Matematica, Cultura e Società

Rivista dell'Unione Matematica Italiana

Alessandro Zaccagnini

$2021 = 43 \cdot 47$

Matematica, Cultura e Società. Rivista dell'Unione Matematica Italiana, Serie 1, Vol. **7** (2022), n.2, p. 121–133.

Unione Matematica Italiana

¡http://www.bdim.eu/item?id=RUMI_2022_1_7_2_121_0¿

Matematica, Cultura e Società. Rivista dell'Unione Matematica Italiana, Unione Matematica Italiana, 2022.

Matematica, Cultura e Società – Rivista dell'Unione Matematica Italiana
Serie I, Vol. 7, N. 2, agosto 2022, 121-133

$2021 = 43 \cdot 47$

Alessandro Zaccagnini
Università di Parma
E-mail: alessandro.zaccagnini@unipr.it

Sommario: *Questo articolo è dedicato ad un esame critico di alcuni metodi di fattorizzazione per i numeri interi, evidenziando i punti di forza e le criticità. Oltre ai metodi classici della fattorizzazione per tentativi, quello di Fermat e quello di Lehman, si descrivono in maggiore dettaglio due metodi moderni, il crivello quadratico e quello con i campi di numeri.*

Abstract: *This paper is devoted to a critical survey of factorisation algorithms for integers, underlining strenghts and weaknesses. Beside classical algorithms, like trial division and the methods due to Fermat and Lehman, we give more detailed descriptions of modern methods like the Quadratic Sieve and the Number Field Sieve.*

> Problema, numeros primos a compositis dignoscendi, hosque in factores suos primos resolvendi, ad gravissima ac utilissima totius arithmeticæ pertinere, et geometrarum tum veterum tum recentiorum industriam ac sagacitatem occupavisse, tam notum est, ut de hac re copiose loqui superfluum foret [...] Prætereaque scientiae dignitas requirere videtur, ut omnia subsidia ad solutionem problematis tam elegantis ac celebris sedulo excolantur
>
> C. F. Gauss, *Disquisitiones Arithmeticæ*, 1801, Art. 329

1. – Perché scomporre in fattori primi

Riportiamo in epigrafe un celebre brano di Gauss sulla fattorizzazione dei numeri interi, in cui dice in termini molto chiari che è un problema degno di studio approfondito. La conoscenza della scomposizione in fattori primi di un intero positivo n permette di rispondere ad alcune domande molto naturali: ne elenchiamo qualcuna, ma senza pretesa di esaurirle. Quanti e quali sono i divisori di n? Quanti sono i gruppi abeliani di ordine n, a meno di isomorfismi? Quanto vale $\varphi(n)$, cioè quanti sono gli interi che non superano n e sono coprimi con n?

Una domanda con applicazioni pratiche è questa: quanto è difficile "attaccare" crittosistemi come il popolare RSA? In altre parole, quanto sono sicuri i moderni metodi crittografici che si basano proprio sulla difficoltà di scomporre in fattori primi? Detta in un altro modo: quanto grandi dobbiamo prendere le chiavi di cifratura nei crittosistemi a chiave pubblica per garantire il livello minimo di sicurezza che intendiamo accettare?

E ancora: le eventuali soluzioni razionali di un'equazione polinomiale a coefficienti interi, ridotte ai minimi termini, hanno numeratore e denominatore che dividono rispettivamente l'ultimo e il primo coefficiente; quindi, la dimostrazione del fatto che un certo polinomio a coefficienti interi non ha soluzioni razionali può essere ricondotta ad un numero *finito* di verifiche, a patto di conoscere tutti i divisori di due soli interi, cioè la loro fattorizzazione.

In questo articolo non ci occuperemo delle applicazioni, siano esse "interne" o "esterne" alla matematica, ma esclusivamente del problema di determinare in modo efficiente la scomposizione in fattori primi di un numero intero positivo.

2. – Quanti sono i fattori primi di un numero intero?

Scomporre un numero intero nei suoi fattori primi è un'operazione tutt'altro che facile: prima ancora di

Accettato: il 6 luglio 2022.

esaminare le possibili strategie, cerchiamo di farci un'idea delle dimensioni del problema. Il "tipico" numero intero N ha circa $\log(\log N)$ fattori primi, mentre fino ad N ci sono circa $N/\log N$ numeri primi, cioè potenziali fattori di N. Come vedremo qui sotto e come è ben noto, se vogliamo scomporre N in fattori possiamo limitare la nostra ricerca ai numeri primi che non superano $N^{1/2}$, ma anche in questo caso il numero dei potenziali fattori primi di N è enorme. Questi sono rispettivamente i Teoremi di Hardy & Ramanujan, e il Teorema dei Numeri Primi di Hadamard e de la Vallée Poussin. In tutto questo articolo, log indica la funzione logaritmo naturale.

Prendiamo per esempio $N \approx 10^{100}$, cioè un intero alla soglia delle applicazioni crittografiche; per quanto detto ci possiamo aspettare che un tipico N di queste dimensioni abbia $\approx \log\log(10^{100}) \approx 5$ fattori primi distinti, mentre ci sono circa $10^{50}/\log(10^{50}) \approx 9 \cdot 10^{47}$ numeri primi fino a $N^{1/2}$.

Di conseguenza, abbiamo bisogno di un procedimento sistematico di ricerca dei fattori primi di N che sia parecchio più sofisticato del più noto algoritmo della divisione per tentativi che descriveremo nel prossimo paragrafo. Naturalmente non è possibile sapere a priori se il numero N che desideriamo scomporre in fattori sia più o meno "tipico." Gli algoritmi che descriveremo hanno punti di forza e, viceversa, debolezze, che dipendono proprio da questo.

3. – Come scomporre in fattori primi: per tentativi

L'algoritmo detto "fattorizzazione per tentativi" consiste nel dividere successivamente l'intero N per tutti i numeri primi che non superano $N^{1/2}$ sperando di trovarne un fattore. Ci si può fermare a $N^{1/2}$ perché se $N = ab$ con $1 < a \leq b < N$ è una scomposizione non banale di N in due fattori, non necessariamente primi, allora $a \leq b = N/a$ e quindi $a \leq N^{1/2}$. In particolare, se N è composto allora ha almeno un fattore primo che non supera $N^{1/2}$.

Il primo e non unico problema che si incontra se si vuole realizzare in pratica questo algoritmo consiste nel procurarsi la lista dei numeri primi fino a $N^{1/2}$: il crivello di Eratostene nelle sue tante varianti è ancora molto utilizzato (vedi anche il recente miglioramento di Harald Helfgott), ma non è materialmente possibile conservare la lista di *tutti* i numeri primi fino a 10^{50}, per tornare all'esempio qui sopra. Se la lista dei numeri primi non è disponibile, possiamo tentare di dividere N per tutti gli interi dispari fra 3 ed $N^{1/2}$; in alternativa, si possono usare dei trucchi come la cosiddetta "ruota" che permettono di ridurre il numero di divisioni necessarie. Si tratta di miglioramenti marginali che sono rilevanti solo dal punto di vista dell'implementazione, e non ce ne occuperemo se non per un breve commento.

