# FACTORISATION ET LOGARITHME DISCRET

RÉSUMÉ ET QUESTIONS

## 1. Logarithme discret dans un groupe

On rappelle qu'un groupe est un ensemble $G$ non-vide muni d'une loi interne associative (notée $\times$, . ou rien du tout) ayant un élément neutre $e$ et telle que tout élément $a$ admet unique inverse $b$ tel que $ab = ba = e$. Si la loi est commutative on dit que le groupe est commutatif. Dans ce cas, le neutre est souvent noté $0$ et la loi est alors notée $+$. Si la loi est non-commutative, ou lorsqu'on souhaite éviter des confusions, elle peut-être notée $\times$ et le neutre est alors noté $1$. On dit qu'un groupe fini $G$ est *cyclique* s'il admet un *générateur*. Autrement dit, si $G = \{1, g, g^2, g^3, \ldots\}$ pour un certain $g$.

**Exercice.** Pour chacun des groupes suivants, dire s'il est cyclique, et si tel est le cas donner un générateur : $(\mathbb{Z}/5\mathbb{Z}, +)$, $((\mathbb{Z}/7\mathbb{Z})^*, \times)$, $((\mathbb{Z}/35\mathbb{Z})^*, \times)$, $(\mathbb{Z}, +)$, $(\mathbb{R}, +)$.

On pourra ensuite décrire l'ensemble des générateurs de chacun de ces groupes et étudier leurs relations (comment ils s'expriment les uns en fonction des autres).

□

Soit $G$ un groupe cyclique fini. On note $e$ son cardinal. Soit $g$ est un générateur de $G$. On appelle exponentielle discrète de base $g$ l'application $\exp_g : \mathbb{Z}/e\mathbb{Z} \to G$ définie par $\exp_g(k) = g^k$. C'est une bijection. L'application réciproque est notée $\log_g$ et appelé logarithme discret de base $g$. On vérifie que $\log_g(ab) = \log_g(a) + \log_g(b)$ et si $h$ est un autre générateur alors $\log_h(a) = \log_g(a)/\log_g(h)$. Et $\exp_g(k + l) = \exp_g(k)\exp_g(l)$. L'algorithme d'exponentiation rapide déjà présenté dans le contexte du groupe cyclique $(\mathbb{Z}/p\mathbb{Z})^*$ se généralise à tout groupe cyclique $G$. Il semble que l'inventeur de cet algorithme soit le poète Indien Piṅgala dans son Chandah-sûtra (avant -200). Voir [DatSin, I,13]. Cet algorithme admet de nombreuses variantes [Gor]. Il n'existe pas d'algorithme générique pour calculer efficacement le logarithme discret [Sho]. Mais pour certaines familles de groupes ce problème n'est pas difficile du tout. C'est le cas par exemple des groupes $\mathbb{Z}/N\mathbb{Z}$ additifs.

Si $G$ est cyclique de cardinal $e$, et si $h_1$ et $h_2$ sont deux générateurs de $G$, alors $h_2 = h_1^a$ avec $a \in (\mathbb{Z}/e\mathbb{Z})^*$. Posant $b = 1/a \in (\mathbb{Z}/e\mathbb{Z})^*$ on note que $h_1 = h_2^b$. On note $H$ l'ensemble des générateurs de $G$ et $A = (\mathbb{Z}/e\mathbb{Z})^*$ l'ensemble des exposants inversibles. Ces deux ensembles ont le même cardinal. Et $A$, qui est un groupe, agit sur $H$. Pour tout couple $(h_1, h_2)$ d'éléments de $H$ il existe un unique exposant $a$ dans $A$ tel que $h_2 = h_1^a$.

**Exercice.** Soit $p = 31$. Montrez que $G = (\mathbb{Z}/p\mathbb{Z})^*$ a un unique sous-groupe d'ordre $2$. On le note $G_1$.

Montrez que $G$ a un unique sous-groupe $G_2$ d'ordre $3$. Et $G_2$ est le sous-groupe de $G$ formé des puissances dixièmes dans $G$.

