UNIVERSITÁ DEGLI STUDI "ROMA TRE"
FACOLTÁ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
DIPARTIMENTO DI MATEMATICA

Fattorizzazione e Test di Primalità con Curve Ellittiche

(SINTESI)

TESI DI LAUREA SPECIALISTICA IN MATEMATICA

Roma, Febbraio 2011

A.A. 2009/2010

Candidato: Pulvano Gabriele

Relatore: prof. Francesco Pappalardi

Le curve ellittiche rappresentano eleganti oggetti matematici, iniziati a studiare intorno alla metà del diciannovesimo secolo, le cui applicazioni riguardano una gran molteplicità di campi come l'algebra, la geometria, la teoria dei numeri e recentemente anche l'informatica (ricordiamo il loro utilizzo nella dimostrazione dell'Ultimo Teorema di Fermat e nei recenti sistemi crittografici). Questa tesi si concentra sulle applicazioni che esse hanno avuto nei problemi della primalità e della fattorizzazione dei numeri interi. Tali applicazioni si concretizzano rispettivamente nei lavori di Goldwasser e Kilian (1986) e di Hendrik Lenstra (1984).
I loro algoritmi, ossia rispettivamente l'*Elliptic Curve Primality Proving* (ECPP) e l'*Elliptic Curve Method* (ECM), sono gli analoghi con le curve ellittiche di alcuni metodi classici. Questi ultimi vengono anche detti *metodi moltiplicativi* perché si basano sul fatto che l'ordine del gruppo moltiplicativo $\mathbb{F}_p^*$ (con p primo) è uguale a $p-1$. Di importanza analoga per i metodi basati sulle curve ellittiche è l'insieme dei punti di una curva a coordinate in $\mathbb{F}_p$, che forma un gruppo rispetto ad una certa operazione di somma di punti che sarà definita. Per il Teorema di Hasse (1934) tale gruppo ha ordine $p+1-a$, dove a è un intero che dipende dalla curva E e da p, e tale che $|a| \leq 2\sqrt{p}$. Il vantaggio risiede nel fatto che, fissato p, il valore di a e quindi dell'ordine del gruppo varia al variare di E. Se ad esempio un metodo fallisce si può tentare nuovamente cambiando curva, opzione non disponibile nei metodi classici.

Nel Capitolo 1 vengono presentati i concetti base della teoria fondamentale delle curve ellittiche.

Nel Capitolo 2 si descrive il classico Metodo di fattorizzazione $p-1$ di Pollard e si analizza l'ECM di Lenstra, che è la sua generalizzazione con l'uso delle curve ellittiche. Una sezione è anche dedicata alla descrizione dei *numeri lisci*, che rappresentano un ingrediente fondamentale per lo studio dei due algoritmi.

Nel Capitolo 3 vengono descritte alcune nozioni di *pseudoprimalità* e vengono proposti due esempi di test di primalità probabilistici basati su di esse. Inoltre viene analizzato in dettaglio l'ECPP di Goldwasser-Kilian nella sua versione standard originaria, accennando prima al suo analogo moltiplicativo, il Test di Pocklington-Lehmer.

1 Generalità sulle Curve Ellittiche

Sia $\mathcal{K}$ un campo. Ad esempio, $\mathcal{K}$ può essere $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$, un campo finito $\mathbb{F}_p$ con un numero p (primo) di elementi, o in generale una sua estensione $\mathbb{F}_q$ dove $q = p^r$, p primo e $r \geq 1$.
Con *equazione di Weierstrass (generalizzata)* intendiamo un'equazione del

tipo:

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6.$$

Con *curva ellittica* intendiamo il grafico di un'equazione di Weierstrass senza punti singolari, e diciamo che la curva è definita sul campo $\mathcal{K}$ se $a_1, a_2, a_3, a_4, a_6 \in \mathcal{K}$. Si può dimostrare che se $\mathrm{Char}\,\mathcal{K} \neq 2, 3$ è possibile, tramite affinità, trasformare equazioni di Weierstrass generalizzate in equazioni di Weierstrass *ridotte* della forma:

$$y^2 = x^3 + Ax + B$$

ovvero che in caratteristica $\neq 2, 3$ non è restrittivo considerare equazioni di questo tipo, perché esse descrivono tutte le equazioni di Weierstrass generalizzate. Dato che lavoreremo sempre in campi di caratteristica $\neq 2, 3$, assumeremo sempre come ridotta l'equazione di una curva ellittica.
Un'equazione di Weierstrass si dice *singolare* se ha radici multiple, ovvero se si annulla il suo discriminante. In base all'espressione ricavata di tale discriminante, si dimostra che la condizione di non-singolarità (per curve ellittiche in caratteristica $\neq 2, 3$) è:

$$4A^3 + 27B^2 \neq 0.$$

Definiamo l'insieme dei punti a coordinate in $\mathcal{K}$ di una curva ellittica E come:

$$E(\mathcal{K}) := \{(x, y) \in \mathcal{K} \times \mathcal{K} \ : \ y^2 = x^3 + Ax + B\} \ \cup \ \{\infty\}.$$