Supponiamo di avere a disposizione la lista dei numeri primi fino ad M, con $M \geq 5$, molto più piccolo

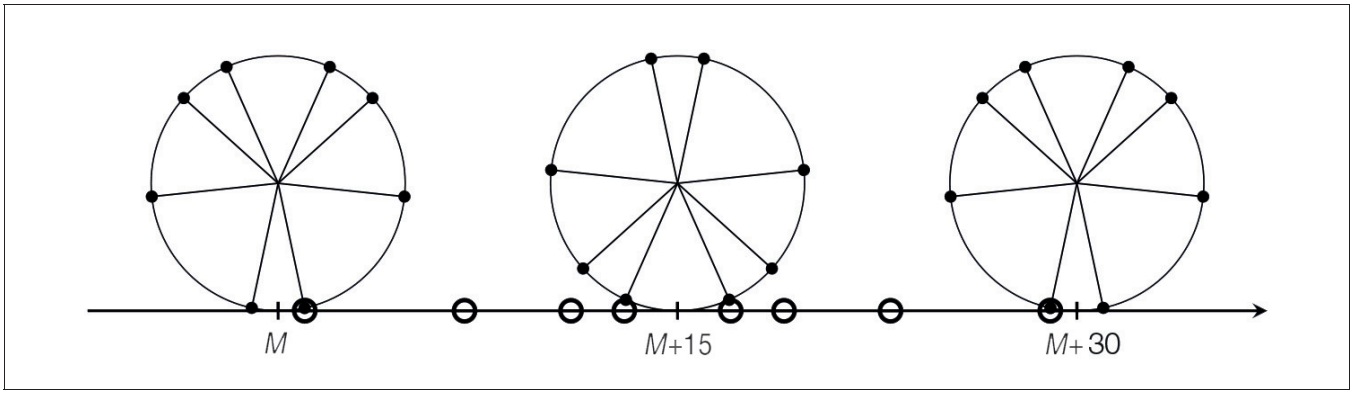

FIGURA 1 – La ruota dei numeri primi: per semplicità in questo disegno supponiamo che M sia un multiplo di 30. I puntini sulle circonferenze corrispondono alle classi di resto 1, 7, 11, 13, 17, 19, 23, 29 modulo 30. Indichiamo a sinistra la posizione iniziale della ruota, al centro la posizione dopo mezzo giro, a destra la posizione finale dopo un giro intero. Facendo rotolare la circonferenza lungo l'asse delle ascisse i puntini toccano la retta nei punti indicati dai circoletti vuoti. In questo modo individuiamo la sequenza di interi $M+1, M+7, M+11, \ldots, M+29, M+31, M+37, M+41, \ldots$ che utilizziamo come possibili divisori di N. Dunque siamo certi di non omettere numeri primi, e di tentare relativamente poche divisioni inutili per numeri composti.

di $N^{1/2}$. Verifichiamo se N è divisibile o meno per uno di questi numeri primi e supponiamo di non aver trovato fattori. Possiamo usare come potenziali fattori di N (non necessariamente primi) i numeri dispari fra M ed $N^{1/2}$, con un certo "spreco" di operazioni, perché ogni tanto ci capiterà di dividere N per un numero composto m, mentre già sappiamo che N non può essere divisibile per m. Al fine di ridurre lo spreco senza complicare eccessivamente l'algoritmo possiamo sfruttare una piccola osservazione che generalizza l'idea di considerare solo fattori dispari. I numeri primi, tranne 2, 3 e 5, sono confinati nelle otto classi di resto 1, 7, 11, 13, 17, 19, 23, 29 modulo 30. Quindi, possiamo limitarci ad utilizzare come potenziali fattori di N solo i numeri tra M e $N^{1/2}$ che si trovano in una di queste otto classi. Il meccanismo si chiama ruota a causa della periodicità: possiamo visualizzarlo mediante la Figura 1.

Al posto di $30 = 2 \cdot 3 \cdot 5$ possiamo usare $210 = 2 \cdot 3 \cdot 5 \cdot 7$, con una lista di $\varphi(210) = 48$ classi di resto ammissibili, o interi più grandi con tanti fattori primi piccoli e le corrispondenti liste di classi di resto: il meccanismo è lo stesso, con qualche piccolo miglioramento nella performance e qualche piccola complicazione nella determinazione e nella gestione delle liste delle classi di resto da includere. Come dicevamo, si tratta comunque di miglioramenti marginali che non cambiano in modo essenziale la natura del problema.

Il metodo della fattorizzazione per tentativi è efficiente per determinare i fattori primi "piccoli" di N, sempre ammesso che ne abbia, ma orribilmente inefficiente se N ha solo fattori primi relativamente grandi, cioè vicini alla sua radice quadrata, come nel caso di $2021 = 43 \cdot 47$. In effetti è un algoritmo ancora molto usato per "rimuovere" i fattori primi piccoli dal numero N che vogliamo fattorizzare, se ne ha, prima di passare ad algoritmi più efficienti. Nei prossimi paragrafi vedremo lo sviluppo di alcune idee alternative in merito.

3.1 – *Osservazioni conclusive sul metodo della "divisione per tentativi."*

Una volta superata $N^{1/2}$ senza successo abbiamo dimostrato che N è primo, che è più di quanto ci interessa. È ben noto che esistono criteri di primalità o pseudo-primalità piuttosto efficienti che derivano in ultima analisi dal Piccolo Teorema di Fermat. Tra questi citiamo AKS, dai nomi di Agrawal, Kayal & Saxena che l'hanno scoperto nel 2004: per la prima volta è stato dimostrato che il problema di decidere se un numero intero N è primo o meno ha una complessità computazionale polinomiale, cioè il numero di operazioni elementari per portare a termine questo criterio è $\mathcal{O}((\log N)^C)$, dove C è una certa costante positiva esplicita. Nell'articolo originale il valore dato è $C = 12$, ma oggi sappiamo che $C = 6$ è ammissibile. Il problema principale di questo algoritmo è la difficoltà dell'implementazione, e anche il fatto che perfino con $C = 6$ il numero di operazioni risulta molto alto. Nella pratica si preferisce spesso utilizzare criteri, come quello di Miller & Rabin, che sono più semplici da implementare e sono più efficienti, ma che riescono solo a garantire che un certo intero è primo con probabilità molto vicina ad 1; per questo motivo si parla di pseudo-primalità. Viceversa, quando questi criteri affermano che un certo intero è composto la risposta è certa, anche nel caso in cui non si riesca ad esibirne esplicitamente un fattore.

In conclusione, questi criteri sono enormemente più efficienti degli algoritmi di fattorizzazione oggi noti e devono quindi essere usati preliminarmente sull'intero N che vogliamo scomporre in fattori primi, perché non vogliamo tentare di scomporre in fattori primi un intero N per poi scoprire che è un numero primo.

Prima di proseguire, proviamo a formulare il nostro obiettivo in modo più preciso. Facciamo sempre tacitamente l'ipotesi non restrittiva che l'intero N che vogliamo scomporre sia grande, dispari, non un quadrato perfetto, composto. Come detto sopra, i calcoli relativi sono piuttosto semplici da effettuare se ci si accontenta di verificare la pseudo-primalità di N, mentre per accertare la primalità vera e propria di N è necessario usare il criterio AKS che è più difficile da realizzare in pratica.

Possiamo anche supporre di aver verificato che N non sia divisibile per nessuno dei numeri primi fino ad M, dove $M \geq 5$ è il limite delle tavole dei numeri primi di cui disponiamo, o comunque è il limite della fattorizzazione per tentativi che consideriamo accettabile. Il nostro obiettivo è dunque spezzare N in due fattori non banali, non necessariamente primi, sui

quali iterare il procedimento: una volta trovati a e b tali che $N = ab$ con $1 < a \leq b < N$, applichiamo ricorsivamente il nostro procedimento di verificare l'eventuale primalità di a e b; in caso uno dei due o entrambi risultino composti, continuiamo con un algoritmo di fattorizzazione ripetendolo finché necessario. Come detto prima, un "tipico" intero non richiede molte iterazioni perché ha relativamente pochi fattori primi, mentre gli interi "atipici," cioè con molti fattori primi, sono evidentemente più facili da spezzare in fattori più piccoli; quindi richiedono più iterazioni di questa procedura ricorsiva, ma con numeri più piccoli. Come vedremo sotto, il caso "difficile" è quello in cui N ha pochi fattori primi di dimensioni molto diverse fra loro.

Per raggiungere il nostro scopo, e cioè algoritmi alternativi alla semplice divisione per tentativi, utilizzeremo qualche risultato di teoria elementare dei numeri e anche qualche teorema proveniente da altre parti della matematica. È forse controintuitivo che sia possibile o necessario rinunciare a tentare di fattorizzare N provando tutti i possibili candidati in sequenza, ma come abbiamo osservato all'inizio nel §2 questo approccio è inefficiente a causa del numero dei tentativi necessari nel caso pessimo, e in realtà nella maggior parte dei casi "standard." Proviamo dunque ad attaccare questo problema da un altro fronte. In un certo senso, il metodo che stiamo per illustrare, invece di cercare i fattori piccoli di N cerca quelli di dimensione "media," vicino alla sua radice quadrata.

