 \times 2957$	×
1	80333	$11 \times 67 \times 109$	×
2	81836	$2^2 \times 41 \times 499$	×
3	110393	101×1093	×
4	10980	$2^2 \times 3^2 \times 5 \times 61$	×
5	48121	48121	×
6	111804	$2^2 \times 3 \times 7 \times 11^3$	✓

Puis, on reconstruit

$$\begin{aligned}\log_g(y) &= -v + 2\log_g(2) + \log_g(3) + \log_g(7) + 3\log_g(11) \\ &= -6 + 2 \times 120659 + 147118 + 156380 + 3 \times 78833 = 139317\end{aligned}$$

Remarque. En pratique, on aurait dû utiliser l'algorithme de Pohlig-Hellman, et calculer le logarithme discret dans les sous-groupes de $\mathbb{Z}/(p-1)\mathbb{Z}$. C'est (grossièrement) équivalent à nos calculs modulo ℓ précédents.

Si on note $\pi(x, B)$ la probabilité qu'un entier $\leq x$ soit B -lisse. Alors :

1. **[base de facteurs]** Complexité en $O(B)$
2. **[collection des relations]** Les éléments g^i peuvent être considérés comme aléatoires dans \mathbb{F}_p^\times , donc complexité en $O(B/\pi(p, B))$.
3. **[algèbre linéaire]** Complexité en $O(B^{2+o(1)})$ avec Wiedemann par blocs (les matrices sont creuses).
4. **[logarithme individuel]** Les $g^v y$ sont aléatoires dans \mathbb{F}_p^\times . Donc, complexité en $O(1/\pi(p, B))$.

Donc la complexité du calcul d'indice est

$$C = \frac{B}{\pi(p, B)} + B^{2+o(1)} + \frac{1}{\pi(p, B)} \leq 2B^{1+o(1)} \max \left\{ \frac{1}{\pi(p, B)}, B \right\}$$

Cette complexité est minimale pour $B = \frac{1}{\pi(p, B)}$. Pour $\pi(p, B) = \frac{\log p}{\log B}$, on obtient après résolution

$$B = 2^{\frac{1}{\sqrt{2}}} \sqrt{\log p \log \log p}$$

Enfin, la complexité de la résolution par calcul d'indice est en

$$O \sim \left(L_{\log p} \left(\frac{1}{2}, \sqrt{2} \right) \right).$$

1. Logarithme discret dans \mathbb{F}_q^\times

Calcul d'indice

État de l'art

2. Exercice

3. Résumé de l'année

Comme pour la factorisation, on peut améliorer la complexité grâce à

- un crible dans un corps de nombres (*number field sieve, NFS*)
- un crible dans un corps de fonctions (*function field sieve, FFS*)

Dans \mathbb{F}_q où q n'est pas nécessairement premier, on atteint alors une complexité en

$$L_{\log q} \left(\frac{1}{3}, \left(\frac{128}{9} \right)^{1/3} \right)$$

Il existe des améliorations suivant la **caractéristique du corps** $\mathbb{F}_q = \mathbb{F}_{p^k}$.

On note $p = L_{\log q}(\ell_p, c_p)$.

Pour $\ell_p < 1/3$, on dit que la caractéristique est **petite**.

Pour $\ell_p > 2/3$, on dit que la caractéristique est **grande**.

Sinon, elle est **moyenne**.

En **petite caractéristique**, les améliorations sont drastiques :

- à p constant, Joux (2013) donne un algorithme en $L_{\log q}(\frac{1}{4}, c)$ où $c > 0$
- pour $p = L_{\log q}(\alpha)$, il existe un algorithme heuristique en temps quasi-polynomial, i.e. $L_{\log q}(\alpha + o(1), 1)$.

Cette amélioration a eu une conséquence pratique pour les **couplages**, utilisés récemment en cryptographie (Diffie-Hellman à trois, IBE, ...), qui étaient originellement implantés en petite caractéristique.

En **grande caractéristique**, on a « seulement » une amélioration sur l'exposant secondaire : on passe de $(\frac{128}{9})^{1/3}$ à $(\frac{32}{9})^{1/3}$ grâce un algorithme de crible par corps de nombres *spécial* (SNFS).

