Block Notes Matematico

TECNICHE DI PRIMALITA'

ing. Rosario Turco, prof. Maria Colonnese([1])

Sommario
Nel seguito vengono esaminati alcuni degli algoritmi più efficaci per la verifica della primalità dei numeri.

L'area dei Test di Primalità, sicuramente non banale, è utile all'indagine di vecchie congetture (primalità, gap, problemi P=NP, etc), di nuovi Teoremi e di percorsi inesplorati.

L'interesse per tale area da parte dei ricercatori è legata soprattutto a trovare test di primalità, deterministici o probabilistici, molto veloci; in questo modo si possono accumulare dati più rapidamente ed effettuare analisi su di essi per poter definitivamente verificare o migliorare le vecchie congetture come quelle di Mersenne, di Fermat, la verifica dei gap e le congetture di Cramer, Cramer-Granville, Shank ed altri.

In tale settore, come in quello della fattorizzazione, si possono trovare anche spunti e risposte per *l'altro problema del millennio P = NP*.

Attualmente, ad esempio, il test di primalità AKS, trovato da tre ricercatori indiani nel 2002, rappresenta un esempio di algoritmo deterministico con complessità polinomiale che non si poggia sulla GRH e che dimostra come i ***test di primalità facciano parte della classe di problemi in P***.

Il settore dei test di primalità coinvolge tutti i settori della *Teoria dei numeri* (curve ellittiche, etc) ed interessa la *crittografia*. Non si esclude nemmeno l'interesse per essi in ambito della *fisica, delle telecomunicazion*i etc (es. teoria delle stringhe, compressione dei dati trasmessi etc), come anche in **campo ludico** (esistono giochi basati sui numeri di Mersenne etc.), quello della **generazione dei numeri pseudo - casuali**, quello dei *problemi decisionali* (borsa, etc).

Altro settore fervente è quello che nasce grazie ai **"recordman"** che, per diletto, sono alla ricerca di scorciatoie veloci e producono Teoremi a tal fine, per trovare la primalità di numeri primi o gap giganteschi ed inventano sempre nuovi algoritmi ad hoc.

In questo articolo gli autori mostrano vari Teoremi noti, ricavandone altri e riportano anche una dimostrazione della Nuova congettura di Mersenne.

mailto:rosario_turco@virgilio.it

[1] *Rosario Turco è un ingegnere elettronico presso Telecom Italia (Napoli) ed ideatore di "Block Notes Matematico" insieme alla prof. Maria Colonnese del Liceo Classico "De Bottis" di Torre del Greco, provincia di Napoli.*

INDICE

Crivello di Eratostene

Per numeri piccoli o non elevati il crivello di Eratostene del 240 a.C. è abbastanza efficace. Qui per "numero piccolo o non elevato" si intende un numero $n \leq 10$ milioni.

L'algoritmo, però, deve scansionare tutti i numeri fino alla radice del valore d'interesse.

Metodo carta e penna. Supponiamo che n=30, la tecnica "carta e penna" è semplice:

- Scriviamo tutti i numeri da 2 a n=30
  1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
- Escludiamo l'1. Non è un primo.
- Il 2 è un primo e lo coloriamo di verde.
- Eliminiamo tutti i suoi multipli (cioè i pari) e coloriamoli di rosso.
  2 3 **4** 5 **6** 7 **8** 9 **10** 11 **12** 13 **14** 15 **16** 17 **18** 19 **20** 21 **22** 23 **24** 25 **26** 27 **28** 29 **30**
- Il primo numero dispari non colorato è il 3 che è un primo e che coloriamo di verde; ora eliminiamo i suoi multipli e sappiamo già che i multipli minori del numero di cui 3 è radice (3^2=9) sono già stati eliminati (il 6, cioè fino al quadrato di 3 già sono stati eliminati) ed eliminiamo i successivi:
  2 3 4 5 **6** 7 **8 9 10** 11 **12** 13 **14 15 16** 17 **18** 19 **20 21 22** 23 **24** 25 **26 27 28** 29 **30**
- Il primo numero non colorato è il 5 che è un primo e che coloriamo di verde; ora eliminiamo i suoi multipli e sappiamo già che i multipli minori del numero di cui 5 è radice (5^2=25) sono già stati eliminati (il 10,15,20, cioè fino al quadrato di 5 già sono stati eliminati) ed eliminiamo i successivi:
  2 3 4 5 6 7 **8 9 10** 11 **12** 13 **14 15 16** 17 **18** 19 **20 21 22** 23 **24 25 26 27 28** 29 **30**
- Il primo numero non colorato è il 7 che è un primo e che coloriamo di verde; ora eliminiamo i suoi multipli e sappiamo già che i multipli minori del numero di cui 7 è radice (7^2=49) sono già stati eliminati (il 14,21,28, cioè fino al quadrato di 7 già sono stati eliminati) ed eliminiamo i successivi:
  2 3 4 5 6 7 8 9 10 11 **12** 13 **14 15 16** 17 **18** 19 **20 21 22** 23 **24 25 26 27 28** 29 **30**
- Ripetendo il procedimento non troviamo più multipli di 11, 13 etc; per cui a questo punto quello che è rimasto tra 2 e 30 sono solo numeri primi (i verdi), anche perché già col 7, il cui quadrato era 49 si era andati ben oltre n=30:
  2 3 5 7 11 13 17 19 23 29

Algoritmo di Eratostene

In pseudo codice l'algoritmo è descrivibile di seguito.

```
Eratostene[n] {

  a[1] ← 0

  for i←2 to n do{
     a[i] ← 1
  }
  p ← 2
```

```
  while (p^2 ≤ n^2) do {

    j←p^2

    while (j ≤ n) do{
      a[j] ←0
      j←j+p
    }
    repeat p ←p+1 until a[p] = 1
  }
  return a;
}
```

L'algoritmo è abbastanza veloce per n non elevati e non richiede, per la sua elaborazione, di disporre di primi memorizzati su file system o su database.

Ovviamente la sua efficienza dipende dalla dimensione n ed essa si dimostra che è in termini di tempo O(n (log n) log log n), mentre è O(n) in termini di spazio.

Per valori di n elevati, per migliorare la tecnica di Eratostene occorre segmentare l'intervallo (Pritchard's "linear segmented wheel sieve"). In tal caso, si può ottenere una efficienza pari a O(n log n) e uno spazio pari a O(sqrt(n)/log log n).

Forma quadratica

Un miglioramento del crivello di Eratostene si può pensare di ottenere sfruttando forme quadratiche binarie.

Atkin e Bernstein hanno proposto la forma quadratica irriducibile $4x^2+y^2$ con la seguente proposizione: "Un intero positivo senza radice quadrata perfetta $p \equiv 1 \bmod 4$ è primo se e solo se vi è un numero dispari di soluzioni intere $p=4x^2+y^2$".

Crivello quadratico (Quadratic Sieve)

Fu ideato da Pomerance negli anni '80. Discende da un'idea di Fermat: è sia una tecnica di fattorizzazione dei numeri interi che un *test di non-primalità*. E' stato utilizzato con successo per il **crack del cifrario RSA a 128 bit**.

La tecnica di Fermat si basa sul prodotto notevole:

$$a^2 - b^2 = (a-b)(a+b)$$

Se passiamo nel campo Z/nZ, se vale che:

$$a^2 - b^2 = (a-b)(a+b) \text{ and } a \neq \pm b \bmod n$$

allora gcd(a-b,n)=1 è un fattore non banale di n (diverso da ± 1 e da sé stesso).

La ricerca però di coppie che soddisfano la precedente relazione è "costosa", per cui una scorciatoia è quella di cercare le congruenze modulari di sopra con numeri particolari detti "lisci", la cui proprietà è di essere costituiti da numeri primi piccoli.

