

Algorithme rho de pollard

TAFFOREAU Nicolas

17 février 2012

Table des matières

1	Rapel sur les courbes elliptique	2
2	Algorithme rho de Pollard	3
2.1	dans un groupe du type F_p	3
2.2	sur les courbe elliptique	3
2.3	algorithme de floyd	4
3	Negation map	5
3.1	methode general	5
3.2	cycle infructueux	5
4	nombre de groupe r	6
5	random walk	7
6	endomorphisme	8
7	parrallelisation	9
7.1	choix des points remarquables	9
8	bibliograpie	10

Chapitre 1

Rapel sur les courbes elliptique

Soit un un corp F_p (p premier) de caractéristique différente de 2 ou 3, soit $a, b \in F_p$. On peut avoir aussi la courbe définie sur F_q avec $q = p^n$, certaines propriétés sont un peu différentes. Une courbe elliptique est définie par le point à l'infini et l'ensemble des points (x, y) tel que soit satisfaite l'équation suivante :

$$y^2 = x^3 + ax + b.$$

Ces points forment un groupe abélien avec comme zéro : le point à l'infini et une loi de groupe. Soit $P, Q \in E$, $P = (x_1, y_1)$, $Q = (x_2, y_2)$ alors $P + Q = (x_3, y_3)$ tel que

$$x_3 = \mu^2 - x_1 - x_2, y_3 = \mu(x_1 - x_3)$$

$$\mu = \frac{y_2 - y_1}{x_2 - x_1} \text{ si } P \neq Q$$

$$\mu = \frac{3x_1^2 + a}{2y_1} \text{ si } P = Q$$

Une propriété importante sur les courbes elliptiques est le calcul de l'inverse d'un point pour la loi d'addition sur F_p . Si $P = (x_1, y_1)$ alors $-P = (x_1, -y_1)$. Pour une courbe elliptique sur F_q , on a une propriété identique. Si $P = (x_1, y_1)$ alors $-P = (x_1, -y_1)$.

Chapitre 2

Algorithme rho de Pollard

L'algorithme rho de pollard est basé sur le paradoxe des anniversaires. Dans un groupe fini si on prend des éléments de façon aléatoire, il faut en moyenne tirer \sqrt{n} éléments pour obtenir un élément que l'on a déjà tiré, ou n est le nombre d'élément de notre ensemble.

La formule générale pour obtenir une probabilité d'intersection ($I(n)$) est $1-p(n)$ où $p(n)$ est la probabilité d'avoir aucune intersection sur n tirage avec remise dans un groupe de m elements

$$p(n) = \frac{m!}{(m-n)!} * \frac{1}{m^n} \quad I(n) = 1 - \frac{m!}{(m-n)!} * \frac{1}{m^n}$$

Le nom de cette algorithme est dû à la forme que prend la suite aléatoire, on a d'abord un pré-cycle puis le cycle.

2.1 dans un groupe du type \mathbb{F}_p

Dans un groupe du type \mathbb{F}_p , Pollard a choisit comme fonction de tirage aléatoire un fonction qui ne tient compte que de l'élément précédent.

$$f(x) = x + P \text{ si } x \in [0; p/3[$$

$$f(x) = x + x \text{ si } x \in [p/3; 2p/3[$$

$$f(x) = x + Q \text{ si } x \in [2p/3; p[$$

Ici P est un generateur du groupe et Q est l'element dont on veut connaitre le logarithme en base P.

Algorithm 1 rho pollard

ENTRÉES: n,P,Q**SORTIES:** x tel que $Q = P^x \bmod n$

2.2 sur les courbe elliptique

L'algorithme rho de pollard est identique sur les courbes elliptiques sauf que l'on se place dans un cas plus générale avec en utilisant l'addition et des scalaires.

// ecriture de l'algo sur les courbe elliptique ?

2.3 algorithme de floyd

Il existe différent moyen pour trouver une collision dans un cycle, le meilleur algorithme pour en trouver une est l'agorithme de floyd qui permet de calculer de façon très rapide la longueur d'un cycle en se basant sur des collisions. La propriété qu'il a énoncé est que dans un groupe cyclyque si on a une marche aléatoire cyclique, alors au lieu de stocké chaque éléments et de vérifier si il n'est pas dans la liste. il calcule l'élément i et $2i$ de cette marche aléatoire jusqu'à avoir une collision.

exemple sur le groupe $\mathbb{Z}/17\mathbb{Z}$, avec la mache aléatoire $f(x) = x^2 + 1 \bmod 17$.
si $x = 1 \rightarrow 2 \rightarrow 5 \rightarrow 9 \rightarrow 13 \rightarrow 16 \rightarrow 0 \rightarrow 1$ donc $a_1 = 1, a_2 = 2, a_3 = 5 \dots$
// mauvais choix puissance de 2

Chapitre 3

Negation map

3.1 methode general

Sur une courbe elliptique sur \mathbb{F}_p l'inverse de $P = (x,y,z)$, simplifié à (x,y) est $-P = (x,-y)$. le but de cette méthode est de reduire le groupe de moitier, donc de faire une recherche de log discret dans $\langle P \rangle / \langle H \rangle$. On obtient donc une relation du type $\pm[a]P \pm [b]Q = \pm[a']P \pm [b']Q$. Il nous suffit donc de retrouver exactement la relation avec les bon signe pour retrouver le logarithme discret.