Il motivo dell'aggiunta di un *punto all'infinito* $\{\infty\}$ nell'insieme dei punti di E è compreso meglio se si considera l'equazione proiettiva piuttosto che quella affine della curva. Sostanzialmente, lo scopo del punto all'infinito è quello di essere il punto di intersezione della curva ellittica con ogni retta verticale del piano. Il piano affine $\mathbb{A}^2(\mathcal{K})$ rappresenta l'insieme dei punti finiti del piano proiettivo $\mathbb{P}^2(\mathcal{K})$, grazie all'immersione:

$$\mathbb{A}^2(\mathcal{K}) \hookrightarrow \mathbb{P}^2(\mathcal{K})$$

data da: $(x, y) \mapsto (x : y : 1)$. Si può scrivere:

$$\mathbb{P}^2(\mathcal{K}) = \mathbb{A}^2(\mathcal{K}) \sqcup \mathbb{P}^1(\mathcal{K})$$

dove $\mathbb{P}^1(\mathcal{K})$ è l'insieme dei punti all'infinito della forma $(x : y : 0)$ del piano proiettivo. In coordinate proiettive si ha che:

$$E(\mathcal{K}) := \{(x : y : z) \in \mathbb{P}^2(\mathcal{K}) \ : \ y^2z = x^3 + Axz^2 + Bz^3\}.$$

Ponendo $z = 0$ si trova che l'unico punto all'infinito di una curva ellittica è il punto $(0 : 1 : 0)$, che è anche il punto di intersezione di due rette verticali del piano di forma omogenea $x = c_1 z$ e $x = c_2 z$. Intuitivamente si può pensare al punto all'infinito come all'estremo superiore e nello stesso tempo inferiore dell'asse y. Esistono molti vantaggi nel trattare una curva ellittica in coordinate proiettive. Tuttavia, in questa tesi, utilizzeremo quasi sempre coordinate affini.

Definiamo un'operazione "somma": $E(\mathcal{K}) \times E(\mathcal{K}) \to E(\mathcal{K})$ che ci consenta di sommare due punti $P = (x_1, y_1)$ e $Q = (x_2, y_2)$ su una curva ellittica E allo scopo di ottenere un terzo punto $P + Q$ su di essa. Tale operazione è definita nel modo seguente (in caratteristica $\neq 2, 3$):

1. se $P \neq Q$ e $x_1 \neq x_2$ allora $P + Q = (x_3, y_3)$, dove:

$$x_3 = \Big(\frac{y_2 - y_1}{x_2 - x_1}\Big)^2 - x_1 - x_2, \qquad y_3 = \frac{y_2 - y_1}{x_2 - x_1}\,(x_1 - x_3) - y_1;$$

2. se $P \neq Q$ e $x_1 = x_2$ allora $P + Q = \infty$;

3. se $P = Q$ e $y_1 \neq 0$ allora $P + Q = 2P = (x_3, y_3)$, dove:

$$x_3 = \Big(\frac{3x_1^2 + A}{2y_1}\Big)^2 - 2x_1, \qquad y_3 = \frac{3x_1^2 + A}{2y_1}\,(x_1 - x_3) - y_1;$$

4. se $P = Q$ e $y_1 = 0$ allora $P + Q = 2P = \infty$;

5. $P + \infty = \infty + P = P$.

Si dimostra che l'insieme $E(\mathcal{K})$ forma un gruppo abeliano rispetto a questa operazione, con elemento neutro ∞. Inoltre l'inverso del punto $P = (x, y)$ è $-P = (x, -y)$.
In altre parole abbiamo considerato la retta passante per P e Q (o la retta tangente alla curva in P, se $P = Q$). Essa interseca la curva in un solo altro punto R (i punti di intersezione tra una retta ed una curva ellittica sono al più 3). Nel caso in cui P e Q siano verticalmente allineati allora $R = \infty$. Il punto $P + Q$ non è altro che il simmetrico rispetto all'asse x di R, ovvero $-R$. Se $R = \infty$ allora anche $P + Q = \infty$ (Figura 1).

Nel corso della tesi servirà di moltiplicare un punto su una curva ellittica per un intero non nullo. Spieghiamo quindi cosa si intende per *multiplo di un punto*. Sia $P \in E(\mathcal{K})$ e sia $k \in \mathbb{Z} \setminus \{0\}$. Se $k > 0$ definiamo:

$$kP := \overbrace{P + P + \ldots + P}^{k \text{ volte}}.$$

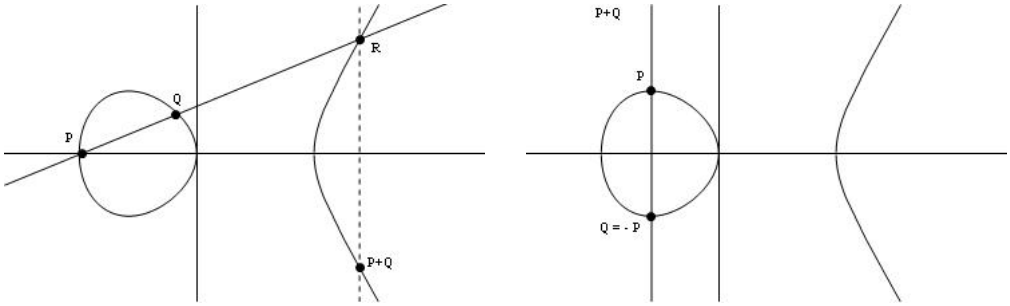

Figure 1: somma di due punti distinti su una curva ellittica.

Se invece $k < 0$:

$$kP := \underbrace{(-P) + (-P) + \ldots + (-P)}_{|k| \ volte}.$$

La definizione data potrebbe apparentemente creare una certa ambiguità, in quanto la moltiplicazione sulle curve ellittiche dipende dalla "catena di addizioni" scelta. Ad esempio, $5P$ può essere calcolato attraverso:

$$P \ \rightarrow \ 2P \ \rightarrow \ 4P = 2(2P) \ \rightarrow \ 5P = P + 4P$$

oppure:

$$P \ \rightarrow \ 2P \ \rightarrow \ 3P = P + 2P \ \rightarrow \ 5P = 3P + 2P.$$

Dall'associatività della somma segue che, qualunque sia la catena di addizioni scelta, il risultato non cambia.