Montrez que $G$ a un unique sous-groupe $G_3$ d'ordre $5$. Et $G_3$ est le sous-groupe de $G$ formé des puissances sixièmes.

Montrez que $G$ est le produit direct de $G_1$, $G_2$ et $G_3$.

□

**Exercice.** Soit $p$ le nombre

$$171962010545840643348334056831754301958457563589574256043877110505832165523$$
$$856261308397965147955578800999455782202456522693290629520826275682227566369411 1$$

Vérifiez que $p$ est probablement premier, à l'aide du test de Miller-Rabin.

Vérifiez que $p-1$ est un entier $400$-friable. En déduire un algorithme rapide pour calculer les logarithmes discrets dans $(\mathbb{Z}/p\mathbb{Z})^*$.

Qu'en déduisez vous ?

□

## 2. Identification

Le protocole suivant est dû à Schnorr. Sa sécurité repose sur la difficulté du logarithme discret dans un groupe cyclique fini $G$. Alice génère un secret. Puis elle se prouve auprès de Bob.

- Alice choisit $G$ cyclique de cardinal $e$. On note $H$ l'ensemble des générateurs de $G$, et $A$ l'ensemble des exposants inversibles modulo $e$
- Alice choisit $h_0 \in H$ et $a_{Alice} \in A$ et calcule $h_{Alice} = {h_0}^{a_{Alice}}$. Elle publie $G, h_0, h_{Alice}$.
- pour se prouver auprès de Bob, elle choisit un exposant $a_r$ au hasard dans $G$ et calcule $h_r = {h_{Alice}}^{a_r} = {h_0}^{a_{Alice}a_r}$ et elle envoie $h_r$ à Bob.
- Bob tire $\epsilon \in \{0,1\}$ au hasard
- si $\epsilon = 0$ Bob demande à Alice quel est l'exposant qui envoie $h_0$ sur $h_r$ et Alice doit répondre $a_r a_{Alice} \bmod e$
- si $\epsilon = 1$ Bob demande à Alice quel est l'exposant qui envoie $h_{Alice}$ sur $h_r$ et Alice doit répondre $a_r$
- preuve sans apport d'information

## 3. Chiffrement d'El Gamal

C'est un chiffrement asymétrique. Il repose sur la difficulté du logarithme discret dans un groupe $G$. Alice a généré sa clé publique et sa clé privé. Bob lui écrit.

(1) Alice a choisi $G$ cyclique de cardinal $e$. On note $H$ l'ensemble des générateurs de $G$, et $A$ l'ensemble des exposants inversibles modulo $e$.
(2) Alice a choisi $h_0 \in H$ et $a_{Alice} \in A$. Elle a calculé $h_{Alice} = {h_0}^{a_{Alice}}$ et a publié $G$, $h_0$ et $h_{Alice}$.
(3) Bob veut envoyer $m$ à Alice. Bob trouve $(G, h_0, h_{Alice})$ dans l'annuaire. Il choisit $a_{Bob}$ dans $A$ et l'applique à $h_0$ ce qui donne $k = {h_0}^{a_{Bob}}$. Il calcule aussi $t = {h_{Alice}}^{a_{Bob}}$ et $c = m \oplus t$
(4) Bob envoie $(k, c)$
(5) Alice calcule $k^{a_{Alice}} = h_0^{a_{Bob}a_{Alice}} = h_0^{a_{Alice}a_{Bob}} = h_{Alice}^{a_{Bob}} = t$ et $m = c \ominus t$

(6) Le chiffré est deux fois plus long que le clair mais il n'y a qu'un échange

Notons que la sécurité des protocoles de Schnorr et El Gamal repose sur un problème un peu plus faible que le logarithme discret. C'est le problème de Diffie et Hellman : étant donnés $G$, $h_0$, $h_1$ et $h_2$, trouver l'unique $h_3$ tel que $h_1 = h_0^{a_1}$, $h_2 = h_0^{a_2}$, $h_3 = h_0^{a_3}$, et $a_3 = a_1 a_2 \in A$.

