Équations du second degré à deux variables

Solving Bivariate Quadratic Congruences in Random Polynomial Time, L. M. Adleman, D. E. Estes, K. S. McCurley

Damien Stehlé

Mardi 19 Mars 2002

- Introduction
- Généralités sur ces algorithmes
- Comment se ramener à $x^2 - ky^2 \equiv m[n]$
- Nombres premiers dans une suite arithmétique
- Résoudre $x^2 - ky^2 \equiv m[n]$
- Conclusion

Introduction:

But: Résoudre en temps polynômial probabiliste l'équation:

$$Ax^2 + Bxy + Cy^2 + Dx + Ey + F \equiv 0[n].$$

- Rabin (1979): C'est impossible dans le cas général! Résoudre $x^2 \equiv 0[n]$ est aussi dur que factoriser n.

- Pollard, Schnorr (1985): Résolution de $x^2 - ky^2 \equiv m[n]$ quand $(km, n) = 1$, sous l'hypothèse de Riemann généralisée.

- Adleman, Estes, McCurley (1986): Résolution d'une plus grande classe d'équations, sans utilisation de l'HRG

1. Généralités sur ces algorithmes:

Au cours des algorithmes qui suivent, on peut sortir avec un diviseur strict de n. Que faire dans ce cas?

$\rightarrow$ Fait:
Si $n = n_1 n_2$ avec $n_1, n_2 < n$ et $(n_1, n_2) > 1$, alors en temps $O(M(n).\log n)$, on peut soit:

a. Construire a, b avec $ab = n$, $a, b < n$ et $(a, b) = 1$

b. S'assurer que $n_0 | n_1$ et $n_0 | n_2$ (avec $n_0 = \prod_{p|n} p$)

1. $g := (n_1, n_2)$, $k = g^2$, $h = n/g^2$

2. si $h = 1$, sortir car b. est vérifiée.

3. $m = (g, h)$. Si $m = 1$, sortir avec h, k.
   Sinon, $h = h/m$, $k = km$ et goto 2.

$\rightarrow$ Théorème Chinois:

Si $n = n_1 n_2$ avec $n_1, n_2 < n$ et $(n_1, n_2) = 1$, en $O(M(n) \log n)$, solutions modulo n_1 et modulo $n_2 \rightarrow$ solution modulo n.

$\rightarrow$ Lemme de Hensel:

En temps $O(M(n))$,
Solution modulo $d^i \rightarrow$ solution modulo d^{i+1}

$Ax_i^2 + Bx_i y_i + Cy_i^2 + Dx_i + Ey_i + F \equiv 0[d^i]$

$x_{i+1} = x_i + d^i X, y_{i+1} = y_i + d^i Y$

Cela revient à résoudre: $\alpha X + \beta Y \equiv \gamma[d]$

2. Comment se ramener à $x^2 - ky^2 \equiv m[n]$:

Théorème:

Soit $f(x,y) = Ax^2 + Bxy + Cy^2 + Dx + Ey + F$
Soit $\delta(f) = det \begin{bmatrix} 2A & B & D \\ B & 2C & E \\ D & E & 2F \end{bmatrix}$
Si $(\delta(f), n) = 1)$, alors en temps $O(M(n)\log^2 n)$ on peut soit:

a. Trouver $n = n_1 n_2$ avec $n_1, n_2 < n$, et $(n_1, n_2) = 1$

b. Trouver une solution de $f(x,y) \equiv 0[n]$

c. Se ramener à $x^2 - ky^2 \equiv m[n]$

La condition $(\delta(f), n) = 1)$ garantit que parmi A, B, D, une au moins est première avec n et les autres sont divisibles par n_0.

$\rightarrow$ Si $(A, n) = 1$, $u = 2x + A^{-1}By + A^{-1}D$, $v = y$:
on est ramené à $u^2 + Gv^2 + Hv + I \equiv 0[n]$ avec $(4GI - H^2, n) = 1$.

$\rightarrow$ Si $(C, n) = 1$, idem.

$\rightarrow$ Si $(B, n) = 1$, $x = u + v$, $y = v$:
on a alors $C' = A + B + C$ et $(C', n) = 1$.

$\rightarrow$ Sinon, $(D, n) = 1$ et n_0 divise A, B et C. Soit $d = (A, B, C)$. On calcule une solution $(x, 0)$ modulo d et on la "lifte" avec le lemme de Hensel.

On simplifie $u^2 + Gv^2 + Hv + I \equiv 0[n]$ de la même manière...