4. – Il metodo di Fermat

Fermat è partito da una semplice osservazione: se troviamo due interi x ed y tali che $N + y^2 = x^2$ allora $N = (x - y)(x + y)$ e quindi lo abbiamo spezzato in due fattori. C'è un caso banale, quando $x - y = \pm 1$ oppure $x - y = \pm N$, ma qualunque altra soluzione permette di spezzare N in fattori più piccoli. Questi fattori non sono necessariamente primi, ma possiamo iterare il procedimento come detto sopra.

In pratica prendiamo y piccolo e verifichiamo in sequenza se $N + y^2$ è un quadrato perfetto: consideriamo gli interi N, $N + 1$, $N + 4$, ... In alternativa, possiamo prendere come x_0 il più piccolo intero che supera $N^{1/2}$ e considerare la successione $x_0^2 - N$, $(x_0 + 1)^2 - N$, ... e verificare se uno di questi interi è un quadrato perfetto. Questo accade certamente quando $x = \frac{1}{2}(N + 1)$ e quindi l'algoritmo termina in un numero finito di passi, nel caso peggiore trovando la soluzione banale in cui $x - y = 1$ ed $x + y = N$.

Esistono metodi efficienti per verificare se un certo intero Q sia o meno un quadrato perfetto: si può usare una variante dell'algoritmo di Newton, oppure usare le congruenze, come faremo più avanti in un contesto diverso, per escludere molto velocemente che Q sia un quadrato ed evitare del tutto verifiche più onerose dal punto di vista computazionale. Per esempio, modulo 16 i quadrati perfetti sono solo 0, 1, 4, 9 e quindi è sufficiente esaminare gli ultimi 4 *bit* di Q: se Q mod 16 non ha uno di questi quattro valori, allora Q non è un quadrato perfetto; in caso contrario procediamo con l'algoritmo di Newton, o magari usiamo congruenze modulo qualche altro intero per escludere ancora qualche caso.

Calcoliamo ora il numero di soluzioni dell'equazione $N = x^2 - y^2$, dove N è un numero intero dispari. Questa ipotesi per noi non è una vera limitazione, come abbiamo già osservato. Se dunque N è un numero intero dispari, quante sono le soluzioni intere non negative (x, y) di $x^2 - y^2 = N$? Se $d \geq 1$ è un divisore di N allora troviamo una soluzione ponendo

$$(1) \qquad \begin{cases} x - y = d \\ x + y = \dfrac{N}{d} \end{cases} \quad \Longrightarrow \quad \begin{cases} x = \dfrac{N + d^2}{2d} \\ y = \dfrac{N - d^2}{2d} \end{cases}$$

Se ci limitiamo a considerare interi dispari N che non siano quadrati perfetti e divisori d tali che $1 \leq d < N^{1/2}$, vediamo che i valori di x ed y restituiti dalle formule (1) sono entrambi positivi, e viceversa. Dunque il numero delle soluzioni positive dell'equazione $x^2 - y^2 = N$ è $\frac{1}{2}d(N)$, dove $d(N)$ è il numero dei divisori di N. Di queste, tutte meno una, quella con $d = 1$, portano a scomporre N in fattori più piccoli.

Un esempio particolarmente interessante, in vista delle applicazioni crittografiche a cui abbiamo alluso sopra, è quello in cui $N = pq$ dove $p > q$ sono primi distinti come in RSA: in questo caso c'è una sola soluzione utile, e cioè $x = \frac{1}{2}(p + q)$, $y = \frac{1}{2}(p - q)$.

Dunque, il numero di "tentativi" o meglio di iterazioni necessarie a far terminare l'algoritmo vale essenzialmente $p - q$ e quindi la complessità può essere molto grande se i due fattori primi di N sono molto sbilanciati fra loro.

4.1 – *Confronto fra i due metodi*

Come abbiamo visto sopra, il metodo della divisione per tentativi fa fatica a scomporre nei loro fattori primi gli interi N che sono prodotto di due numeri primi relativamente vicini a $N^{1/2}$, mentre identifica velocemente gli eventuali fattori primi di N molto piccoli. Viceversa, il metodo di Fermat funziona bene proprio per gli interi sui quali il metodo della fattorizzazione per tentativi è poco efficace, mentre richiede "troppe" iterazioni per numeri della forma $N = 3p$ dove p è un numero primo grande, per le formule che abbiamo visto qui sopra. Infatti, il numero di iterazioni necessarie è $y \approx \frac{1}{2}p = \frac{1}{6}N$.

4.2 – *Il numero di Fermat* $F = 2\,027\,651\,281$

Lasciamo ai nostri lettori l'esercizio di scomporre in fattori primi il numero $F = 2\,027\,651\,281$. È il numero con il quale Fermat stesso illustrò il suo metodo di fattorizzazione. Ricordiamo che stiamo cercando due interi positivi x ed y tali che $x^2 - y^2 = F$, cioè $x^2 = F + y^2$. Da questa relazione segue immediatamente che $x \geq s = 45\,030$, che è la radice quadrata di F arrotondata all'intero successivo. Possiamo dunque procedere per tentativi, assegnando ad y i valori successivi $1, 2, 3, 4, \ldots$ e verificando volta per volta se $F + y^2$ sia o meno un quadrato perfetto, oppure cominciare da $x = s$ e verificare se $x^2 - F$ sia un quadrato perfetto.

Daremo la risposta nel corso dell'articolo. Nel §6 proveremo invece a ridurre la "dimensione" del problema utilizzando le congruenze.

5. – Il metodo di Lehman

Abbiamo notato come i due algoritmi di fattorizzazione per tentativi e di Fermat abbiano una complessità alta, ciascuno nel proprio caso pessimo. Un'osservazione interessante è che i due casi pessimi sono in un certo senso l'uno l'opposto dell'altro. Purtroppo nessuno sa dire a priori se N ha fattori primi piccoli o grandi, e quindi se sia più opportuno procedere con il primo o con il secondo algoritmo.

Inoltre, c'è un'enorme "terra di nessuno" fra la regione in cui è efficiente la divisione per tentativi e quella in cui è efficiente il metodo di Fermat: vedi la Figura 2. Cerchiamo dunque un algoritmo che sia efficace nella regione indicata in bianco nella Figura. Il metodo di Lehman fornisce un bilanciamento dei due algoritmi che abbiamo discusso sopra e proprio per questo motivo risulta più efficiente di entrambi.

Proviamo a visualizzare la situazione ricorrendo ad un esempio familiare a tutti: la tavola pitagorica. L'intero N compare nella tavola pitagorica $N \times N$, che chiameremo per brevità $\mathcal{TP}(N)$, esattamente $d(N)$ volte, dove d indica il numero totale dei divisori di N. Se N è primo non compare nella tavola pitagorica $\mathcal{TP}(N-1)$ né in quelle di dimensioni più

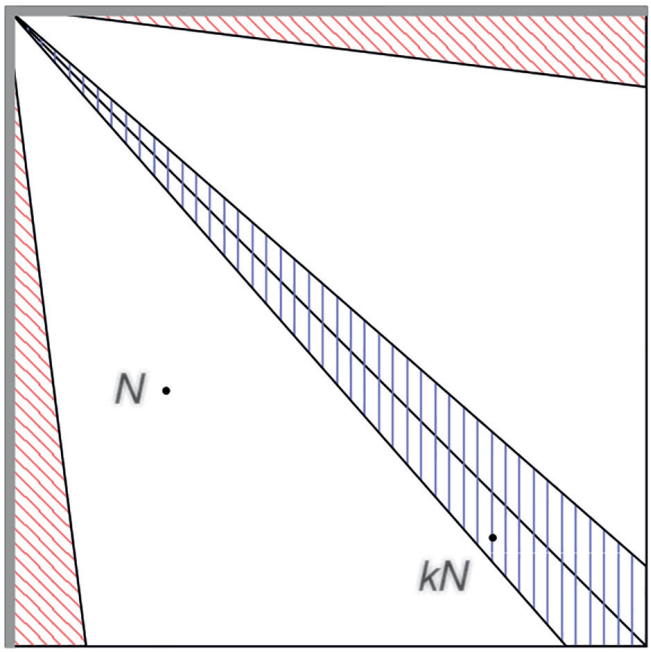

FIGURA 2 – Dalla tavola pitagorica togliamo la prima riga e la prima colonna; nella zona campita in diagonale è efficiente la fattorizzazione per tentativi, in quella campita in verticale l'algoritmo di Fermat. Naturalmente, la regione bianca, quella in cui entrambi gli algoritmi sono inefficienti, rappresenta la maggior parte ...