Vediamo come funziona l'**algoritmo**:

- Sia n il numero da fattorizzare.
- Sia B ={$p_1, p_2, ..., p_B$} un *factor base,* ovvero un insieme di numeri primi piccoli
- Sia C un insieme di dimensione di poco superiore a B e supponiamo di aver ottenuto (con un generatore) un numero nC di congruenze del tipo:

$$x_j^{\,2} \equiv p_1^{\alpha_{1j}} p_2^{\alpha_{2j}} \ldots p_B^{\alpha_{Bj}} (\mathrm{mod}\, n) \quad \text{per } 1 \le j \le n\mathrm{C}$$

Gli xj si generano con un crivello quadratico, si selezionano cioè quei numeri per cui ogni valore α del suo quadrato appartiene a B

- Per ogni x_j si considera il vettore $\alpha_j = \{\alpha_{1j} \bmod 2, \alpha_{2j} \bmod 2, \ldots, \alpha_{Bj} \bmod 2\} \in (Z_2)^B$
  Si cerca un sottoinsieme M delle equazioni tale per cui gli aj sommati modulo 2 diano il vettore (0, ..., 0). Se questo sottoinsieme esiste allora il prodotto degli xj corrispondenti userà ogni fattore di B un numero pari di volte. Perciò è:
  $$(x_1, x_2, \ldots, x_h)^2 \equiv (p_1, p_2, \ldots, p_k)^2 (\mathrm{mod}\, n) \quad x, p \in M$$
- Siano x_{tot} il prodotto degli x_j e p_{tot} il prodotto dei primi scelti da B, allora un fattore non banale di n si calcola con $\gcd(x_{tot}-p_{tot}, n)$.

Esempio semplificato
n=15770708441
B ={2,3,5,7,11,13} factor base

Consideriamo tre congruenze generate:

$(8340934156)^2 = 3 \times 7 \bmod n$
$(12044942944)^2 = 2 \times 7 \times 13 \bmod n$
$(2773700011)^2 = 2 \times 3 \times 13 \bmod n$

Se prendiamo il prodotto di queste tre congruenze, abbiamo:

$(8340934156 \times 12044942944 \times 2773700011)^2 = (2 \times 3 \times 7 \times 13)^2 \bmod n$

Riducendo l'espressione nella parentesi modulo n, abbiamo
$(9503435785)^2 = (546)^2 \bmod n$

Quindi calcolando gcd(9503435785-546, 15770708441) troviamo il fattore 115759 di n.

Trial Division Test (TDT)

Il "Trial Division Test (TDT)" è un metodo in base al quale si divide un numero n per i numeri $p \le \sqrt{n}$ per verificare se è divisibile. In poche parole è un test di primalità ed in particolare basta che si individui solo il primo divisore di n per dire che n non è primo.

Come deve esseri i numeri per cui si verifica la divisibilità?

Per verificare se un numero è composto abbiamo almeno due opzioni:

- se stiamo in PARI/GP e disponiamo di "forprime" e del suo database di numeri primi possiamo considerare se è divisibile per almeno un numero primo (compreso il 2) fino alla $\sqrt{n}$.
- se siamo in un linguaggio (es: C/C++, Java, QBASIC etc) in cui non disponiamo di numeri primi allora possiamo verificare se il numero n è divisibile per almeno un numero dispari fino alla $\sqrt{n}$.

La radice di n è il **criterio di arresto** del TDT (altrimenti si potrebbe in teoria ciclare all'infinito e inutilmente!). In entrambi i casi se il resto è zero, allora il numero trovato è un divisore di n.

Infatti se è divisibile per un primo o per un dispari allora n è composto, ma se il numero n è divisibile solo per 1 e per sé stesso è un primo.

Un modo diverso di ridire quanto sopra è attraverso il seguente Teorema.

Teorema equivalente TDT

" Se $a \leq n \leq a^2$ allora n è primo se MCD(n,a!)=1. "

Classificazioni e Prescreening

Nel 1992 Samuel Yates definì:

- **Titanic prime** un numero con più di 1000 cifre,
- **Gigantic prime** un numero con più di 10.000 cifre,
- **Mega prime** quelli con almeno 1.000.000 di cifre.

Oggi si parla tranquillamente di primi che vanno oltre 100 milioni di cifre; ovviamente la classificazione dei primi è datata ed inutile. Si dovrebbe coniare sempre un nuovo termine perché la frontiera è sempre oltrepassata ad ogni nuova generazione di computer, il che avviene ogni 2-3 anni.