Non studieremo curve ellittiche definite su $\mathbb{R}$ e su $\mathbb{C}$, e accenneremo soltanto alle curve ellittiche su $\mathbb{Q}$. Quello che ci interessa sono curve a coefficienti interi. Se riduciamo una di queste modulo un primo $p \neq 2, 3$ si ottiene una curva E_p con coefficienti in $\mathbb{F}_p$. Questa è una frequente situazione in cui una curva può risultare singolare, nel caso in cui $p \mid 4A^3 + 27B^2$. L'insieme dei primi per cui ciò accade è detto *insieme dei primi di cattiva riduzione* per E. Se invece ridurre una curva modulo un primo p non da luogo a singolarità allora si ha una *buona riduzione*.
Consideriamo curve non singolari in $\mathbb{F}_p$ o, in generale, in $\mathbb{F}_q$ dove $q = p^r$. L'operazione di somma nel gruppo $E(\mathbb{F}_q)$ è definita nello stesso modo, ed i calcoli si effettuano normalmente ma riducendo modulo p (non ci sono problemi per le inversioni dato che siamo in un campo). Per calcolare l'ordine di $E(\mathbb{F}_q)$ utilizzeremo implicitamente l'algoritmo di Schoof, di complessità polinomiale $O(\log^8 q)$. Il risultato più importante per le curve ellittiche sui campi finiti è il seguente teorema.

Teorema 1 (Hasse) *Sia E una curva ellittica definita su $\mathbb{F}_q$. Allora l'ordine di $E(\mathbb{F}_q)$ soddisfa:*

$$q + 1 - 2\sqrt{q} \leq \#E(\mathbb{F}_q) \leq q + 1 + 2\sqrt{q}.$$

□

L'intervallo $I_q := [q + 1 - 2\sqrt{q}, q + 1 + 2\sqrt{q}] = [(\sqrt{q} - 1)^2, (\sqrt{q} + 1)^2]$ è detto *intervallo di Hasse*, ed ha ampiezza $4\sqrt{q}$.

Consideriamo ora l'anello $\mathbb{Z}/n\mathbb{Z}$, dove n è un numero composto. Lavorare con una curva ellittica definita su $\mathbb{Z}/n\mathbb{Z}$ significa ridurre mod n l'equazione della curva ed effettuare ogni operazione mod n, allo stesso modo che nel campo $\mathbb{F}_p$. Il problema è che $\mathbb{Z}/n\mathbb{Z}$ non è un campo, ma solamente un anello. Ciò vuol dire che potrebbero presentarsi problemi quando si tenta di invertire i denominatori nelle formule di somma. Esiste una teoria specifica per le curve ellittiche su particolari anelli che non discuteremo, se non in linea generale. Quando dovremo fattorizzare un intero composto n, lo scopo sarà proprio quello di trovare un denominatore d non invertibile mod n. L'algoritmo di Lenstra si basa su questo principio. Il Massimo Comun Divisore (d, n) sarà proprio un fattore non banale di n.
Ci servirà il seguente risultato. Se $n = pq$ è il prodotto di soli due primi dispari distinti, allora si ha:

$$E(\mathbb{Z}/n\mathbb{Z}) \simeq E(\mathbb{F}_p) \oplus E(\mathbb{F}_q).$$

Di conseguenza $\#E(\mathbb{Z}/n\mathbb{Z}) = \#E(\mathbb{F}_p) \cdot \#E(\mathbb{F}_q)$.

2 Metodi di Fattorizzazione

2.1 Numeri Lisci

Un intero si dice *liscio* se ha solo fattori primi "piccoli". Vedremo che tale condizione è essenziale per il successo degli algoritmi che vogliamo esaminare. In particolare, sia $\mathcal{B} \in \mathbb{N}$. Un numero intero n si dice $\mathcal{B}$-*liscio* se i suoi fattori primi sono $\leq \mathcal{B}$.
Definiamo il numero di interi $\mathcal{B}$-lisci nell'intervallo $[1, x]$ come:

$$\psi(x, \mathcal{B}) := \#\{n \in \mathbb{N},\ n \leq x\ :\ p \mid n \Rightarrow p \leq \mathcal{B}\}.$$

La probabilità di scegliere casualmente un intero $\mathcal{B}$-liscio nell'intervallo $[1, x]$ è data dal valore di:

$$\frac{\psi(x, \mathcal{B})}{x}.$$

Definiamo:

$$L(x) := \exp(\sqrt{\log x \log\log x}).$$

Siamo interessati al caso $\mathcal{B} = L(x)^\alpha$, per qualche $\alpha > 0$. Un Teorema di Canfield-Erdős-Pomerance ci dice che, per $x \to \infty$:

$$\frac{\psi(x, L(x)^\alpha)}{x} = L(x)^{-1/(2\alpha)+o(1)}.$$

2.2 Metodo $p-1$ di Pollard

Il Metodo $p-1$ di Pollard è un algoritmo probabilistico di fattorizzazione pubblicato da J.M. Pollard nel 1974.
Sia fissato un intero $n \geq 2$ che vogliamo fattorizzare. Supponiamo per semplicità $n = pq$. Dal Teorema Cinese dei Resti:

$$U(\mathbb{Z}/n\mathbb{Z}) \simeq \mathbb{F}_p^* \times \mathbb{F}_q^*.$$

L'idea alla base di questo algoritmo è che, dato $a \in U(\mathbb{Z}/n\mathbb{Z})$, esista k (ad esempio $k = p-1$, dal Piccolo Teorema di Fermat) tale che $a^k \equiv 1 \pmod{p}$, per cui $(a^k - 1, n) \geq p$. Se trovassimo casualmente un a, con $2 \leq a \leq n-1$, tale che $(a, n) \neq 1$, allora avremmo comunque fattorizzato n, ma la probabilità che questo accada è molto bassa, specialmente se n è composto solo da due fattori primi dispari p e q.
La condizione fondamentale richiesta dall'algoritmo è che $p-1$ sia $\mathcal{B}$-liscio, per un certo limite $\mathcal{B}$. Infatti, se questa condizione è soddisfatta, l'intero k sarà relativamente piccolo, facilitando l'operazione di esponenziazione modulare $a^k \bmod p$. Una possibile scelta dell'esponente è quella di porre $k = \mathcal{B}!$. In questo modo k ha ottime possibilità di essere un multiplo di $p-1$ in quanto è multiplo di tutti gli interi $\leq \mathcal{B}$. Risulta comunque più conveniente scegliere k come il Minimo Comune Multiplo tra tutti gli interi $\leq \mathcal{B}$, che chiamiamo $M(\mathcal{B}) := \mathrm{mcm}(2, \ldots, \mathcal{B})$. Esso è dato dalla seguente espressione:

$$M(\mathcal{B}) = \prod_{p \leq \mathcal{B}} p^{[\log \mathcal{B} / \log p]}$$

E' necessario che solamente $p-1$, e non anche $q-1$, sia $\mathcal{B}$-liscio, altrimenti si avrebbe $(a^k - 1, n) = n$. Il problema principale è la scelta di $\mathcal{B}$. Se esso è troppo piccolo la probabilità che $p-1$ (oppure $q-1$) sia $\mathcal{B}$-liscio potrebbe essere troppo bassa; d'altra parte se esso è troppo grande il calcolo di a^k potrebbe risultare troppo lento.

Una schematizzazione del Metodo $p-1$ di Pollard è la seguente:

Metodo $p-1$ di Pollard

INPUT: n, $\mathcal{B}$
OUTPUT: un fattore non banale di n;
0 (il metodo fallisce)

1. `generare casualmente un intero` $a \in \{2, \ldots, n-2\}$;
2. $d \leftarrow (a, n)$;
3. `se` $d \neq 1$, `RETURN` d;
4. $k \leftarrow M(\mathcal{B})$;
5. $x \leftarrow a^k \pmod{n}$;
6. $d \leftarrow (x-1, n)$;
7. `se` $d \neq 1$ & $d \neq n$, `RETURN` d;
8. `RETURN` **0**.

Se entrambi $p-1$ e $q-1$ hanno fattori primi molto grandi, il Metodo di Pollard è impraticabile. Si può procedere in questo caso effettuando un alto numero di scelte casuali di a, sperando di trovare quello giusto, ma risulta più conveniente incrementare $\mathcal{B}$, anche se questo accrescerebbe di molto la complessità computazionale (essa risulta polinomiale in $\mathcal{B}$ e quindi esponenziale nel numero di cifre di $\mathcal{B}$).

2.3 ECM di Lenstra

L'*Elliptic Curve Method*, ideato da H.W. Lenstra Jr. nel 1985, è un algoritmo di fattorizzazione probabilistico di complessità congetturata subesponenziale, ed è la generalizzazione con le curve ellittiche del Metodo $p-1$ di Pollard.
L'idea consiste nel sostituire il gruppo $\mathbb{F}_p^*$ con il gruppo dei punti di un curva ellittica $E(\mathbb{F}_p)$. Mentre il primo gruppo, fissato n (e quindi p), ha cardinalità univocamente determinata uguale a $p-1$, il secondo ha cardinalità variabile in un intervallo (di Hasse) di ampiezza $4\sqrt{p}$, al variare della curva scelta. Come nel Metodo di Pollard richiediamo una condizione essenziale per la riuscita dell'algoritmo sulla cardinalità del gruppo dei punti della curva ellittica, ovvero che $\#E(\mathbb{F}_p)$ sia $\mathcal{B}$-liscio, per qualche limite scelto $\mathcal{B}$. Contrariamente ad esso, se tale condizione non risulta soddisfatta è possibile cambiare un certo numero di volte l'equazione della curva prima di incrementare il valore di $\mathcal{B}$, e questo rappresenta il vantaggio nell'uso delle curve ellittiche nella fattorizzazione.

Una volta costruito un opportuno intero k e scelta casualmente una curva ellittica E ed un punto P su di essa, l'idea è semplicemente quella di moltiplicare il punto P per k. Similmente al metodo di Pollard, si spera che k sia un multiplo di $\#E(\mathbb{F}_p)$ ma non di $\#E(\mathbb{F}_q)$, in modo che:

$$kP = \infty \pmod{p} \qquad \text{e} \qquad kP \neq \infty \pmod{q}$$

(che è probabile se solamente $\#E(\mathbb{F}_p)$ è $\mathcal{B}$-liscio).
Come abbiamo detto, una moltiplicazione di un punto su una curva ellittica non è altro che una serie di addizioni di punti. Se ad un certo punto del processo si presenta il problema dell'impossibilità dell'inversione di un denominatore D, data dalla presenza di zero-divisori in $\mathbb{Z}/n\mathbb{Z}$, allora si è trovato un fattore non banale di n, dato da (D, n).
Una schematizzazione dell'ECM di Lenstra, che utilizza una sola curva ellittica, è la seguente:

ECM di Lenstra

INPUT: n, $\mathcal{B}$, v
OUTPUT: un fattore non banale di n;
0 (il metodo fallisce)

1. $flag \leftarrow 0$;
2. `mentre` $flag = 0$ `esegui:`
3. `generare casualmente` $a, x, y \in \{0, \ldots, n-1\}$;
4. $b \leftarrow y^2 - x^3 - ax \pmod{n}$;
5. $d \leftarrow (6(4a^3 + 27b^2), n)$;
6. `se` $d \neq 1$ & $d \neq n$, `RETURN` d;
7. `se` $d = 1$, $flag \leftarrow 1$;
8. $P \leftarrow (x, y)$;
9. $\mathcal{C} \leftarrow [v + 1 + 2\sqrt{v}]$;
10. `per` $i = 1, \ldots, \pi(\mathcal{B})$ `esegui:`
11. $r \leftarrow i$`-esimo primo` $\leq \mathcal{B}$;
12. $e_r \leftarrow [\log \mathcal{C} / \log r]$;
13. `per` $j = 1, \ldots, e_r$ `esegui:`
14. $P \leftarrow rP$ `(se il calcolo è possibile);`
`se il denominatore` D `del coefficiente angolare non è invertibile` mod n, `allora` $d \leftarrow (D, n)$;
15. `se` $d \neq n$ `RETURN` d,
`altrimenti RETURN` **0**;
16. `RETURN` **0**.

Per aumentare la probabilità di successo, vogliamo costruire un intero k che sia divisibile da potenze di primi r piccoli (minori di un certo $\mathcal{B}$), tutte minori di un secondo limite $\mathcal{C}$. Quindi definiamo:

$$k = \prod_{r \leq \mathcal{B}} r^{e_r}$$

dove $e_r = [\log \mathcal{C} / \log r]$ è il più grande esponente possibile tale che $r^{e_r} \leq \mathcal{C}$. Supponiamo che $n = pq$, ma che p e q non abbiano lo stesso ordine di grandezza, e definiamo p come il fattore più piccolo di n. Supponiamo inoltre che p sia minore di un certo limite v, e poniamo $\mathcal{C} := [v + 1 + 2\sqrt{v}]$. Se il calcolo di kP ha avuto successo l'algoritmo fallisce, e si procede con un possibile incremento di $\mathcal{B}$ e/o di $\mathcal{C}$ (cioè di v), oppure con il cambiamento dell'equazione della curva ellittica. In ogni caso non vogliamo che il numero di tentativi superi un certo limite prestabilito, denotato con h. Il limite $\mathcal{B}$ misura essenzialmente il tempo speso dall'algoritmo, ed è un fattore direttamente proporzionale alla probabilità di successo, come per il Metodo $p-1$ di Pollard. La seguente proposizione dimostra come, sotto certe ipotesi, l'algoritmo di Lenstra riesca a trovare fattori non banali di n.

Proposizione 1 *Siano: $n, \mathcal{B}, v$ interi > 1, $a, x, y \in \mathbb{Z}/n\mathbb{Z}$, $b \equiv y^2 - x^3 - ax \pmod{n}$, e P un punto sulla curva $E : y^2 = x^3 + ax + b$. Supponiamo che $n = pq$. L'algoritmo ha successo se i primi p e q soddisfano:*

1. *$p \leq v$;*
2. *$(6(4a^3 + 27b^2), n) = 1$;*
3. *$\#E(\mathbb{F}_p)$ è $\mathcal{B}$-liscio;*
4. *$\#E(\mathbb{F}_q)$ non è divisibile dal più grande primo che divide l'ordine di P mod p.* □

L'analisi della complessità dell'algoritmo di Lenstra si basa sulla distribuzione dei numeri lisci all'interno di intervalli del tipo:

$$(p + 1 - \sqrt{p}, p + 1 + \sqrt{p})$$

(si noti che tali intervalli hanno ampiezza minore dell'intervallo di Hasse). La scelta casuale di una curva ellittica su $\mathbb{Z}/n\mathbb{Z}$ equivale alla scelta casuale ed uniforme di un intero X da un tale intervallo. L'algoritmo ha buone probabilità di trovare un fattore p di n se X ha buone probabilità di dividere k. Sia:

$$u = \#\{s \in \mathbb{Z} \ : \ |s - (p+1)| < \sqrt{p}, \ s \text{ è } \mathcal{B}\text{-liscio}\}.$$

Denotiamo con $f(\mathcal{B}) := u/(2[\sqrt{p}]+1)$ la probabilità che un intero scelto casualmente nell'intervallo $(p+1-\sqrt{p}, p+1+\sqrt{p})$ sia $\mathcal{B}$-liscio. Lenstra dimostra che esiste una costante c, effettivamente calcolabile, tale che la probabilità di successo dell'algoritmo completo con inputs $n, \mathcal{B}, v, h$ sia almeno:

$$1 - e^{hcf(\mathcal{B})/(3\log v)}.$$

Poniamo $\mathcal{B} = L(p)^\alpha$, per qualche $\alpha > 0$, dove p è sempre il fattore più piccolo di n. Con semplici calcoli si trova che il valore ottimale di α è $\alpha = 1/\sqrt{2}$. La congettura che ci serve è che l'analogo del risultato di Canfield-Erdős-Pomerance valga anche per intervalli della forma $(p+1-\sqrt{p}, p+1+\sqrt{p})$. Questo porta alla seguente stima congetturata del tempo di esecuzione dell'algoritmo di Lenstra:

Congettura: *Siano n, g interi positivi, dove n non è la potenza di un primo ed è tale che $(n,6)=1$ (questa scelta consente di lavorare con curve ellittiche in forma ridotta). Sia p il più piccolo fattore primo di n. Allora esiste una funzione $K: \mathbb{R}^+ \to \mathbb{R}^+$ con:*

$$K(x) = e^{\sqrt{(2+o(1))\log x \log\log x}}, \qquad \text{per } x \to \infty$$

tale che l'algoritmo di Lenstra, applicato con opportuni valori di $\mathcal{B}, v, h$, trovi un fattore non banale di n, con probabilità almeno $1-e^{-g}$, in tempo:

$$g \cdot K(p) M(n),$$

dove $M(n)$ è il tempo speso per effettuare una singola somma sulla curva, e varia a seconda del metodo utilizzato. Possiamo supporre $M(n) = O(\log^2 n)$.
□

Contrariamente ai suoi antagonisti, l'ECM ha un tempo di esecuzione che dipende dalla dimensione dei fattori di n. In altre parole, più questi sono piccoli, più l'algoritmo è efficiente. Non è conveniente utilizzarlo quindi su moduli RSA. Il caso peggiore infatti si verifica se n è il prodotto di due primi dello stesso ordine di grandezza. In tal caso, dato che $p \approx \sqrt{n}$, la complessità diventa dell'ordine di $e^{(1+o(1))\sqrt{\log n \log\log n}}$. Per questo motivo l'ECM è più indicato per trovare fattori primi di "piccole" dimensioni.
Contrariamente ai test di primalità, ancora oggi non è conosciuto un algoritmo di fattorizzazione con costo computazionale polinomiale.

3 Test di Primalità

Quelli che esamineremo saranno test di primalità probabilistici. Si distinguono due categorie di test probabilistici: i test di tipo *Monte Carlo* e

quelli di tipo *Las Vegas*. Il primo tipo produce sempre una risposta: essa è certamente corretta nel caso in cui n sia composto, mentre potrebbe essere affetta da un minimo errore quando afferma che n sia primo. Il secondo tipo invece potrebbe non produrre alcuna risposta. Tuttavia, se una risposta ci fosse, questa sarebbe certamente corretta. Esempi di test di tipo Monte Carlo sono il Test di Solovay-Strassen e quello di Miller-Rabin, con probabilità di errore rispettivamente minore di 0.5 e 0.25 per iterazione.

3.1 Test di Pocklington-Lehmer

Il Test di Pocklington-Lehmer è un test di primalità probabilistico di tipo Las Vegas e basa sul seguente teorema:

Teorema 2 (Pocklington-Lehmer) *Sia $n \geq 3$ un intero. Sia $n-1 = FR$, con $(F, R) = 1$ ed $F \geq \sqrt{n}$. Supponiamo sia nota la fattorizzazione completa di F. Se per ogni primo $q \mid F$ esiste a_q tale che:*

$$a_q^{n-1} \equiv 1 \pmod{n} \qquad e \qquad (a_q^{(n-1)/q} - 1, n) = 1$$

allora n è primo. □

Per provare la primalità di un intero n utilizzando il Test di Pocklington-Lehmer bisogna innanzitutto fattorizzare parzialmente $n-1$ come $n-1 = FR$, con $(F, R) = 1$ e $F \geq \sqrt{n}$, e fattorizzare completamente F. Dopodiché, per ogni fattore primo q di F, occorre cercare un a_q che soddisfi le ipotesi del teorema. Si potrebbe aver bisogno di provare inoltre la primalità dei fattori q di F applicando ulteriormente il medesimo test su q, e cosi via finché non si ottengano numeri primi già noti o valori per cui la divisione per tentativi non abbia un costo troppo alto.
Se annotassimo tutti i valori di q e dei corrispondenti a_q otterremmo una prova della primalità di n, ovvero un certificato di primalità. Chiunque può utilizzare tali informazioni per verificare per proprio conto la correttezza della dimostrazione, senza bisogno di sapere come questi fattori siano stati trovati. Il limite principale del Test di Pocklington-Lehmer è che esso necessita di una fattorizzazione, anche se parziale.

3.2 Standard ECPP di Goldwasser-Kilian

Nel 1986 S. Goldwasser e J. Kilian proposero un test di primalità con tempo atteso di esecuzione polinomiale basato sul principio del teorema di Pocklington-Lehmer. Esso si basa sul seguente teorema:

Teorema 3 *Sia $n > 1$ un intero, con $(n, 6) = 1$, e sia E una curva ellittica su $\mathbb{Z}/n\mathbb{Z}$. Supponiamo che esista un intero m che abbia un fattore primo q tale che $q > (n^{1/4} + 1)^2$. Se esiste un punto $P \in E(\mathbb{Z}/n\mathbb{Z})$ tale che:*