Nella seconda fase dell'algoritmo di Lehman si cerca un intero k moderatamente grande per il quale kN sia nella "zona di influenza" del metodo di Fermat. Nel nostro esempio, il numero F compare per la prima volta nella tavola pitagorica $\mathcal{TP}(46061)$, mentre $1935F$ compare per la prima volta nella tavola $\mathcal{TP}(1980945)$, che è ben più grande, ma vicinissimo alla diagonale.

piccole, perché si può trovare solo nella prima riga o nella prima colonna. Se invece N è composto allora compare anche in tavole pitagoriche di dimensione più piccola. Ci chiediamo dunque quale sia la più piccola tavola pitagorica su cui compare N. Per esempio, se N è un quadrato perfetto, allora compare già sulla tavola pitagorica $\mathcal{TP}(N^{1/2})$, precisamente sulla sua diagonale. In generale, N compare per la prima volta nella tavola pitagorica $\mathcal{TP}(m)$ dove m è il più piccolo divisore di N tale che $m \geq N^{1/2}$.

Tornando al nostro problema, se N compare molto vicino al bordo superiore o sinistro (escluse per ovvi motivi la prima riga e la prima colonna), allora la fattorizzazione per tentativi funziona bene, quanto meno per eliminare da N i suoi fattori primi piccoli; se compare anche molto vicino alla diagonale, allora funziona bene il metodo di Fermat. Purtroppo, la stragrande maggioranza degli interi si trova nella "terra di mezzo" fra queste due situazioni e comunque, come già detto, non è possibile stabilire a priori in quale situazione ci troviamo.

Informalmente, l'algoritmo di Lehman funziona cosí. La prima fase consiste nel tentare la fattorizzazione, come abbiamo descritto nel §3, ma, in caso di insuccesso ci si ferma ad $R = N^{1/3}$ invece di arrivare ad $N^{1/2}$. Poiché per ipotesi di partenza N è composto, non è difficile vedere che ha esattamente *due* fattori primi p e q che soddisfano $R \leq p \leq N^{1/2} \leq q \leq R^2$, perché N non può avere più di due fattori primi $> R$. A questo punto comincia la seconda fase: cerchiamo $k \in \mathbb{N}$ in modo tale che kN abbia due fattori (non primi) di dimensioni comparabili, cioè in modo che kN si trovi nella regione adiacente alla diagonale della Figura 2. Usando la teoria delle frazioni continue, si può dimostrare che la ricerca di un valore di k appropriato richiede al massimo un numero di iterazioni dell'ordine di grandezza di R; ciascuna iterazione poi richiede qualche operazione elementare e l'estrazione di una radice quadrata intera. Quindi la complessità totale dell'algoritmo di Lehman è poco più grande di $\mathcal{O}(N^{1/3})$.

La spiegazione dettagliata del funzionamento dell'algoritmo non è indispensabile, ma l'idea di base è molto interessante ed è giusto dedicarle un paragrafo. Come dicevamo, cerchiamo un intero k moderatamente grande tale che kN si spezzi in due fattori non primi ma di grandezza confrontabile; vogliamo che la fattorizzazione di kN sia attaccabile dal metodo di Fermat.

Consideriamo le frazioni di Farey $\mathcal{F}(Q)$ di livello $Q \geq 1$, cioè l'insieme

$$\left\{\frac{a}{b} : 1 \leq a \leq b \leq Q \wedge \mathrm{mcd}(a,b) = 1\right\}.$$

Si tratta delle frazioni "proprie," ridotte ai minimi termini, il cui denominatore non supera il parametro Q. Dalla teoria delle frazioni continue sappiamo che dato $x \in [0,1]$ esiste un elemento $a_0/b_0 \in \mathcal{F}(Q)$ tale che

$$\left|x - \frac{a_0}{b_0}\right| \leq \frac{1}{b_0 Q}. \tag{2}$$

Applichiamo questa cosa ad $N = pq$, dove ricordiamo che p e q sono da determinare, prendendo $x = p/q$. La disuguaglianza qui sopra dice che una delle frazioni di $\mathcal{F}(Q)$ è vicinissima a p/q, ma quale? Omettiamo il dettaglio di questa fase, concentrandoci sulle conseguenze della disuguaglianza (2): ne deduciamo immediatamente che

$$|pb_0 - a_0 q| \leq \frac{q}{Q}.$$

Il nostro obiettivo è trovare k relativamente piccolo in modo tale che $kN = x^2 - y^2$ con y più piccolo possibile: prendiamo $k = a_0 b_0$. Utilizzando le formule (1) con kN al posto di N e $d = a_0 q$, troviamo che

$$|y| = \frac{1}{2}|pb_0 - a_0 q| \leq \frac{q}{2Q}.$$

Se Q è grande, allora $|y|$ è piccolo e quindi il metodo di Fermat funziona bene; d'altra parte, l'insieme $\mathcal{F}(Q)$ ha cardinalità che cresce proporzionalmente a Q^2 e quindi la fase di ricerca di a_0 e b_0 è più lunga. Dall'equilibrio fra queste due esigenze segue la scelta ottimale di Q e in definitiva la stima della complessità computazionale a cui abbiamo accennato prima.

Tornando alla nostra analogia della tavola pitagorica, il numero $kN = a_0 b_0 pq$ si trova molto vicino alla diagonale e quindi nella "regione" dove il metodo di Fermat è efficiente.

5.1 – *Un esempio*

Vediamo come usare in pratica l'informazione $a_0/b_0 \approx p/q$. Nel caso del numero di Fermat F

sappiamo, ovviamente a posteriori, che $\frac{p}{q} \approx \frac{43}{45}$ e quindi l'intero $M = 43 \cdot 45 \cdot p \cdot q = 1935F$ si spezza in due fattori molto vicini alla sua radice quadrata, che vale $\approx 1\,980\,784$. In effetti,

$$M = 1935F = (1980784 + 161) \cdot (1980784 - 161).$$

Infine scopriamo un fattore primo di F calcolando $\mathrm{mcd}(1\,980\,784 + 161, F)$ poiché $1\,980\,784 + 161 = 45 \cdot 44021$.

Un piccolo chiarimento finale è doveroso: calcoliamo il $\mathrm{mcd}(x + y, N)$ usando l'algoritmo di Euclide, ovviamente, e non quello che talvolta si insegna a scuola, che richiede la fattorizzazione completa di $x + y$ ed N, cioè ci porterebbe ad un circolo vizioso!

6. – Introduciamo le congruenze

Prima di passare ad illustrare i metodi più moderni, sviluppiamo una semplice idea che ci permette di ridurre il numero di tentativi dei metodi illustrati finora, che contengono una fase di ricerca esaustiva. Di per sé questa idea non ci fornisce un metodo di fattorizzazione, ma permette di ridurre lo spreco di operazioni intrinseco negli algoritmi già descritti. Anche la "ruota" citata qui sopra ha lo scopo di ridurre il numero di operazioni inutili, in un contesto diverso, ma in fondo anche in quel caso teniamo conto di congruenze simultanee modulo numeri primi piccoli. Nei paragrafi successivi vedremo invece la descrizione dettagliata del crivello quadratico e del crivello con i campi di numeri in cui le congruenze giocano un ruolo fondamentale.

Partiamo da un'osservazione sostanzialmente banale:

$$\text{(3)} \qquad \text{se } x^2 = N + y^2 \text{ allora } x^2 \equiv N + y^2 \bmod n$$

per ogni intero positivo n. Ricordando il Teorema cinese del resto, sappiamo che possiamo combinare le soluzioni di questa congruenza modulo interi n ed m, se questi sono primi fra loro, e quindi scegliamo come n un numero primo relativamente piccolo o una sua potenza. Dato un qualsiasi intero n chiameremo Q_n l'insieme dei quadrati perfetti modulo n, cioè $Q_n = \{x^2 \bmod n : x \in \mathbb{Z}_n\}$. Se n è un numero primo dispari allora $|Q_n| = \frac{1}{2}(n + 1)$. Il vincolo dato dalla (3) permette essenzialmente di dimezzare le classi di resto in cui x ed y possono stare.