Comunque se il numero è un "numero titanico" diventa difficoltoso dividere il numero fino alla radice quadrata, ma il TDT viene usato come "*prescreening del numero*"; ovvero si prova a testare per qualche migliaio di primi (per una frazione della √n, perchè la radice potrebbe essere comunque un valore grande), poi, si passa ad un altro tipo di test.

Nell'algoritmo di esempio in Appendice non c'è il test di arresto ma è facilmente introducibile e lasciamo al lettore tale esercizio.

Wheel Factorization (WF)

Il "Wheel Factorization" consiste nel fare la scomposizione in fattori con una "regola della ruota e dei raggi":

- scomporre in fattori, dividendo subito il numero per 2,3 e 5
- dividere, poi, per i numeri dispari congrui modulo 30: 1, 7, 11, 13, 17, 19, 23, 29
- si prosegue aggiungendo numeri dispari fermandosi solo quando si incontra la radice quadrata del numero, però escludendo quelli che erano divisibili per 2, per 3 per 5.

E' chiaro che se il numero è scomponibile in fattori, allora non è primo.

Esempio devo sapere se 3331 è primo. La radice è 57,71 quindi circa 57; a questo punto si divide per 2,3,5, poi per 1, 7,11,13,17,19, 23, 29, 31, 37, 41, 43, 49, 53.

In particolare 49 non è primo ma è "relativamente primo" per la tecnica dei raggi della ruota nel WF. Il WF ha escluso i dispari 33 divisibile per 3, 35 divisibile per 5, 45 divisibile per 5, 51 divisibile per 3.

La ruota potrebbe essere scelta di varie dimensioni. Ad esempio se scegliessimo una ruota solo con il primo 2 e raggio 1, significa che dovremmo dividere per 2 poi anche i dispari. Se scegliessimo, invece, una ruota con i numeri 2 e 3 dovremmo considerare

anche i raggi 1 e 5. Dovremmo quindi dividere per 2 e 3, poi per i numeri dispari successivi. E' come se i numeri li potessimo scrivere in una tabella o matrice di larghezza 6 (perché usando solo 2 e 3 è 2*3=6 e ragioniamo modulo 6), dove escludiamo 2 e 3 e i multipli come nella figura successiva.

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36
37	38	39	40	41	42

I primi si vanno a posizionare sotto i raggi 1 e 5 e sono colorati in grigio. Ovviamente vanno eliminati quei numeri sotto i raggi che sono multipli (ad esempio 25, 35 multipli di 5).

Se usassimo nella ruota 2,3,5,7, quindi in modulo 210, dovremmo fare il check dei numeri modulo 210 congruenti con:

1, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 121, 127, 131, 137, 139, 143, 149, 151, 157, 163, 167, 169, 173, 179, 181, 187, 191, 193, 197, 199, 209

In questa ruota oltre l'1 i più piccoli raggi sono tutti primi. Questo perché se si considera un numero minore o uguale della radice quadrata del più grande numero usato nella ruota e non rimosso da essa, è primo. Questo però può portare a sviarci, perché non esistono buone ruote ma dipende dalla dimensione della lista di interi che si considera.

In ogni caso alcuni studi dimostrano che larghe ruote non sono molto efficienti ed oggi è poco usato.

Test primalità N-1

Sono test dove è facile trovare la scomposizione del numero precedente a quello dato. Appartengono a questa categoria ad esempio il Test di Lucas e il Piccolo Teorema di Fermat che vedremo nel seguito. Questa tecnica è spesso sfruttata in molti teoremi e test.

Parecchi numeri primi, tra l'altro, rispettano tale concetto:

- i numeri di Mersenne: Mp=2^p-1
- i numeri di Mersenne generalizzati o di Proth: k*(2^p) - 1
- i numeri di Fermat: 2^(2^p) + 1

In questi casi i test di primalità possono usare algoritmi polinomiali visto che non è richiesta l'intera scomposizione in fattori ma solo la certezza della primalità.

Test primalità N+1

Sono test più sofisticati dei precedenti.