## 4. Recherche d'un générateur

Afin de mettre en oeuvre les protocoles cryptographiques à base de logarithme discret, on doit trouver un groupe cyclique $G$ et un générateur de $G$. Une autre manière de poser le problème est de chercher dans un groupe $\mathbf{G}$ un élément $g$ d'ordre assez grand et non-friable. On pose ensuite $G =< g >$ le groupe engendré par $g$. On doit pouvoir prouver que l'ordre $e$ de $g$ est grand et non-friable. Il faut donc connaître cet ordre. Et il faut pouvoir le factoriser. On verra qu'il est difficile de factoriser un nombre quelconque. Mais si un nombre est premier, alors on peut facilement s'en convaincre à l'aide du test de Miller-Rabin. Et la factorisation s'arrête là.

En pratique, on part d'un groupe $\mathbf{G}$ d'origine arithmétique. Par exemple $\mathbf{G} = (\mathbb{Z}/p\mathbb{Z})^*$ ou bien $\mathbf{G} = E(K)$ le groupe des points d'une courbe elliptique sur un corps fini $K$. Dans ces deux cas, on connaît le cardinal $\#\mathbf{G}$ de $\mathbf{G}$. Si ce cardinal est premier, n'importe quel élément différent de l'élément neutre est un générateur. Cette situation ne se produit jamais pour $(\mathbb{Z}/p\mathbb{Z})^*$ car $p-1$ est pair si $p \geq 3$. On peut naturellement se restreindre aux premiers $p$ tels que $(p-1)/2$ soit lui aussi premier. Dans ce cas, on pose $q = (p-1)/2$ et on cherche un élément d'ordre $q$ dans $\mathbf{G} = (\mathbb{Z}/p\mathbb{Z})^*$. L'ensemble des éléments $x$ de $\mathbf{G}$ tels que $x^q = 1$ est un groupe : c'est l'unique sous-groupe de $\mathbf{G}$ d'ordre $q$. C'est aussi l'ensemble des carrés de $\mathbf{G}$. On note $G$ ce sous-groupe. Pour obtenir un élément de $G$, on choisit $y$ au hasard dans $\mathbf{G}$ et l'on pose $x = y^2$. Si $x \neq 1$ alors il engendre $G$.

Plus généralement, on est satisfait si le cardinal de $\mathbf{G}$ s'écrit

$$\#\mathbf{G} = q \prod_{\ell \text{ petit premier}} \ell.$$

Dans ce cas, il est très facile de factoriser $\#\mathbf{G}$ : on divise par tous les petits premiers et si le quotient est un grand nombre premier on a gagné. En cas d'échec il ne reste plus qu'à changer de groupe.

On pourra implémenter cette méthode pour les groupes $\mathbf{G} = (\mathbb{Z}/p\mathbb{Z})^*$. On commence par cherche un pseudo-premier $p$ tel que $p-1$ est le produit de petits premiers par un grand pseudo-premier $q$. Si $p - 1 = qL$ avec $L$ le produit des petits premiers, on choisit un $y$ au hasard dans $(\mathbb{Z}/p\mathbb{Z})^*$ et on pose $x = y^L$. Si $x \neq 1$ alors il est un générateur du seul sous-groupe $G$ d'ordre $q$ dans $(\mathbb{Z}/p\mathbb{Z})^*$.

## 5. Premières attaques sur le logarithme discret

La première attaque sur le logarithme discret est l'attaque exhaustive. Pour calculer $\log_g h$ on calcule les puissances successives de $g$ jusqu'à ce qu'on trouve $h$.

On a vu aussi que le logarithme discret est trivial dans les groupes additifs $\mathbb{Z}/N\mathbb{Z}$ et aussi dans les groupes dont l'ordre est friable.