Per fare un esempio concreto, prendiamo $N = F$ ed $n = 5$: la congruenza (3) si riduce a $x^2 \equiv 1 + y^2 \bmod 5$. Per definizione abbiamo x^2, $y^2 \in Q_5 = \{0, 1, 4\}$ qualunque siano gli interi x, y. Cerchiamo dunque interi α, $\beta \in Q_5$ tali che $\alpha \equiv 1 + \beta \bmod 5$. Facendo tutti i tentativi scopriamo che le soluzioni sono $(\alpha, \beta) = (1, 0)$, $(0, 4)$, che corrispondono alle congruenze $(x, y) \equiv (\pm 1, 0) \bmod 5$, $(x, y) \equiv (0, \pm 2) \bmod 5$.

Analogamente, sfruttando il fatto che $F \equiv 1 \bmod 4$, possiamo dire che $(x,\ y) \equiv (\pm\ 1,\ 0) \bmod 4$ o, più semplicemente, che $(x,\ y) \equiv (1,\ 0) \bmod 2$. Infine, poiché $F \equiv 1 \bmod 3$, utilizzando lo stesso principio concludiamo che $(x,\ y) \equiv (\pm\ 1,\ 0) \bmod 3$.

Le congruenze trovate si "incollano" utilizzando una semplice variante dell'Algoritmo di Euclide esteso. In conclusione, abbiamo

$$(x, y) \in \{(\pm 1, 0), (\pm 5, 12), (\pm 5, -12), (\pm 11, 0) \bmod 30\}.$$

Utilizzando le congruenze modulo tre soli numeri primi e facendo considerazioni essenzialmente elementari, abbiamo "confinato" la coppia (x, y) a solo otto possibili valori modulo 30. Tenendo conto del fatto che $x \geq s + 1 = 45031$, i primi valori di x da verificare sono 45031, 45035, 45041. Quest'ultimo valore corrisponde ad $y = 1020$ e così troviamo la fattorizzazione $F = (x - y)(x + y) = 44021 \cdot 46061$.

Nel prossimo paragrafo vedremo la formalizzazione di questa idea.

7. – Lo schema di Kraitchik

Gli algoritmi di fattorizzazione in uso negli ultimi quaranta anni si basano su un'idea del matematico Maurice Kraitchik che risale all'inizio del XX secolo. Vogliamo scomporre N in fattori più piccoli: ricordiamo che possiamo supporre che N sia un intero grande, dispari, non quadrato perfetto, composto. Invece di insistere per avere interi x ed y tali che $x^2 - y^2 = N$, cerchiamo x e y in modo che $x^2 - y^2$ sia divisibile per N, cioè cerchiamo di risolvere la congruenza

$$\text{(4)} \qquad x^2 \equiv y^2 \bmod N.$$

Se troviamo una soluzione di questa congruenza, calcoliamo

$$d = \text{mcd}(x+y,\ N)$$

che è un fattore di N. Se $1 < d < N$ abbiamo trovato un fattore non banale di N e abbiamo raggiunto il nostro obiettivo per quanto osservato qui sopra.

Illustriamo ora il motivo per cui questa idea è efficiente: ci limitiamo al caso in cui $N = pq$ con p e q numeri primi dispari e distinti. Ricordiamo che l'algoritmo di divisione per tentativi, quello di Fermat e quello di Lehman si trovano in difficoltà proprio in questa situazione.

Per prima cosa, osserviamo che l'equazione (4) ha $(2p-1)(2q-1)$ soluzioni distinte, cioè coppie (x, y) definite modulo N. Questo fatto si verifica facilmente considerando che (4) equivale al sistema

$$\begin{cases} x^2 \equiv y^2 \bmod p \\ x^2 \equiv y^2 \bmod q. \end{cases}$$

Consideriamo la prima equazione: una volta scelto y, possiamo prendere $x \equiv \pm\, y \bmod p$; queste sono due soluzioni distinte tranne nel caso in cui $y \equiv 0 \bmod p$. Dunque la prima equazione ha $2p-1$ soluzioni e la seconda ne ha $2q-1$ e il risultato segue dal Teorema cinese del resto. Dobbiamo ora contare quante di queste soluzioni portano ad una scomposizione di N: questo accade se $\text{mcd}(x+y, N) \in \{p, q\}$.

Il numero totale di soluzioni di (4) che portano effettivamente alla scomposizione di N è $2pq - p - q > pq = N$: vediamo perché. Consideriamo separatamente il valore di $\text{mcd}(y_0, N) \in \{1, p, q, N\}$.

1. Se $\text{mcd}(y_0, N) = 1$, delle quattro soluzioni x di (4) vanno bene le due con $x \equiv \pm x_0 \bmod N$, dove x_0 è la soluzione del sistema
$$\begin{cases} x_0 \equiv \ \ y_0 \bmod p \\ x_0 \equiv -y_0 \bmod q, \end{cases}$$
perché, per esempio, $p \mid (x_0 - y_0)$ e $q \nmid (x_0 - y_0)$ quindi $\text{mcd}(N, x_0 - y_0) = p$ e $\text{mcd}(N, x_0 + y_0) = q$.
2. Se $\text{mcd}(y_0, N) = p$ delle due soluzioni di (4) va bene solo quella con $x \equiv -y_0 \bmod q$. In totale, $p-1$ soluzioni. Naturalmente, se $\text{mcd}(y_0, N) > 1$ abbiamo già scomposto N in fattori e non dobbiamo fare altro.
3. Se $\text{mcd}(y_0, N) = q$ vale lo stesso, con q al posto di p.
4. Se $\text{mcd}(y_0, N) = N$ non troviamo scomposizioni interessanti di N.

In totale abbiamo quindi $2(p-1)(q-1) + (p-1) + (q-1) = 2pq - p - q$ soluzioni utili. Come già osservato, questo numero è maggiore di N; per fare un confronto, in questo caso particolare in cui N ha esattamente due fattori primi, l'equazione di Fermat $x^2 = N + y^2$ ha una sola soluzione utile.

Nei paragrafi §§8-9 vedremo due diverse realizzazioni pratiche di questo schema.

8. – Il crivello quadratico

Il crivello quadratico è un algoritmo di fattorizzazione dovuto a Carl Pomerance e prende il suo nome dal fatto che usa un polinomio di secondo grado e che il cuore dell'algoritmo contiene una variante del crivello di Eratostene. Vediamo per sommi capi come funziona, ricordando che l'obiettivo è quello di determinare almeno una soluzione dell'equazione (4).

Sia dunque N il numero da scomporre in fattori, poniamo $s = \lfloor N^{1/2} \rfloor$ e $Q(a) = (a+s)^2 - N$. Con questa scelta $Q(a) \equiv (a+s)^2 \bmod N$ qualunque sia l'intero a. Prendiamo un insieme $\mathcal{B}$ di numeri primi relativamente piccoli, con la proprietà che se $p \in \mathcal{B}$ allora l'equazione $Q(a) \equiv 0 \bmod p$ ha soluzione; poniamo $k = |\mathcal{B}|$. Possiamo supporre che $p \nmid N$ per ogni $p \in \mathcal{B}$, altrimenti abbiamo già trovato un fattore primo di N prima ancora di cominciare. La scelta della "base di fattori" $\mathcal{B}$ è delicata: da una parte è bene prendere k relativamente grande per semplificare la prima fase dell'algoritmo, ma nella seconda fase avere k troppo grande fa aumentare la complessità. Il punto di equilibrio fra queste due esigenze non è scontato.

Esaminiamo in successione i valori $Q(1)$, $Q(2)$, $Q(3)$, …, $Q(M)$ memorizzando quelli che si scompongono completamente usando *solo* i fattori primi in $\mathcal{B}$ e scartando gli altri. Non appena abbiamo trovato $k+1$ di questi valori, siamo certi che esiste una scelta di valori distinti b_1, b_2, …, $b_j \in \{1, \dots, M\}$ tale che $Q(b_1) \cdot Q(b_2) \cdots Q(b_j)$ sia un quadrato perfetto, i cui fattori primi sono tutti elementi di $\mathcal{B}$. Perché? Se $Q(a)$ si fattorizza completamente su $\mathcal{B}$ gli associamo il vettore $\vec{v}(a)$ degli esponenti con cui i numeri primi di $\mathcal{B}$, disposti in

ordine crescente, compaiono nella sua fattorizzazione. Di questo vettore prendiamo anche la riduzione modulo 2, $\vec{v}(a) \bmod 2$. Ricordiamo che questi sono vettori di $\mathbb{Z}_2^k$: se ne abbiamo $k+1$ siamo certi di avere almeno un sottoinsieme linearmente dipendente modulo 2.