Definiamo come sequenza di Lucas il sistema alle differenze del secondo ordine così descritto:

- U(m)=pU(m-1)-qU(m-2), U(0)=0, U(1)=1
- V(m)=pV(m-1)-qV(m-2), U(0)=2, U(1)=p

Tali variabili possono essere visti come i coefficienti della m-esima potenza delle radici di x^2-px+q mod n (dove p^2-4q non deve essere un quadrato modulo n).

Teorema N+1

"Sia n>1. Se per ogni fattore r di n+1 esistono p e q tali che:

- U è la variabile definita precedentemente tramite p e q
- p^2-4q non è un residuo quadratico in n
- $U(n+1) \equiv 0 \bmod n$
- $U((n+1)/r) \neq 0 \bmod n$

allora n è primo".

Proprietà vantaggiose

- U(2m) = U(m)V(m)
- $V(2m) = V^2(m) - 2q^m$

Se il numero primo da verificare è un numero di Mersenne 2^n-1 allora è possibile anche usare il test di Lucas-Lehmer che vedremo nel seguito.

Test primalità di Fermat – Piccolo Teorema di Fermat (PTF)

Il PTF dice: "Sia p un numero primo non divisibile per il numero intero a, allora

$$a^{p-1} \equiv 1 \bmod p"$$

Il Teorema fu dimostrato da Leibniz ma ciò era rimasto ignoto, poi successivamente da Eulero che lo rese pubblico.

Dimostrazione

Ipotizziamo di considerare i p-1 multipli di a:

$$a,\ 2a,\ 3a,\ \ldots,\ (p-1)\ a$$

Se moltiplichiamo il tutto è:

$$a\ 2a\ 3a \ldots (p-1)a \qquad (1)$$

Ora se r a = s a mod p allora è r = s mod p, per cui è:

$$1\ 2\ 3\ 4\ \ldots\ (p-1) = (p-1)! \bmod p$$

Per cui la (1) si può scrivere in due modi:

$$a^{p-1}\ (p-1)! = (p-1)! \bmod p$$

Dividendo per (p-1)! Si ottiene:

$$a^{p-1}=1 \bmod p$$

equivalente anche a:

$$a^{p} = a \bmod p$$

che rappresenta in pratica un

Corollario del PTF

"Sia p un numero primo ed a un intero, primo con p, allora $a^p = a \bmod p$**"**.

Il che significa lo stesso che se $a^p - a = a^{p-1}$ è multiplo di p, allora p è primo.

Esempi
Se a=2 p=7, $2^7-2=126$ che è multiplo di 7. Infatti 7 è primo.
Se a=3 p=4, $3^4-3=78$ non è multiplo di 4 perché abbiamo resto 2 e non resto 0. Per cui 4 non è primo.

Se a=3 b=1037, $3^{1037}-3$ è difficile da calcolare. Per fortuna l'aritmetica modulare ci fornisce la regola **"**Il modulo (o il resto) di una potenza è la potenza del modulo (o del resto)**"**.

Si osserva che: 1037=1024+8+4+1

Ora basta calcolare le potenze seguenti:
$3 \bmod 1037 = 3$
$3^2 \bmod 1037 = 9$
$3^4 \bmod 1037 = 9^2 = 81$
$3^8 \bmod 1037 = 6561 \bmod 1037 = 339 \bmod 1037$
$3^{16} \bmod 1037 = 339^2 = 114921 \bmod 1037 = 851 \bmod 1037$
$3^{32} \bmod 1037 = 851^2 = 724201 \bmod 1037 = 375 \bmod 1037$
$3^{64} \bmod 1037 = 375^2 \bmod 1037 = 630 \bmod 1037$
$3^{128} \bmod 1037 = 630^2 \bmod 1037 = 766 \bmod 1037$
$3^{256} \bmod 1037 = 766^2 \bmod 1037 = 851 \bmod 1037$
$3^{512} \bmod 1037 = 851^2 \bmod 1037 = 375 \bmod 1037$
$3^{1024} \bmod 1037 = 375^2 \bmod 1037 = 630 \bmod 1037$

Di conseguenza è:

$3^{1037} \bmod 1037 = 3^{1024} \bmod 1037 * 3^8 \bmod 1037 * 3^4 \bmod 1037 * 3 \bmod 1037 =$
$(630*339*81*3) \bmod 1037 = 845 \bmod 1037$

Da cui:
$3^{1037}-3 \equiv 842 \bmod 1037$.