Premier problème on retombe très rapidement sur le meme point avec les meme scalaires, si on évite ses point et que l'on continue la boucle, elle continue indéfiniment. En plaçant des prints sur l'égalité et un compteur générale de boucle, on remarque que comme décrit dans l'article on a l'apparition de cycle court de longueur 2.

La méthode décrite dans l'article ...nous dis de garder en mémoire quelque points de la marche aléatoire, puis de faire un comparaison pour detecter si on entre dans un cycle longueur 2, cette comparaison s'effectue entre le terme que l'on vient de calculer et celui qui était deux crant avant. S'ils sont différents alors on continue la marche comme elle était, mais s'ils sont identiques alors le point suivant est $2 * \min(W_{-1}, W_{-2})$ où le min est le minimum lexicographique et la multiplication par 2 est l'opération de doublement sur la courbe elliptique.

Après enlèvement de ces cycle infructueux de taille 2, on retombe dans des cycles de taille 4.

3.2 cycle infructueux

appartition très fréquente de cycle de taille 2 n'important pas d'information sur le log discret on les elimine donc en

Chapitre 4

nombre de groupe r

Dans l'article de *Bernstein*, il est fait référence au fait que *Pollard* n'utilisait que trois groupes pour faire sa marche aléatoire, alors que Teske dans son papier *On random walks for pollard's rho method* nous dit qu'il est beaucoup plus efficace d'utiliser plus de groupe pour faire une marche aléatoire.

Dans son article *Teske* introduit un indice L :

$$L = \frac{\text{nombre d'iteration avant de trouver une collision}}{\sqrt{|G|}}$$

j'ai donc implémenté une fonction qui au lieu d'utiliser une marche aléatoire sur 3 groupe en faisant soit $+P$, $+Q$, ou un doublement, utilise r groupe où on ajoute à notre élément un élément spécifique du groupe. Ces différents éléments sont au préalable tirés aléatoirement. Je fais donc une marche additive, pour chaque W_i je lui ajoute un R_j où j est un hash de i. On a donc la formule suivante : $W_{i+1} = W_i + R_{h(i)}$.

Pour connaître à quel groupe appartient l'élément, je regarde sa première composante modulo r.

Teske après expérimentation trouve que pour la valeur $r \geq 20$ ont atteint un bien meilleur ratio pour l'indice L.

En refaisant les tests j'obtiens une valeur identique, au-delà de 20 on fait beaucoup de précalcul pour peu d'amélioration.

Chapitre 5

random walk

—attention choix stupide ne pas laisser —

Mon premier choix de marche aléatoire était de prendre les scalaires a et b puis de les regarder modulo 3 pour les séparer dans les trois groupes différents et donc faire l'opération que d'ajout de P ou de Q , ou le doublement. La première remarque avec les différents tests que je faisais était que je mettais plus de temps à trouver le logarithme discret qu'une recherche exhaustive. J'ai donc remarqué que la plupart du temps la fonction qui prenait un point de la courbe comme argument, ainsi que les scalaires a et b , ne retournait pas à chaque fois la même image pour un point donné. Concrètement si $W1 = [2]P \oplus [3]Q$ et $W2 = [4]P \oplus [5]Q$ avec $W1 = W2$ alors il n'avait pas le même résultat par f .

point important : plus trop une marche aléatoire pour 3 groupes donc plus pour r groupes $r > 19$.

Chapitre 6

endomorphisme

????

Chapitre 7

parrallelisation

pour parrallelisé le log discret on definie une règle pour avoir des point remarquable (tel que) a chaque fois que l'on a après une marche un point remarquable on le transmet a un serveur central ainsi que les donné de départ quand on a deux intersection de point remarquable on a une collision.

7.1 choix des points remarquables

Mon choix est de prendre des point dont la première variable (x) possède un certain nombre de bit de poids faible à zeros. J'ai fait différents tests : prendre $\log(p)/\log(2)/3$ ou $\log(p)/\log(2)/4$ appartition de pallier entre le nombre de point remarquable et le cardinal du groupe. groupe de 3 avec même poucentage d'elements remarquables dans le groupe .

```
? for( p = 15,30, n = prem_n_bit(p); E = courbe_ell_Fp(n); P = random(E); w = proba_remar
0.0340
0.0312
0.0307
0.0155
0.0155
0.0157
0.0079
0.0078
0.0078
0.0039
0.0039
0.0039
0.0020
0.0020
0.0020
```

calcule à faire en se basant sur le paradoxe des anniversaire pour tomber sur un point remarquable et pour avoir deux point remarquables identique.
detail de l'algorithme.

Chapitre 8

bibliograpie

Edlyn Teske, on a random walks for pollard's rho method.

Daniel J. Bernstein, Tanja Lange, Peter Schwabe, On the correct use of negationmap in the pollard's rho method.

John M.Pollard, monte carlo method for index computation mod p.