1. *$mP = \infty$ in $E(\mathbb{Z}/n\mathbb{Z})$;*
2. *$\frac{m}{q}P \neq \infty$ in $E(\mathbb{F}_p)$, per ogni primo $p \mid n$,*

allora n è primo. □

Vogliamo provare la primalità di un intero n che già supponiamo primo con alta probabilità (applicando su di esso il Test di Miller-Rabin un certo numero di volte).
Dopo aver scelto casualmente una curva ellittica E su $\mathbb{Z}/n\mathbb{Z}$ si calcola m, l'ordine del gruppo $E(\mathbb{Z}/n\mathbb{Z})$, e si controlla che questo sia il doppio[1] di un fattore $q > (n^{1/4} + 1)^2$, anch'esso probabilmente primo (il controllo viene effettuato ancora col Miller-Rabin). L'ordine del gruppo dei punti viene calcolato con l'algoritmo di Schoof che terminerebbe in tempo polinomiale se n fosse davvero primo. Poi si sceglie un punto P su E di ordine q. Per il teorema precedente n sarebbe primo, a condizione che lo sia anche q. Quindi si itera il test inserendo q come nuovo input, riducendo il problema della primalità di n a quello della primalità di q.
Iteriamo il test su valori sempre più bassi di q fino a quando non è possibile provarne la primalità tramite un test deterministico. Useremo a tale scopo il test deterministico "quasi-polinomiale" (in realtà subesponenziale) di Cohen-Lenstra. Di seguito denotiamo con k la lunghezza dell'input iniziale n e con C una costante positiva tale che il test di Cohen-Lenstra termini in tempo $O(k)$ su input di lunghezza

$$2^{k^{C/\log\log k}}.$$

Si dimostra che una tale costante C esiste sempre. Il processo iterativo dell'algoritmo si arresta non appena il corrente numero primo diventi minore del limite dato dalla precedente espressione. Usando l'ultimo primo trovato come input nel Test di Cohen-Lenstra siamo sicuri che esso termini in tempo lineare, non rappresentando asintoticamente alcun costo aggiuntivo rispetto al costo dominante dell'intero algoritmo.
Illustriamo il passo principale dell'algoritmo:

[1] Nella versione standard proposta vogliamo che m sia esattamente il doppio di q, ma andrebbe bene comunque se fosse un qualsiasi multiplo.

Generate_Curve (n)

1. generare casualmente (A, B) fino a che $(n, 4A^3 + 27B^2) = 1$;
2. calcolare $m = \#E_n(A, B)(\mathbb{Z}/n\mathbb{Z})$ usando Schoof;
3. se m è pari porre $q = m/2$, altrimenti tornare al passo 1);
4. testare $2k$ volte la primalità di q con Miller-Rabin:
   se almeno un test restituisce 0 tornare al passo 1);
   se $2, 3 \mid q$ tornare al passo 1);
5. RETURN $(A, B), q$.

Select_Point $(n, q, (A, B))$

1. scegliere uniformemente $x \in \mathbb{Z}/n\mathbb{Z}$ fino a che $z = x^3 + Ax + B$
   sia un residuo quadratico;
2. calcolare $y = \sqrt{z}$ con un algoritmo polinomiale di estrazione
   di radici, scegliendo uniformemente quale radice prendere,
   e porre $P = (x, y)$;
3. calcolare qP usando l'algoritmo delle duplicazioni successive:
   se $qP \neq \infty$ tornare al passo 1), altrimenti RETURN P.

Main_Step (n)

1. $(A, B), q \leftarrow$ Generate_Curve (n);
2. $P \leftarrow$ Select_Point $(n, q, (A, B))$;
3. RETURN $(A, B), P, q$.

L'algoritmo completo itera il passo principale finché il primo da verificare diventi sufficientemente piccolo per essere certificato in tempo lineare. Inoltre inseriamo una condizione di arresto: per evitare che l'algoritmo entri in un loop cercando di provare la primalità di un numero in realtà composto (situazione causata dall'eventuale errore di un test probabilistico), l'algoritmo si riavvia dopo aver eseguito di un certo gran numero di operazioni.

Prove_Prime (n)

1. $i \leftarrow 0$; $q_0 \leftarrow n$; $lowerbound \leftarrow \max(2^{k^{C/\log\log k}}, 37)$;
2. mentre $q_i > lowerbound$:
3. se $2, 3 \mid q_i$ tornare al passo 1);
4. $(A_i, B_i), P_i, q_{i+1} \leftarrow$ Main_Step (q_i);
5. $i \leftarrow i + 1$;
6. usare il Cohen-Lenstra su q_i :
   se q_i non è primo tornare al passo 1),
   altrimenti RETURN $((A_0, B_0), P_0, q_1), \ldots, ((A_{i-1}, B_{i-1}), P_{i-1}, q_i)$.

* Se dal passo 1) sono state eseguite più di $k^{\log k}$ operazioni,
interrompere e riprendere dal passo 1).

La scelta del *lowerbound* serve a non dover verificare la condizione $q_{j+1} > (q_j^{1/4}+1)^2$, in quanto si ha che:

$$q_{j+1} \geq \frac{q_j + 1 - 2\sqrt{q_j}}{2} > (q_j^{1/4}+1)^2$$

per $q_j > 37$.

Chiamiamo:

$$\texttt{certificate} \;=\; ((A_0, B_0), P_0, q_1), \ldots, ((A_{i-1}, B_{i-1}), P_{i-1}, q_i)$$

l'output di `Prove_Prime`. Il programma di verifica prende in input n e `certificate`, e comincia col verificare se q_i è abbastanza piccolo per essere usato come input nel Test di Cohen-Lenstra. In caso affermativo si verifica la primalità di q_i. Riportiamo l'algoritmo di verifica:

`Check` $(n,$ `certificate`$)$

1. `sospendere se` $q_i > \max(2^{k^{C/\log\log k}}, 37)$,
   `altrimenti verificare la primalità di` q_i `usando il Cohen-Lenstra;`
2. $q_0 \leftarrow n$;
3. `per` $j = i-1, \ldots, 0$ `verificare che:`
   - $2, 3 \nmid q_j$;
   - $(q_j, 4A_j^3 + 27B_j^2) = 1$;
   - $q_{j+1} > (q_j^{1/4}+1)^2$;
   - $P_j \neq \infty$ e $q_{j+1}P_j = \infty \pmod{q_j}$;