Avvertenza: Il piccolo Teorema di Fermat va applicato su almeno un centinaio di basi a, per escludere che si incappi, come vedremo, su numeri di Carmicheal (a-PRP) o su quelli Strong (a-SPRP), che purtroppo superano il test del PTF.

Se un numero supera il piccolo Teorema di Fermat per almeno 100 basi è **"**abbastanza probabile**"** che sia primo ma si preferisce di parlare di probabile primo a-PRP; in altri

termini ci si affida alla probabilità, poiché sia i numeri di Carmicheal che quelli Strong sono poco probabili come frequenza, come vedremo.

a-PRP
Dall'esempio precedente se si calcola il resto di tale divisione è diverso da zero per cui 1037 è composto. Sarebbe meglio dire che p=1037 è un pseudo primo o un "Probable Prime base a" (**a-PRP**) con a=3, cioè 3-PRP.

Volendo dirla in altro modo il PTF è molto affidabile sui numeri che dichiara immediatamente composti, mentre se si supera il PTF siamo in presenza di un a-PRP o un a-SPRP o effettivamente di un numero primo.

Se siamo in linguaggio come PARI/GP, che dispone di numeri primi, una **sua possibile implementazione** che garantisca di non far passare i composti come a-PRP e a-SPRP, è quella di ciclare su almeno 100 basi, scelte tra 100 numeri primi (forprime) visto che gcd(a,n)=1. In questo modo si garantisce che almeno i composti non possono passare e il test è abbastanza sicuro e veloce: ovviamente la velocità dipende dal numero di cifre (digit) di cui è costituito il numero.

Se non siamo in tali tipi di linguaggi allora occorre usare tecniche di affinamento del PTF e far precedere il test da un prescreening in un range di valori che sia solo una frazione di $\sqrt{n}$.

Riassumendo il PTF ha vantaggi e svantaggi.

I vantaggi del PTF sono che:
- si può lavorare su numeri molto grandi con l'aritmetica modulare
- non c'è bisogno realmente di fare divisioni o di fare scomposizione in fattori
- si può sfruttare la regola "il modulo di una potenza è uguale alla potenza del modulo" avendo a che fare con risultati più contenuti

Lo svantaggio del PTF è che anche i "*numeri di Carmicheal*" rispettano il PTF per molte basi **ma sono composti**; poiché tali numeri sono molto rari, per evitare di incappare in uno di essi è necessario provare il test per almeno 100 basi a, così da abbattere la probabilità di incorrere in uno di essi; in tal caso la probabilità è circa 2^{100} o 10^{30}: ecco perché prima si diceva che se il test è superato è "abbastanza probabile che il numero sia primo".

Questo giustifica che se si è testato p per poche basi allora si deve parlare di a-PRP (probabile primo o probabile composto). Inoltre usando un prescreening si abbassa l'evenienza leggermente di più.

Negli anni si è fatta un po' di confusione sul termine pseudo-primo attribuendolo anche ai primi veri e propri, è preferibile chiamare i composti come numeri di Carmichael o composti a-PRP. Il termine pseudo - primo oggi si intende un probabile primo da verificare con un test di primalità preciso (esempio: Miller-Rabin, AKS, Lehmer, etc).

Qualche esempio di composto a-PRP:

composto a-PRP	fattori	a-PRP
25	5*5	5-PRP
91	7*13	3-PRP
217	7*31	2-PRP
341	11*31	5-PRP

I numeri di Carmicheal superano il PTF per tutte le basi a che sono relativamente prime a p ovvero MCD(a,p)=1([2]). Ad esempio un numero di Carmicheal è 561=3*11*17. In particolare il test a^{560} mod 561 è superato per a=2,4,5,7... ma non è superato per a=3,6,9,11,12,15,17,18...