La meilleure attaque sur les groupes générique est due à Shanks. C'est la méthode des pas de bébé/pas de géant. On peut l'illustrer très simplement sur le groupe $G = (\mathbb{Z}/101\mathbb{Z})^*$. On vérifie que $g = 2 \bmod 101$ est un générateur. S'il ne l'était pas, il serait contenu dans un sous groupe de $G$. Donc son ordre serait un diviseur strict de $\#G = 100$. Les diviseurs stricts maximaux de $100$ sont $20 = 100/5$ et $50 = 100/2$. Or $2^{50} = -1 \bmod 101$ et $2^{20} = 95 \bmod 101$. On pose $r = \lceil 100 \rceil = 10$ et $\gamma = g^r = 14 \bmod 101$. Posons $h = 48 \bmod 101$. On veut calculer $\log_g h \in \mathbb{Z}/100\mathbb{Z}$. On calcule la liste des $\gamma^k$ pour $0 \leq k \leq r-1$ soit

| $k$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| $\gamma^k$ | 1 | 14 | 95 | 17 | 36 | 100 | 87 | 6 | 84 | 65 |

On calcule maintenant la liste des $hg^{-l}$ pour $0 \leq l \leq r-1$ soit

| $l$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| $hg^{-l}$ | 48 | 24 | 12 | 6 | 3 | 52 | 26 | 13 | 57 | 79 |

Ces deux listes ont un élément en commun qui est $6$. Il correspond aux valeurs $k = 7$ et $l = 3$. Donc $\gamma^7 = hg^{-3}$ donc $h = g^{73}$.

Le temps de calcul de cette méthode est le temps nécessaire pour constituer et trier les deux listes de longueur $O(\sqrt{e})$. Ce temps est $e^{1/2+o(1)}$ en utilisant les algorithmes de tri rapides comme le tri par insertion.

## 6. FACTORISATION DES ENTIERS

On a vu qu'il est assez facile de se convaincre qu'un nombres est premier ou de prouver qu'il est composé. En revanche, on ne connaît pas d'algorithme général efficace pour calculer la décomposition en produit de facteurs premiers d'un entier. En particulier, si $N = pq$ avec $p$ et $q$ des premiers de deux mille bits chacun, on ne sait pas retrouver $p$ et $q$ à partir de $N$ efficacement. Nous allons quand même examiner les algorithmes les plus simple pour résoudre ce problème. Et nous évaluerons leur complexité. C'est un travail nécessaire car la sécurité de nombreux cryptosystèmes repose sur la difficulté de la factorisation d'entiers.

La première méthode consiste à calculer la division euclidienne de $N$ par $2$, $3$, $5$, $\ldots$, et tous les nombres impairs en suivant. On trouve ainsi le premier diviseur non-trivial de $N$. C'est toujours un nombre premier. Si l'on n'a pas trouvé de diviseur inférieur ou égal à $\sqrt{N}$ on peut s'arrêter car $N$ est alors premier (ceci ne devrait pas se produire car on a vérifié que $N$ est composé avant de chercher à le factoriser). Une fois trouvé un facteur premier $p$ de $N$, on remplace $N$ par $N/p$ et on recommence. Le temps de calcul est $N^{1/2+o(1)}$ ce qui est très mauvais. Nous verrons de meilleurs algorithmes plus tard. Mais aucun n'est polynomial en temps.

## 7. LE SYSTÈME RSA

Le système de Rabin repose sur la difficulté de calculer des racines carrées modulo $N$, ce qui est équivalent à factoriser $N$. Le système RSA est une généralisation du système de Rabin.

Si $p$ et $q$ sont deux premiers impairs on pose $N = pq$ et $\lambda = \mathrm{ppcm}(p-1, q-1)$. On vérifie que pour tout $x$ dans $(\mathbb{Z}/N\mathbb{Z})^*$ on a $x^\lambda = 1 \bmod N$. C'est une affirmation un peu plus forte que le théorème d'Euler. Elle est vraie parce que $(\mathbb{Z}/N\mathbb{Z})^*$ est isomorphe à $(\mathbb{Z}/p\mathbb{Z})^* \times (\mathbb{Z}/q\mathbb{Z})^*$. C'est le théorème des restes chinois. Donc tout élément $x$ de $(\mathbb{Z}/N\mathbb{Z})^*$ s'écrit $x = x_p x_q$ avec $x_p \in (\mathbb{Z}/p\mathbb{Z})^*$ et $x_q \in (\mathbb{Z}/q\mathbb{Z})^*$. Le résultat s'en déduit.