   `sospendere se una di queste condizioni non è verificata;`
4. `accettare` n `come primo.`

Il procedimento logico della verifica è il seguente:

$$q_i \text{ è primo } \Rightarrow q_{i-1} \text{ è primo } \Rightarrow \ldots \Rightarrow q_0 = n \text{ è primo.}$$

L'ECPP di Goldwasser-Kilian ha le seguenti proprietà:

- dato un input di lunghezza k, esso produce un certificato di primalità di lunghezza $O(k^2)$ che verifica in modo deterministico la sua primalità in un tempo polinomiale $O(k^4)$;
- esso termina in tempo atteso polinomiale per ogni primo, a patto che sia vera la seguente congettura:

  $$\exists\, c_1, c_2 > 0 \;:\; \pi(x+\sqrt{x}) - \pi(x) \geq \frac{c_2\sqrt{x}}{\log^{c_1} x}$$

  per x sufficientemente grande;

- esistono costanti c_1 e c_2 tali che, per k sufficientemente grande, esso termini in tempo atteso $c_1 k^{11}$ per tutti tranne al massimo

$$\frac{2^k}{2^{k^{c_2/\log\log k}}}$$

degli inputs.

Il Teorema dei Numeri Primi suggerisce (ma non implica) tale congettura con $c_1 = 1$, mentre l'attendibile congettura di Cramer:

$$\pi(x + \log^2 x) - \pi(x) > 0, \qquad \text{per } x \to \infty$$

dimostrerebbe la nostra con $c_1 = 2$. Inoltre le proprietà elencate implicano l'esistenza di un insieme infinito di primi riconoscibili in tempo atteso polinomiale. La nostra congettura implica che tale insieme conterrebbe tutti i primi di lunghezza k, per ogni $k > 0$.

References

[1] Adleman L.M., Huang A. *Primality Testing and Abelian Varieties over Finite Fields*, "Lecture Notes in Mathematics", 1512, Springer-Verlag, 1992.

[2] Agrawal M., Kayal N., Saxena N. *PRIMES in P*, "Annals of Mathematics", 160, 2004, 781-793.

[3] Atkin A., Morain F. *Elliptic Curves and Primality Proving*, "Mathematics of Computation", Volume 61, 1993, 29-67.

[4] Blake I.F., Seroussi G., Smart N.P. *Elliptic Curves in Cryptography*, volume 265 of London Mathematical Society Lecture Note Series, Cambridge University Press, Cambridge 2000. Reprint of 1999 original.

[5] Brent R.P. *Primality Testing*, MSI & CECS, ANU, 2010.

[6] Brent R.P. *Some integer factorization algorithms using elliptic curves*, "Australian Computer Science Communications" 8, 1986, 149-163.

[7] Cohen H., Lenstra H. *Primality Testing and Jacobi Sums*, "Mathematics of Computation", Volume 42, 1984.

[8] Cheng Q. *Primality Proving via one round in ECPP and one iteration in AKS*, "Journal of Cryptology", 2003, 375-387.

[9] Crandall R., Pomerance C. *Prime Numbers: A Computational Perspective*, Springer, 2005.

[10] Goldwasser S., Kilian J. *Almost all primes can be quickly certified*, "Proceedings of the 18-th Annual ACM Symposium on Theory of Computing", New York 1986, 316-329.

[11] Goldwasser S., Kilian J. *Primality Testing using Elliptic Curves*, "Journal of the ACM", 46, No. 4, 1999, 450-472.

[12] Granville A. *Smooth Numbers: computational number theory and beyond*, "Surveys in algorithmic number theory", MSRI Publications, Volume 44, New York 2008, 267-323.

[13] Koblitz N. *A course in number theory and cryptography, 2ed.*, GTM 114, Springer, 1994.

[14] Languasco A., Zaccagnini A. *Introduzione alla Crittografia*, Ulrico Hoepli, Milano 2004.

[15] Lenstra H. *Elliptic Curves and Number-Theoretic Algorithms*, "Proceedings of the International Congress of Mathematicians", AMS, Volume 1,2 (Berkeley, Calif., 1986), 99-120, Providence, RI, 1987.

[16] Lenstra H. *Factoring Integers with Elliptic Curves*, "Annals of Mathematics", 126 (1987), 649-673.

[17] Machì A. *Gruppi*, Springer-Verlag Italia, Milano 2007.

[18] Montgomery P.L. *Speeding the Pollard and Elliptic Curve Methods of Factorization*, "Mathematics of Computation", Volume 48, Issue 177, Jan. 1987, 243-264.

[19] Morain F. *Implementing the Asymptotically Fast Version of the Elliptic Curve Primality Proving Algorithm*, INRIA, 2005.

[20] Pomerance C. *Smooth Numbers and the Quadratic Sieve*, "Algorithmic Number Theory", MSRI Publications, Volume 44, 2008, 69-81.

[21] Poonen B. *Elliptic Curves*, "Algorithmic Number Theory", MSRI Publications, 2008, 1-19.

[22] Schoof R. *Four Primality Testing Algorithms*, "Algorithmic number theory", MSRI Publications, Volume 44, Cambridge University Press 2008, 101-126.

[23] Silverman J.H. *The Arithmetic of Elliptic Curves*, Springer-Verlag, 1986.

[24] Washington L.C. *Elliptic Curves, number theory and cryptography*, Chapman & Hall/CRC, 2003.

[25] Waterhouse W.C. *Abelian Varieties over Finite Fields*, Ann. Sci. Ecole Norm. Sup., 1969.