Se $\vec{v}(b_1) \bmod 2$, $\vec{v}(b_2) \bmod 2$, ..., $\vec{v}(b_j) \bmod 2$ sono linearmente dipendenti, vuol dire che $\vec{v}(b_1) + \vec{v}(b_2) + \ldots + \vec{v}(b_j) \equiv \vec{0} \bmod 2$, cioè che *tutte* le componenti di $\vec{v}(b_1) + \vec{v}(b_2) + \ldots + \vec{v}(b_j)$ sono pari. In altre parole, tornando alla fattorizzazione, *tutti* gli esponenti dei numeri primi di $\mathcal{B}$ in $C = Q(b_1) \cdot Q(b_2) \cdots Q(b_j)$ sono pari, ovvero C è un quadrato perfetto, diciamo $C = D^2$. Ma, per costruzione

$$D^2 = C = Q(b_1) \cdot Q(b_2) \cdots Q(b_j) \equiv$$

$$\equiv (b_1 + s)^2 \cdot (b_2 + s)^2 \cdots (b_j + s)^2 \bmod N$$

e quindi abbiamo prodotto una soluzione di (4) con $x = D$ ed $y = (b_1 + s) \cdot (b_2 + s) \cdots (b_j + s)$. Non ci resta che calcolare $\mathrm{mcd}(N, x + y)$: se questo massimo comun divisore non è banale allora abbiamo scomposto N. In pratica, come vedremo oltre, il procedimento fornisce ben più di $k+1$ valori di a per cui $Q(a)$ si fattorizza completamente su $\mathcal{B}$ e questo dà molte diverse possibilità di scomporre N, nella remota ma non impossibile eventualità che il massimo comun divisore sia talvolta banale.

8.1 – *Un esempio concreto*

Illustriamo il procedimento per mezzo di un esempio concreto. Per motivi di spazio scegliamo un intero relativamente piccolo; alla fine faremo qualche commento su quello che succede nelle applicazioni reali di questo algoritmo.

Vogliamo fattorizzare $N = 12707 = 97 \cdot 131$; prendiamo dunque $s = 112$ e poniamo $Q(a) = (a + s)^2 - N = a^2 + 224a - 163$. Scegliamo come base di fattori l'insieme $\mathcal{B} = \{2, 7, 17, 29, 31\}$. Siccome $|\mathcal{B}| = 5$ abbiamo bisogno di avere sei valori di a per cui $Q(a)$ si scomponga completamente usando solo i fattori primi di $\mathcal{B}$. Esaminando in sequenza i valori $Q(1)$, $Q(2)$, ..., troviamo delle scomposizioni complete nei casi riportati nella tabella della Figura 3.

Abbiamo trovato sei vettori di $\mathbb{Z}_2^5$: deve esserci almeno una dipendenza lineare. Usando il metodo di eliminazione di Gauss troviamo queste dipendenze:

$$\vec{v}(2) \equiv 0 \bmod 2$$
$$\vec{v}(1) + \vec{v}(63) \equiv 0 \bmod 2$$
$$\vec{v}(1) + \vec{v}(80) + \vec{v}(1117) \equiv 0 \bmod 2$$
$$\vec{v}(1) + \vec{v}(6511) \equiv 0 \bmod 2$$

Qual è il contenuto di questa informazione? Prendiamo la seconda relazione: $\vec{v}(1) + \vec{v}(63) \equiv 0 \bmod 2$ significa che tutte le componenti di $\vec{v}(1) + \vec{v}(63) = (2, 0, 2, 0, 2)$ sono pari, come possiamo verificare immediatamente. Di conseguenza $Q(1)Q(63) = 62 \cdot 171\,918 = 1\,110\,916 = 2^2 \cdot 17^2 \cdot 31^2$ è un quadrato perfetto; poniamo $x = 2 \cdot 17 \cdot 31 = 1054$. Ricordando che $Q(1) \equiv (s+1)^2 \bmod N$ e che $Q(63) \equiv (s+63)^2 \bmod N$ e ponendo $y = (s+1)(s+63) = 113 \cdot 175 = 19775$ abbiamo trovato una soluzione non banale di (4). Non ci resta che calcolare $\mathrm{mcd}(N, x + y) = \mathrm{mcd}(12707, 20829) = 131$, che è uno dei due fattori primi di N.

In realtà $Q(2) = 289$ è un quadrato perfetto e quindi abbiamo trovato la congruenza $Q(2) = 17^2 \equiv (s+2)^2 = 114^2 \bmod N$ e questo ci dà la possibilità di scomporre N calcolando $\mathrm{mcd}(N, 17 + 114) = 131$. Si noti che questa, per un bizzarro caso, è la stessa soluzione che avremmo trovato applicando il metodo di Fermat.

a	$Q(a)$	Fattorizzazione	$\vec{v}(a)$	$\vec{v}(a) \bmod 2$
1	62	$2 \cdot 31$	(1, 0, 0, 0, 1)	(1, 0, 0, 0, 1)
2	289	17^2	(0, 0, 2, 0, 0)	(0, 0, 0, 0, 0)
63	17918	$2 \cdot 17^2 \cdot 31$	(1, 0, 2, 0, 1)	(1, 0, 0, 0, 1)
80	24157	$7^2 \cdot 17 \cdot 29$	(0, 2, 1, 1, 0)	(0, 0, 1, 1, 0)
1117	1497734	$2 \cdot 7^2 \cdot 17 \cdot 29 \cdot 31$	(1, 2, 1, 1, 1)	(1, 0, 1, 1, 1)
6511	43851422	$2 \cdot 29^4 \cdot 31$	(1, 0, 0, 4, 1)	(1, 0, 0, 0, 1)

FIGURA 3 – Le fattorizzazioni complete.

Prima di concludere, osserviamo che nelle applicazioni reali la matrice ottenuta riducendo modulo 2 i vettori corrispondenti alle fattorizzazioni è *sparsa* e ha il suo peso concentrato sulla sinistra, perché questo è il lato corrispondente ai numeri primi piccoli. Questo fatto permette un utilizzo efficiente del metodo di eliminazione di Gauss, che non dà problemi di stabilità numerica perché si lavora su $\mathbb{Z}_2$ e quindi non abbiamo realmente bisogno di calcolare inversi moltiplicativi.

8.2 – *Il crivello*

I lettori più attenti avranno notato che non ho più parlato di crivello. L'algoritmo di Pomerance, cosí come l'ho esposto finora, non è efficiente perché richiede la scomposizione in fattori primi della sequenza di interi positivi $Q(1)$, $Q(2)$, $Q(3), \ldots$, utilizzando solo gli elementi della base di fattori $\mathcal{B}$. Le idee che abbiamo già esposto permettono però di evitare gran parte dei calcoli apparentemente necessari, rendendo questo algoritmo estremamente efficiente.

Prendiamo un numero primo $p \in \mathcal{B}$; per semplicità supponiamo che p sia dispari. Per definizione, l'equazione $Q(a) \equiv 0 \bmod p$ ha soluzione. Esplicitando Q, questo significa che $(a+s)^2 \equiv N \bmod p$ ha soluzione. Poiché per ipotesi p è dispari e non divide N, questa equazione ha esattamente *due* soluzioni $a \equiv -s \pm x_p \bmod p$, dove $x_p^2 \equiv N \bmod p$. Se a non sta in una di queste due classi di equivalenza, allora $(a+s)^2 \not\equiv N \bmod p$, cioè $p \nmid Q(a)$. Nel nostro esempio qui sopra, se prendiamo $p = 7$ troviamo che $N \equiv 2 \bmod 7$ e quindi $x_p \equiv 3 \bmod 7$. Se a non è in una delle due classi $a_1 \equiv -112+3 \equiv 3 \bmod 7$, $a_2 \equiv -112 - 3 \equiv 4 \bmod 7$ allora $Q(a)$ non è divisibile per 7. Nel caso in cui $p = 2$ c'è una sola soluzione invece di due, e cioè $a \equiv s + 1 \bmod 2$.