Remarques:

- La borne de complexité donnée est imprécise:
  en fait, c'est $O(M(n)\log^2 n)$ pour résoudre complètement et $O(M(n)\log n)$ pour sortir avec un diviseur strict ou être ramené à l'équation de c.

- Dans la condition sur le déterminant, on reconnaît une Hessienne. Faut-il en déduire quelque chose?

- Cette condition sur le déterminantest-elle "minimale" ?

- Si on utilise une arithmétique rapide on descend à $O(M(n)\log n \log\log n)$ pour b. et $O(M(n)\log\log n)$ pour a. et c.

3. Nombres premiers dans une suite arithmétique :

On distingue deux types d'entiers: les n communs, et les n exceptionnels, qui sont très rares. n est dit exceptionnel s'il existe un charactère exceptionnel χ modulo n.

Soit n impair et a avec $(a, n) = 1$.
Si n est commun, ou exceptionnel pour χ avec $\chi(a) = -1$, alors il existe D calculable telle que si $x \geq n^D$:

$$\sum_{p \leq x, p \equiv a[n], p \not\equiv 1[8]} 1 > \frac{x}{3\phi(n) \log x}$$

D'où le théorème central suivant:

Théorème:

Soit n impair, m tel que $(m, n) = 1$ et $\epsilon > 0$.
Alors il existe D tel que en temps $O(Pr(n) \log^2 n)$, on peut construire avec probabilité supérieure à $1 - \epsilon$ soit:

a. p_1 premier, $p_1 \not\equiv 1[8]$, $p_1 \equiv m[n]$ et $p_1 < n^D$

b. L avec $(L, n) = 1$ et p_2, p_3 tels que:
$p_2 \not\equiv 1[8]$, $p_2 \equiv L[n]$ et $p_2 < n^D$ $p_3 \not\equiv 1[8]$, $p_3 \equiv Lm[n]$ et $p_3 < n^D$

c. un diviseur strict de n

$Pr(n) = O(M(n) \log n)$ est le coût d'un test de primalité pour des entrées de taille $\log n$.

1. Choisir $x < n^D$ au hasard avec $x \equiv m[n]$, $x \not\equiv 1[8]$ et x impair

2. Tester la primalité de x (avec Solovay-Strassen par exemple). Si x est "premier", on a réalisé a.

3. Choisir $L < n$ au hasard. Si $(L, n) > 1$, on a réalisé c.

4. Choisir $y < n^D$ et $z < n^D$ au hasard de façon indépendante avec $y \equiv L[n]$, $y \not\equiv 1[8]$, y impair, $z \equiv Lm[n]$, $z \not\equiv 1[8]$ et z impair

5. Tester la primalité de y et z. S'ils sont premiers, alors on a réalisé c. Sinon, retourner en 1.

Si n n'est pas exceptionnel, ou exceptionnel pour χ avec l'égalité $\chi(a) = -1$, alors:

Pr(succès en 2.) $\geq \frac{3n^{D-1}/4}{9D \log n} > \frac{4}{9D \log n}$.
$\rightarrow O(\log n)$ tentatives.

Sinon, pour la moitié des L choisis, $\chi(L) = -1$. Dans ce cas, $\chi(mL) = -1$, et l'argument précédent marche encore ici:
$\rightarrow O(\log^2 n)$ tentatives

Remarques:

- On peut faire la même chose en remplaçant les $p_i \not\equiv 1[8]$ par des $p_i \equiv 1[8]$.

- Comme $Pr(n) = O(M(n)\log n)$, ces algorithmes fonctionnent en temps $O(M(n)\log^3 n)$. Peut-on faire mieux?
  Par exemple en profitant de la rareté des n exceptionnels, qui rajouttent un $\log n$ supplémentaire.

- Comment calcule-t-on D?

4. Résoudre $x^2 - ky^2 \equiv m[n]$:

Commençons par quelques cas particuliers...

$\rightarrow$ Si $k = 1$, on prend $x = \frac{m+1}{2}$ et $y = \frac{m-1}{2}$.