Nei primi 25 miliardi di interi vi sono 1.091.987.405 primi, di cui 21.583 sono primi in base 2-PRP, che valse loro il titolo di "*industrial grade prime*".

Affinamenti del test PTF

Un test migliore si potrebbe ideare tenendo presente che se un numero n dispari è primo, allora il numero 1 ha due radici mod n, 1 e -1. Se n è dispari le radici di a^{n-1} e $a^{(n-1/2)}$ sono entrambe 1 e -1.

La cosa si può esprimere nel seguente modo:
"*Se n-1= 2^sd, dove d è un dispari ed s≥0, allora n è uno* ***strong probable prime base a*** *(a-SPRP) se sono valide entrambe $a^d = 1 \bmod n$ oppure $(a^d)^{2^r} = -1 \bmod n$ per alcuni r≥0 e r≤s*"

Se il test fallisce n è un "composto a-SPRP". Quelli che superano il test potrebbero essere primi.

composto a-PRP	fattori	a-SPRP
25	5*5	7-SPRP
121	11*11	3-SPRP
781	11*71	5-SPRP
2047	23*89	2-SPRP

In generale il metodo 3 volte su 4 riesce a trovare dei primi effettivi; però per avere una riuscita certa spesso si combinano più test.

Prescreening con criteri di arresto e test combinati possono però essere necessari in linguaggi che non dispongono dei numeri primi; **per cui attenzione nelle implementazioni!**

Test di Miller-Rabin– Ipotesi estesa di Riemann

Un'altra conseguenza sui numeri a-SPRP proviene dall'ipotesi estesa di Riemann (REH): " Se l'ipotesi estesa di Riemann è vera, se n è un a-SPRP per tutti gli interi a con $1 < a < 2(\log n)^2$ allora n è primo".

[2] MCD in italiano, gcd in inglese

Il test di Rabin-Miller è superato in qualche caso dai numeri Strong Probable Prime a-SPRP. Ad esempio il più piccolo a-SPRP che supera il test è: 15841=7*31*73. La probabilità che sbagli è piccola, circa 10^-60 (*Test di Montecarlo*).

Un numero p intero dispari è detto pseudoprimo forte (a-SPRP) alla base b (dove b è un intero) se, posto p sotto la forma p = 1 + 2^k * q con q dispari e k intero, una delle seguenti condizioni è soddisfatta :

a) il resto della divisione di b^q mod p = 1 oppure = p-1

altrimenti

b) si ottiene resto di valore p-1 nella divisione per p di uno dei seguenti termini : b^(2*q), b^(2^2 * q), b^(2^3* q), ..., b^(2^(k-1)* q) ([3])

Il test di Rabin-Miller consiste nel controllare se p è uno pseudoprimo forte per diverse basi, cioè per diversi numeri casuali , ciascuno dei quali è scelto in appunto in modo casuale nell'intervallo numerico da 2 a (p-1) ; se p non risulta essere uno pseudoprimo forte, in riferimento anche ad una sola delle basi scelte, il numero p è certamente composto; se invece il numero p risulta pseudoprimo forte per tutte le basi scelte, si dimostra che esso è probabilmente primo, con una probabilità non minore di 1 - 1/ 4^(nb) , dove "nb" è il numero delle basi considerate.

La strategia, in altri termini, dell'algoritmo è estendere il test dei pseudo primi del Piccolo Teorema di Fermat, perché cerca di avere informazioni non solo dal risultato finale ma anche durante le elevazioni a potenze (metodo *square and multiply*).

Test di primalità di Lucas (N-1)

Il test di primalità proposto da Lucas è abbastanza solido e valido per qualsiasi numero n. E' basato sempre sull'idea di partenza di Fermat ma con primi certi e non probabili. Il test è sicuro, solo che costringe a trovare i fattori di n-1 e se non si possono trovare è inutilizzabile.

Teorema di Lucas

"Se esiste un numero intero a tale che:
$a^{n-1} \equiv 1 \bmod n$,
$a^{(n-1/p)} \neq 1 \bmod n$, per ogni fattore primo p di n-1, allora n è primo".