Questa semplice osservazione permette di ridurre drasticamente il numero di operazioni da eseguire e allo stesso tempo spiega il nome dell'algoritmo. Una volta creata la lista dei valori $Q(1)$, $Q(2)$, $Q(3)$, $\ldots$, per ogni numero primo $p \in \mathcal{B}$ cerchiamo qual è l'esponente di p che divide $Q(a)$, limitandoci però ai valori di a che sono in una delle due classi a_1 e a_2 individuate sopra (una sola se $p = 2$). Come nel crivello di Eratostene, ripetiamo le operazioni relative ad un certo numero primo p solo in un numero prefissato di classi di resto modulo p. Di fatto, tenteremo di dividere per $p \in \mathcal{B}$ solo i valori $Q(a_1)$, $Q(a_1 + p)$, $Q(a_1 + 2p)$, $Q(a_1 + 3p)$, $\ldots$, $Q(a_2)$, $Q(a_2 + p)$, $Q(a_2 + 2p)$, $Q(a_2 + 3p)$, $\ldots$, sapendo *a priori* che tutti questi interi sono divisibili per p, e che *nessuno* degli altri elementi della lista lo è.

8.3 – *Osservazioni conclusive*

Il crivello quadratico di base, che abbiamo visto qui sopra, può essere migliorato in tanti aspetti. Il polinomio Q è scelto in modo che i suoi valori per $a = 1, 2, \ldots$, siano positivi e relativamente piccoli. Valutandolo anche per $a = 0, -1, -2, \ldots$ si ottengono valori piccoli in valore assoluto ma negativi.

In questo caso è sufficiente aggiungere il "numero primo" -1 alla base di fattori per ovviare al fatto che alcuni dei valori di Q trovati sono negativi. È naturalmente possibile cambiare polinomio e modificare altri dettagli per ottenere varianti più efficienti in pratica.

Un aspetto molto interessante è la possibilità di distribuire la fase di crivello su più processori: basta fornire ad ogni processore la base di fattori, le soluzioni x_p trovate sopra e un intervallo di interi a da esplorare, disgiunto da quello degli altri. Alla fine del calcolo, ciascun processore fornisce la lista dei valori di a per cui $Q(a)$ si fattorizza e la relativa scomposizione: il processore centrale raccoglie queste informazioni, crea la matrice con i vettori degli esponenti ridotti modulo 2 e procede con l'eliminazione di Gauss, la determinazione delle congruenze e il calcolo dei massimi comuni divisori richiesti.

8.4 – *La complessità computazionale*

Nella descrizione degli algoritmi visti nei paragrafi precedenti ci siamo preoccupati di indicare, almeno per sommi capi, qual è la complessità computazionale nel caso pessimo. Non abbiamo dato dimostrazioni formali per non appesantire la trattazione, ma a parte qualche dettaglio abbiamo visto l'essenziale.

Qui ci troviamo in una situazione molto diversa. Il crivello quadratico è in pratica di gran lunga più efficiente di tutti gli algoritmi descritti sopra, ma non esiste una dimostrazione rigorosa della sua

complessità computazionale che resta solo congetturata. Vediamo di capire perché.

Il cuore dell'algoritmo è la determinazione di valori del polinomio Q che siano completamente scomponibili in fattori primi appartenenti alla base di fattori $\mathcal{B}$. Abbiamo discusso del fatto che ci sono due esigenze in conflitto: se $k = |\mathcal{B}|$ è grande, allora è facile determinare questi valori perché usiamo molti numeri primi, ma naturalmente dovremo poi gestire una matrice di vettori di $\mathbb{Z}_2^k$ ed eseguire una eliminazione di Gauss. Per dare un'idea, nelle applicazioni reali un valore ragionevole per k è intorno al milione.

Per stimare il numero di valori a che dobbiamo esplorare per ottenere le almeno $k+1$ fattorizzazioni complete di cui abbiamo bisogno ci serve una buona stima per la funzione

$$\Psi(x,y) = |\{n \le x : p \mid n \Rightarrow p \le y\}|.$$

Questa funzione conta il numero di interi $n \le x$ i cui fattori primi non superano y. In realtà abbiamo bisogno di una cosa leggermente diversa, perché gli interi che consideriamo sono solo i valori di Q, ma lasciamo da parte questa ulteriore complicazione.

Non è troppo difficile dare buone stime per $\Psi(x,y)$ quando y è relativamente grande (cioè $y \approx x^\alpha$, con $\alpha \in (0,1)$) o quando y è relativamente piccolo (cioè $y \approx (\log x)^A$). Sfortunatamente, anche in questo caso ci interessa la "terra di nessuno" fra queste situazioni estreme, e quindi non è possibile, al giorno d'oggi, dare stime precise e rigorose per la complessità del crivello quadratico. Una ragionevole congettura, basata appunto su quello che si pensa sia il comportamento della funzione Ψ nella regione intermedia, afferma che la complessità computazionale del crivello quadratico è una potenza fissata della funzione

$$L_1(N) = \exp\Big(\big(\log(N)\log(\log(N))\big)^{1/2}\Big).$$

Come si vede, la complessità congetturata è di gran lunga inferiore a quella dei metodi illustrati sopra, perché $L_1(N) = \mathcal{O}(N^\varepsilon)$ per ogni $\varepsilon > 0$. Quindi il crivello quadratico rientra nella categoria degli algoritmi *subesponenziali*. Spieghiamo questa parola: la complessità di un algoritmo misura il numero di operazioni elementari da eseguire per portarlo a termine, in funzione della lunghezza dell'*input*. Un intero positivo N ha una lunghezza in *bit* pari al suo logaritmo in base 2, a parte arrotondamenti, e proporzionale al suo logaritmo naturale. Un algoritmo ha complessità esponenziale se questa è del tipo $\mathcal{O}(\exp(A\log N)) = \mathcal{O}(N^A)$ per qualche $A > 0$. Si tratta di un'esponenziale della lunghezza dell'input. Rientrano in questa categoria gli algoritmi visti nei §§3, 4, 5, per i quali, essenzialmente, $A = \frac{1}{2}$, $A = 1$, $A = \frac{1}{3}$.

Se un algoritmo ha complessità (almeno quella congetturata, come in questo caso) più piccola di $\mathcal{O}(N^\varepsilon) = \mathcal{O}(\exp(\varepsilon \log N))$ per ogni $\varepsilon > 0$, allora diciamo che ha complessità subesponenziale.

9. – Il crivello con i campi di numeri

Il crivello quadratico realizza lo schema di Kraitchik risolvendo in modo molto elegante il problema di trovare soluzioni dell'equazione (4), perché di fatto il problema di avere due quadrati perfetti è dimezzato. L'algoritmo che vediamo ora è più complicato ma più efficiente a partire da interi di un centinaio abbondante di cifre. Questo algoritmo è nato da un'idea di John Pollard, sviluppata in seguito da molti altri matematici.

Il crivello con i campi di numeri è nato dalla ricerca di algoritmi efficienti per fattorizzare interi di forma speciale, come n^2+1, o i numeri di Fermat $F_n = 2^{2^n}+1$; più in generale, sono stati sviluppati algoritmi *ad hoc* per fattorizzare numeri della forma $a^n \pm b$, con $a \ge 2$, $|b|$ ed n moderatamente grandi.

Vediamo per sommi capi come funziona la più semplice versione del crivello con i campi di numeri. Vogliamo scomporre in fattori il numero N, e cerchiamo un polinomio monico irriducibile $f \in \mathbb{Z}[x]$ ed un intero m tale che $f(m) \equiv 0 \mod N$. (Nel caso di $N = a^2+1$ prendiamo $f(x) = x^2+1$ ed $m = a$).