**Exercice.** Donnez un exemple explicite de la décomposition $x = x_p x_q$ pour $p = 5$ et $q = 7$.

□

On donne maintenant un exemple de RSA. Alice génère sa clé. Puis Bob lui écrit.

Alice prépare ses clés. Elle choisit deux premiers $p$ et $q$ au hasard. Par exemple avec pari/gp

```
gp >p=nextprime(random(2^20))
%1 = 761669
gp >q=nextprime(random(2^20))
%2 = 341729
```

Elle calcule $n = pq$ et $\lambda(n) = \mathrm{ppcm}(p-1, q-1)$.

```
gp > n=p*q
%3 = 260284385701
gp > L=lcm(p-1,q-1)
%4 = 65070820576
```

Elle choisit un exposant de chiffrement $e$ premier à $\lambda(n)$. Par exemple si elle ne craint pas un petit $e$ elle peut prendre $e = 3$ car $3$ est premier à $\lambda$ ici :

```
gp > e=3
%5 = 3
gp > gcd(e,L)
%6 = 1
```

Elle calcule l'inverse $f$ de $e \bmod \lambda(n)$

```
gp > f=(1/e)% L
%7 = 43380547051
```

Elle publie $n$ et $e$. C'est la clé de chiffrement.

Elle garde secret l'exposant de déchifrement $f$ : c'est la clé de déchiffrement.

Bob veut écrire à Alice. Son message clair est $m = 1234567$.

Il trouve dans l'annuaire la clé publique $n = 260284385701$ et $e = 3$. Il calcule $c = m^e \bmod n$.

```
gp > m=1234567
%1 = 1234567
gp > n=260284385701
%2 = 260284385701
gp > e=3
```

```
%3 = 3
gp > c=Mod(m,n)^e
%4 = Mod(214733022870, 260284385701)
```

Il envoie $c = 214733022870$ à Alice.

Alice reçoit le message chiffré $c$ de Bob. Elle calcule $c^f \bmod n$ car elle connaît la clé secrète de déchiffrement $f$.

```
gp > c=214733022870
%1 = 214733022870
gp > f=43380547051
%2 = 43380547051
gp > n=260284385701
%3 = 260284385701
gp > Mod(c,n)^f
%4 = Mod(1234567, 260284385701)
```

On voit que la sécurité du système RSA repose sur la difficulté de calculer $f = 1/e \bmod \lambda(n)$ si on ne connaît pas $p$ et $q$.

En pratique RSA ne doit pas être implémenté ainsi. Il faut au minimum ajouter au message un suffixe aléatoire.

**Exercice.** Expliquer pourquoi Alice retrouve $m$ en élevant $c$ à la puissance $f$.

Quelle est la complexité du chiffrement RSA ? du déchiffrement ?

□

## 8. La méthode $p-1$ de Pollard

Cette méthode de factorisation est efficace seulement dans des cas très particulier. Cependant elle introduit quelques idées intéressantes. C'est une bonne raison pour la présenter. On suppose que $N$ est produit de deux premiers $p$ et $q$ et que $p-1$ est un nombre friable. Par exemple prenons $n = 713 = pq$ avec $p = 31$ et $q = 23$. Alors $p - 1 = 2 \times 3 \times 5$ est $B$-friable avec $B = 5$.

On choisit un résidu $x$ modulo $n$ au hasard. Par exemple $x = 2 \bmod n$. On définit une suite en posant $x_1 = x$, $x_2 = x_1^2 \bmod n$, $x_3 = x_2^3 \bmod n$, $x_4 = x_3^4$, $x_5 = x_4^5 \bmod n$, ... Ainsi $x_{k+1}$ est obtenu en élevant $x_k$ à la puissance $k + 1$ modulo $n$. Autrement dit $x_k = x^{k!}$. Pour tout $k$ on calcule le pgcd de $x_k - 1$ et de $n$. Si l'un des facteurs $p$ de $n$ est friable alors il est probable que $p-1$ divise $k!$ pour un petit $k$. On obtiendra donc un pgcd non-trivial. En effet $x^{p-1} = 1 \bmod p$.