$\rightarrow$ Si $k = -1$, voici une solution en temps $O(M(n)\log^3 n)$:

1. Appliquer l'algorithme "$\equiv 1[8]$". Cela donne p_1, ou p_2 et p_3. Supposons pour simplifier que l'on a p_1.

2. Résoudre $w^2 \equiv -1[p_1]$ (avec Tonelli-Shanks par exemple)

3. On cherche le vecteur le plus court du réseau $[(1, w)\ ,\ (0, p)]$. On utilise pour cela l'algorithme de Gauss pour $||.||_\infty$.
Ça nous donne $aw + b \equiv 0[p]$ avec $||(a, b)||_\infty < \sqrt{p}$ (d'après le théorème de Minkowsky)... et donc $a^2 + b^2 = p$.
Comme $p \equiv m[n]$, c'est fini.

... et deux théorèmes de réduction...

$\rightarrow$ Si $(kLm, n) = 1$, alors à partir de solutions de deux des trois équations suivantes, on peut résoudre la troisième en temps $O(M(n)\log n)$.

$$x^2 - ky^2 \equiv m[n],\ x^2 - ky^2 \equiv L[n],\ x^2 - ky^2 \equiv Lm[n]$$

$(x^2 - ky^2)(z^2 - kw^2) = u^2 - kv^2$ avec $u = xz + kyw$ et $v = xw + yz$. Il suffit de faire une inversion modulo n...

$\rightarrow$ Si $(mk_1k_2, n) = 1$, $x_1^2 + k_1y_1^2 \equiv m[n]$, $x_2^2 + k_2y_2^2 \equiv m[n]$, alors en temps $O(M(n)\log n)$ on peut trouver une solution à $x^2 + k_1k_2y^2 \equiv m[n]$, ou un diviseur strict de n.

...et encore un cas particulier.

$\rightarrow$ Si $|k| = 2$, on construit aussi p_1, ou p_2 et p_3 par l'algorithme "$\equiv 1[8]$". Supposons que l'on ait obtenu p_1, pour simplifier.

$(\frac{2}{p_1}) = (\frac{-2}{p_1}) = 1$: Soit w une racine de k modulo p_1.

$q_1 := \frac{w^2-k}{p_1}$
Tant que $|q_i| \geq 2$, faire:
$x_i :=$ reste centré de x_{i-1} par q_i
$q_{i+1} = \frac{x_i^2-k}{q_i}$

On a $|q_{i+1}| \leq \frac{5}{6}|q_i|$, d'où un nombre de boucles en $O(\log n)$.
De plus:
$(w^2 - k)...(x_j^2 - k) = p_1 q_1^2 ... q_j^2 q_{j+1}$ avec $|q_{j+1}| = 1$

Cela donne $a^2 - kb^2 = p_1 c^2 q_{j+1}$, c'est quasiment fini!
C'est la même chose si l'on ait tombé sur p_2 et p_3.

Cet algorithme fonctionne en temps $O(M(n) \log^3 n)$

et enfin la résolution de l'équation.

Input: k, m, n avec $(km, n) = 1$
Output: Une solution à $x^2 - ky^2 \equiv m[n]$

1. Si $|k| \leq 2$, utiliser ce qui précède

2. Utiliser l'algorithme "$\not\equiv 1[8]$" pour m.
Supposons que l'on a trouvé p_1.

3. Si $(\frac{k}{p_1}) = 1$, utiliser le même algorithme que pour $|k| = 2$, sauf que l'on s'arête à $|q_j| \leq \sqrt{(\frac{13|k|}{9})}$, pour obtenir $u^2 - kv^2 = p_1 k_{j+1} w^2$

4. À ce niveau, soit on a un diviseur strict de n, soit on multiplie le tout par w^{-2}.
On appelle l'algorithme avec k_1, k et n (cf les résultats de réduction).

5. Si $(\frac{-k}{p_1}) = 1$ ou $(\frac{2k}{p_1}) = 1$, idem.

6. Si on était tombé sur p_1 et p_2, on aurait fait la même chose, mais on aurait eu deux appels récursifs.

Finalement: $C_{n,k} \leq [2C_{n,O(\sqrt{k})}$ ou $2C_{O(\sqrt{n}),k}] + O(M(n)\log^3 n)$

ET $C_{n,k} = O(M(n)\log^4 n)$.

Conclusion

- Ce qui coûte cher, ce sont les recherches de nombres premiers dans les suites arithmétiques. Peut-on les rendre moins coûteuses? Sont-elles indispensables?

- Peut-on à partir de là obtenir une implémentation efficace?

- Peut-on enlever Monte-Carlo?

- Peut-on enlever Las Vegas, quitte à réutiliser l'hypothèse de Riemann généralisée?

- Que peut-on en déduire pour les cas "limites", en particulier la factorisation?