Prendiamo una radice, eventualmente complessa, di f che chiameremo α. Questa radice si trova nel campo dei numeri che si ottiene estendendo $\mathbb{Q}$ con α, e questo dà il nome all'algoritmo. Consideriamo l'anello $\mathbb{Z}[\alpha]$, che contiene il valore di tutti i polinomi in α a coefficienti interi (di grado $< \deg(f)$) e l'omomorfismo di anelli $\phi\colon \mathbb{Z}[\alpha] \to \mathbb{Z}/N\mathbb{Z}$ tale che

$$\phi(\alpha) = m.$$

Il nostro obiettivo è quello di determinare un insieme S di coppie di interi (a, b) relativamente primi con queste due proprietà:

1. il prodotto di tutti gli interi algebrici $a - \alpha b$, dove $(a, b) \in S$, è un quadrato perfetto in $\mathbb{Z}[\alpha]$;
2. il prodotto di tutti gli interi $a - mb$, dove $(a, b) \in S$, è un quadrato perfetto in $\mathbb{Z}$.

Chiamiamo β^2 il primo quadrato perfetto e poniamo $x = \phi(\beta)$; chiamiamo y^2 il secondo quadrato perfetto. La coppia (x, y) risolve l'equazione (4): vediamo perché. Usando ripetutamente il fatto che ϕ è un omomorfismo troviamo la catena di congruenze

$$x^2 = \phi(\beta)^2 = \phi(\beta^2) = \phi\left(\prod_{(a,b)\in S}(a - \alpha b)\right) = \prod_{(a,b)\in S}\phi(a - \alpha b)$$

$$= \prod_{(a,b)\in S}(a - mb) \equiv y^2 \bmod N.$$

La ricerca dei valori di a e b per cui sono soddisfatte le due richieste indicate qui sopra utilizza un crivello, come nell'algoritmo descritto nel paragrafo precedente. Per quanto riguarda la seconda richiesta, invece di avere un polinomio quadratico abbiamo una famiglia di polinomi lineari, ma il procedimento di base resta quello già descritto, con le opportune modifiche: una base di fattori $\mathcal{B}$ con i numeri primi "piccoli" e con il numero -1 perché $a - mb$ potrebbe essere negativo.

Vediamo come ottemperare alla prima richiesta, nel caso in cui $\mathbb{Z}[\alpha]$ è l'intero anello degli interi algebrici in $\mathbb{Q}(\alpha)$, l'anello è un dominio a fattorizzazione unica e conosciamo una base per le unità. In questo caso, possiamo procedere esattamente come sopra, utilizzando l'idea del crivello quadratico: al posto della base di fattori, avremo un insieme $\mathcal{B}_\alpha$ che contiene le unità di $\mathbb{Z}[\alpha]$ e gli ideali primi "piccoli" di $\mathbb{Z}[\alpha]$, cioè gli ideali primi di norma piccola.

Per far sí che le due cose valgano simultaneamente basta avere un'unico vettore che contenga gli esponenti relativi alla fattorizzazione in $\mathbb{Z}$ di $a - mb$ (tenendo conto del "numero primo" -1 nel caso in cui questa quantità risulti negativa) e della fattorizzazione in $\mathbb{Z}[\alpha]$ di $a - \alpha b$ (tenendo conto delle unità di $\mathbb{Z}[\alpha]$).

Questa spiegazione è molto sommaria e lascia aperti molte questioni; la complessità computazionale è determinata dal grado del polinomio f che si sceglie, oltre che dalla grandezza delle due basi di fattori che sono necessarie. In questo caso, usando argomentazioni euristiche come quelle che abbiamo citato sopra a proposito del crivello quadratico, si stima che la complessità computazionale sia una potenza di

$$L_2(N) = \exp\left(\left(\log(N)\log^2(\log(N))\right)^{1/3}\right).$$

Quindi, per N sufficientemente grande, il crivello con i campi di numeri è, si congettura, più efficiente del crivello quadratico, che risulta più efficiente per numeri "piccoli." Sperimentalmente le cose stanno proprio cosí e il "passaggio di consegne" fra i due algoritmi sembra avvenire poco sopra le 100 cifre decimali. Gli attuali record di fattorizzazione sono stati tutti ottenuti con il crivello con i campi di numeri.

10. – Un commento per concludere

In questo articolo abbiamo trattato metodi per scomporre in fattori primi numeri interi grandi "generici." Il limite attuale per interi di questo tipo è qualche centinaio di cifre decimali.

Per numeri "speciali" come quelli di Fermat $F_n = 2^{2^n} + 1$ esistono invece algoritmi *ad hoc* per la ricerca dei fattori primi che permettono di stabilire record impensabili anche solo qualche decina di anni fa. In un certo senso il crivello con i campi di numeri è nato cosí.

11. – Note e riferimenti

Il testo principale dove trovare algoritmi di fattorizzazione e in generale algoritmi sui numeri primi è Crandall & Pomerance [2]. Una selezione di questi algoritmi e le applicazioni dei numeri primi alla crittografia si possono trovare in Languasco & Zaccagnini [6]. In entrambi questi testi ci sono trattazioni dettagliate degli algoritmi ausiliari, quali il crivello di Eratostene, l'algoritmo di Euclide per il massimo comun divisore, i criteri di primalità e

pseudo-primalità (AKS, Miller-Rabin ed altri). Una divertente introduzione agli algoritmi di fattorizzazione è l'articolo di Pomerance [7], che contiene anche molte informazioni di carattere storico sullo sviluppo delle idee utilizzate nel crivello quadratico e nel crivello con i campi di numeri; il materiale esposto qui nei §§8-9 è ispirato in gran parte da questo articolo.

§1. Gauss [3].

§2. Teorema dei Numeri Primi e Teorema di Hardy-Ramanujan: Hardy & Wright [4], rispettivamente Teorema 6 e Teorema 430.

§3. Crivello di Eratostene come algoritmo: vedi Helfgott [5]. Crivello come procedimento astratto: vedi anche [8]. Quantità di memoria necessaria a conservare tabelle di numeri primi: Dialogo sopra i numeri primi, giornata quarta: [9]. AKS: [1].

§5. Tavola pitagorica: [10].

§7. Per il numero di soluzioni dell'equazione (4) vedi, in generale, l'Esercizio 5.4.3 di Languasco & Zaccagnini [6].

Ringraziamenti

Ringrazio Maria Valentino e Alessandro Gambini per aver letto una versione preliminare di questo articolo e per i loro commenti.

RIFERIMENTI BIBLIOGRAFICI

[1] M. Agrawal, N. Kayal, & N. Saxena, *PRIMES is in P*, Ann. Math. **160** (2004), 781-793.

[2] R. Crandall & C. Pomerance, *Prime numbers. A computational perspective*, Springer, New York, 2001.

[3] K. F. Gauss, *Disquisitiones Arithmeticae*, G. Fleischer, Leipzig, 1801, English translation by W. C. Waterhouse. Springer, New York, 1986.

[4] G. H. Hardy & E. M. Wright, *An Introduction to the Theory of Numbers*, sixth ed., Oxford University Press, Oxford, 2008.

[5] H. A. Helfgott, *An improved sieve of Eratosthenes*, Math. Comp. **89** (2020), 333-350, https://www.ams.org/journals/mcom/2020-89-321/S0025-5718-2019-03438-5/.

[6] A. Languasco & A. Zaccagnini, *Manuale di crittografia*, Ulrico Hoepli Editore, Milano, 2015.

[7] C. Pomerance, *A tale of two sieves*, Notices American Mathematical Society **43** (1996), 1473-1485.

[8] A. Zaccagnini, *Macchine che producono numeri primi*, Matematica, Cultura e Società **1** (2016), no. 1, 5-19, http://www.bdim.eu/item?id=RUMI_2016_1_1_1_5_0.

[9] _____, *Dialogo sui numeri primi – Un dialogo galileiano*, I librini di MaddMaths!, Roma, 2021, http://maddmaths.simai.eu/divulgazione/zaccagnini-dialogo-ebook/.

[10] _____, *La tavola pitagorica come non l'avete mai vista prima!*, Sito web MaddMaths! (2021), http://maddmaths.simai.eu/divulgazione/focus/tavola-pitagorica/.

Alessandro Zaccagnini

Alessandro Zaccagnini si occupa di Teoria Analitica dei Numeri; in particolare ha studiato problemi additivi in cui le variabili sono numeri primi, problemi diofantei in cui le variabili sono numeri primi, e problemi legati alla distribuzione dei numeri primi negli intervalli. Si è dedicato da sempre all'attività di divulgazione con articoli su riviste, su siti web e recentemente anche con un canale su YouTube.