Sur notre exemple on obtient

| $k$ | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $x_k = x^{k!} \bmod n$ | 2 | 4 | 64 | 326 | 311 | 32 | 280 |
| $\mathrm{pgcd}(x_k - 1, n)$ | 1 | 1 | 1 | 1 | 31 | 31 | 31 |

Cette méthode est intéressant si $n$ a un petit facteur premier. En effet, si $p$ est petit alors $p-1$ a beaucoup de chances d'être friable.

Il existe une variante de la méthode $p-1$ de Pollard, appelée *large prime variation*. Elle aboutit si $p-1$ est le produit de premiers $\leq B$ et d'au plus un premier dans l'intervalle $]B,C]$. Un tel nombre est dit $(B,C)$-friable. On commence comme précédemment. On calcule $x_{k+1} = x_k^{k+1} \bmod n$ jusqu'à $x_B$. Ensuite on calcule $y_1 = x_B^{l_1} \bmod n$, $y_2 = x_B^{l_2} \bmod n$, ..., où les $l_i$ sont les premiers dans $]B,C]$ par ordre croissant. L'astuce est que pour calculer $y_2$ on n'a pas besoin d'une exponentiation modulaire. En effet

$$y_2 = x_B^{l_2} = x_B^{l_1+(l_2-l_1)} = y_1 x_B^{l_2-l_1} \bmod n.$$

Or la différence entre deux nombres premiers consécutifs est très petite. On calcule donc une fois pour toute les premières puissance de $x_B$ soit $z_j = x_B^j$ et on a

$$y_{k+1} = y_k z_{l_{k+1}-l_k} \bmod n$$

qui se calcule au prix d'une seule multiplication.

Par exemple, si $n = 2721749 = pq$ avec $p = 2671$ et $q = 1019$ alors $p-1 = 2 \times 3 \times 5 \times 89$ est $(B,C)$-friable avec $B = 7$ et $C = 100$ par exemple.

On calcule $x_1 = 2$, $x_2 = x_1^2 = 4$, $x_3 = 64$, $x_4 = 446722$, $x_5 = 1416863$, $x_6 = 715291$, $x_7 = 795854$.

On pose $y = x_7$. On élève $y$ à la puissance $l_i$ pour tous les premiers $l_i$ entre $11$ et $97$. On voit sans peine que la différence entre deux tels premiers consécutifs est au plus de $8$.

On calcule donc les huit premières puissances de $y$ soit $z_1 = y = 795854$, $z_2 = y^2 = 2657777$, $z_3 = y^3 = 664706$, $z_4 = 1628037$, $z_5 = 2034144$, $z_6 = 1664270$, $z_7 = 1281471$, $z_8 = 2696942$.

On peut maintenant calculer $y_1 = y^{11} = 1715449$, $y_2 = y^{13} = y_1 z_2 = 216252$, $y_3 = y^{17} = y_2 z_4 = 2580676$, ..., $y_{20} = y^{89} = y_{19} z_6 = 1279410$.

On trouve que $\mathrm{pgcd}(y_{20} - 1, n) = 2671$.

## REFERENCES

[DatSin] B. Datta and A.N. Singh. *History of Hindu Mathematics.* Motilal Banarsi Das, Lahore, 1935.

[Gor] D. M. Gordon. *A Survey of Fast Exponentiation Methods.* J. Algorithms 27(1): 129-146 (1998)

[Sho] V. Shoup *Lower bounds for discrete logarithms and related problems.* Lecture Notes in Computer Science. 1233. Advances in Cryptology — Eurocrypt 97. Springer-Verlag. pp. 256–266 (1997).