En **moyenne caractéristique**, les améliorations sont encore moins notables (SNFS également).

Évolution des résolutions du logarithme discret en **petite** caractéristique :

année	q	auteurs	heures CPU
1992	2^{401}	Gordon, McCurley	114 000
2005	2^{613}	Joux, Lercier	26 000
2013	2^{1778}	Joux	220
2013	2^{6168}	Joux	550
2014	2^{9234}	Granger, Kleinjung, Zumbragel	398 000

En **moyenne** caractéristique :

- \mathbb{F}_q avec $q = 33341353^{57}$ (1425 bits)
- auteur : Antoine Joux
- 32 000 heures CPU

En **grande** caractéristique :

- \mathbb{F}_{p^2} avec p premier de 160 chiffres (532 bits)
- auteurs : Barbulescu, Gaudry, Guillevic, Morain (2014)
- 68 jours CPU + 30 heures GPU

On pourrait penser **adapter** l'idée du calcul d'indice pour n'importe quel groupe.

- Cependant, il faut une notion de **friabilité**, autrement dit de « norme » pour les éléments du groupe.
- C'est difficile à définir pour les courbes elliptiques, par exemple.

Un autre groupe ? Le groupe des points d'une Jacobienne de courbe hyperelliptique ?
Notons g le genre de la courbe.


- La taille du groupe $\text{Jac}(\mathcal{C})(\mathbb{F}_q)$ est en $O(q^g)$
- Alors, il existe un algorithme (Gaudry, Thomé, Thériault, Die) résolvant le problème du logarithme discret en temps $O(\min q^{g/2}, q^{2-2/g})$


\Rightarrow on ne peut prendre que $g = 2$ ou $g = 3$, mais les opérations sont bien plus coûteuses que pour les courbes elliptiques.

Certaines courbes elliptiques sont « fragiles ». Grâce à une **descente de Weil**, on transporte le problème du logarithme discret dans le groupe de points $E(\mathbb{F}_{q^k})$, $k > 1$, dans un groupe associé à une courbe hyperelliptique \mathcal{C} définie sur \mathbb{F}_q avec $q \ll q^k$.


Conseil pour de la cryptographie : courbes elliptiques avec p (presque) premier ou courbes hyperelliptiques de genre 2. En évitant les sous-familles fragiles.


Calcul d'indice :

 *A subexponential algorithm for the discrete logarithm problem with applications to cryptography.* Adleman. FOCS. **1979**.


 *Fast, rigorous factorization and discrete logarithm algorithms.* Pomerance. Discrete algorithms and complexity. **1987**.

NFS et FFS :


 *Using number fields to compute logarithms in finite fields.* Schirokauer. Math. Comp. **2000**.

 *Function field sieve method for discrete logarithm over finite fields..* Adleman, Huang. Inf. Comput.. **1999**.

Algorithme quasi-polynomial en petite caractéristique :

 *A Heuristic Quasi-Polynomial Algorithm for Discrete Logarithm in Finite Fields of Small Characteristic.* Barbuslecu, Gaudry, Joux, Thomé. EUROCRYPT. **2014**.

Un état de l'art assez récent :

 *The Past, evolving Present and Future of Discrete Logarithm.* Joux, Odlyzko, Pierrot.
<https://members.loria.fr/CPierrot/papers/DlogSurvey.pdf>. **2014**.

1. Logarithme discret dans \mathbb{F}_q^\times

Calcul d'indice
État de l'art

2. Exercice

3. Résumé de l'année

Exercice. Soit G un groupe cyclique d'ordre q connu, et g un générateur de G .

On suppose qu'on dispose d'un algorithme A qui, en temps T , résout correctement le problème du logarithme discret sur un sous-ensemble S de G , inconnu, de taille $\geq \varepsilon q$.

Démontrer que l'on peut alors construire un algorithme B qui résout le problème du logarithme discret en temps $O(T/\varepsilon)$

Réponse. Voici l'algorithme B :

Entrée : $y \in G$

Sortie : x tel que $y = g^x$

1. tirer $c \in \{0, \dots, q-1\}$ uniformément
2. calculer $u = g^c$
3. calculer $\ell = A(u \times y)$
4. si $g^\ell = u \times y$, alors retourner $\ell - c$
5. sinon, revenir à l'étape 1.

Il effectue $O(1/\varepsilon)$ itérations, sa complexité est donc en $O(T/\varepsilon)$.

1. Logarithme discret dans \mathbb{F}_q^\times

Calcul d'indice
État de l'art

2. Exercice

3. Résumé de l'année

Factorisations matricielles :

- ▶ factorisation LU (par Gauss)
- ▶ factorisation QR (par Gram-Schmidt)
- ▶ factorisation de Choleski

LFSR :

- ▶ LFSR, complexité linéaire
- ▶ Algorithme de Berlekamp-Massey pour la reconstruction du polynôme de connexion d'un LFSR

Suites récurrentes linéaires :

- ▶ Polynôme minimal d'une suite récurrente linéaire, d'une matrice
- ▶ Algorithme d'Euclide étendu pour la reconstruction d'un polynôme minimal.
- ▶ Application à la résolution de systèmes linéaires creux en $o(n^3)$
- ▶ Algorithme de Wiedemann. Mentions des algorithmes de Wiedemann par bloc et Lanczos.

Calcul de racines carrées :

- ▶ Dans \mathbb{Z} (entières) : algorithme « scolaire », méthode de Newton
- ▶ Dans \mathbb{F}_q :
 - rappels de calculs de résidus quadratiques (symboles de Jacobi et Legendre)
 - méthodes de Shanks ($q \equiv 1[4]$), Atkin ($q \equiv 5[8]$)
 - algorithme de Cipolla
 - algorithme de Tonelli-Shanks (en DM)
- ▶ Dans $\mathbb{Z}/N\mathbb{Z}$, où la factorisation de N est connue
 - théorème des restes chinois si $N = pq$
 - relèvement de Hensel si $N = p^k$ (cas p pair et p impair à distinguer)

Factorisation de polynômes dans $\mathbb{F}_q[X]$:

- ▶ élimination des facteurs carrés en caractéristique p
- ▶ algorithme de Berlekamp :
 - algèbre et théorème de Berlekamp
 - cardinalité petite et grande
- ▶ algorithme de Cantor-Zassenhaus :
 - factorisation en degré distincts
 - factorisation à degré égal

Factorisation de polynômes dans $\mathbb{Q}[X]$:

- ▶ factoriser dans $\mathbb{Z}[X] \iff$ factoriser dans $\mathbb{Q}[X]$ et factoriser dans \mathbb{Z}
- ▶ contenu, polynôme primitif
- ▶ factorisation de polynômes primitifs dans $\mathbb{Z}[X]$
 - réduction au problème dans $\mathbb{F}_p[X]$
 - borne de Landau-Mignotte
 - relèvement de Hensel
 - phase de regroupement des facteurs (exponentielle), mention de LLL pour obtenir un algorithme polynomial

Factorisation dans \mathbb{Z} :

- ▶ méthode de divisions successives, algorithme de Fermat
- ▶ algorithme ρ de Pollard, algorithme de Floyd
- ▶ méthode $p - 1$ de Pollard, méthode $p + 1$ de Williams pour des facteurs tels que $p - 1$ ou $p + 1$ est superfriable
- ▶ aperçu de la méthode ECM de Lenstra

Factorisation dans \mathbb{Z} :

- ▶ notation L pour la complexité des fonctions sous-exponentielles
- ▶ algorithme de Kraitchik/Dixon
- ▶ crible quadratique :
 1. choix de la base de facteurs
 2. effritement par crible (inclut un calcul de racine carrée...)
 3. reconstruction par algèbre linéaire creuse
 4. analyse de complexité
- ▶ mention du crible algébrique (par corps de nombres)

Logarithme discret dans G générique

- ▶ algorithme pas de bébé, pas de géant
- ▶ algorithme de Pohlig-Hellman
 1. $n = pq$: restes chinois
 2. $n = p^k$: relèvement de Hensel
 3. analyse de complexité
- ▶ méthodes ρ , λ (=kangourous) de Pollard
- ▶ théorème de Shoup sur la complexité minimale de la résolution

Questions ?