Dimostrazione

Per dimostrare il teorema, basta mostrare che la funzione di Eulero $\phi(n)=n-1$ o che n-1 divide $\phi(n)$. Se non fosse così allora esisterebbe un primo p ed un esponente r tale che p^r divide n-1 ma non divide $\phi(n)$. Per questo primo p dovremmo trovare un intero a che soddisfa le condizioni di sopra. Sia adesso m l'ordine di a mod n, allora m divide n-1 (prima condizione), ma non n-1/ p (seconda condizione). Così p^r divide m che divide a sua volta $\phi(n)$, cadendo in contraddizione.

Esempio

a=3 n=257 n-1=256=2^8 : qui i fattori di 256 sono tutti 2.

3 La spiegazione sull'algoritmo è tratta brevemente dall'articolo Cristiano Teodoro, "Verifica e generazione di numeri primi relativamente grandi"- La Comunicazione Anno 2003 Volume LII – pagg.113 ÷124

$3^{256} \equiv 1 \bmod 257$
$3^{256/2} \neq 1 \bmod n$

Per cui 257 è primo.

Spesso tutti i fattori di n-1 non si possono trovare, ma possono essere sufficienti alcuni, come dimostra il teorema successivo.

Teorema di Pocklington

Sia $n-1=q^kR$ dove q è un primo che non divide R. Se vi è un intero a tale che:

$a^{n-1}=1 \bmod n$
$MCD(a^{(n-1/q)}-1,n)=1$

allora ogni primo fattore q è nella forma q^kr+1.

Dimostrazione
Sia p un qualunque primo divisore di n, mentre m sia l'ordine di a mod p. Per la prima condizione su a m divide n-1 ma non (n-1/q) (seconda condizione). Così q^k divide n-1 ed m divide p-1. Da qui segue la conclusione.

Corollario di Pocklington

Una conseguenza è il seguente **Teorema:**
"Supponiamo $n-1=F\ R$ con F>R e MCD(F,R)=1 e la fattorizzazione di F sia nota. Se per ogni fattore primo q di F vi è un intero a>1, tale che:

$a^{n-1}=1 \bmod n$
$MCD(a^{(n-1/q)}-1,n)=1$

allora n è primo"

E' da sottolineare che differenti a possono essere utilizzati per ogni primo q. Il teorema di sopra è migliorabile: "se F<R ma tutti i fattori di R sono maggiori della sqrt(R/F) o $n<2F^3$, R=rF+s con r dispari, oppure s2-4r non è un quadrato allora n è primo".

Scomposizione delle equazioni $z^n-1=0$ e $z^n+1=0$

L'equazione $z^n-1=0$ ammette ovviamente lo zero a z=1. Se si effettua la divisione (regola di Ruffini) di z^n-1 per (z-1) si ottiene:

$$z^n-1=(z-1)(1+z+z^2+z^3+...+z^{n-1})$$

Se n non è primo il secondo fattore si scompone ulteriormente.

Esempi:
$z^7-1=(z-1)(1+z+z^2+z^3+...+z^6)$
$z^4-1=(z-1)(1+z+z^2+z^3)=(z-1)(z+1)(z^2+1)$

L'equazione $z^n+1=0$ ha una analoga scomposizione solo con uno zero in z=-1.

Esempi:

$z^7+1=(z+1)(1-z+z^2-z^3+...+z^6)$
$z^4+1= z^4+1$

Tutto questo per iniziare a comprendere che ad esempio 8^5-1 è divisibile per 8-1=7.

Inoltre un numero $M(n)=2^n-1$ (***numeri di Mersenne***) ha il solo fattore banale 2-1=1 e, quindi, ha parecchie possibilità di essere primo se n è primo.

Lemma sui numeri di Mersenne

"Se n non è primo allora M(n) è composto".

Dimostrazione
Se n non è primo è prodotto di almeno due primi n = p q allora $M(n)=(2^p)^q -1$ e dalla scomposizione dell'equazione $z^n-1=0$ esiste allora una ulteriore scomposizione $M(n) = (2^p -1)$(...ulteriore fattore...) per cui M(n) non è primo.

Esempio:
$2^6-1=8^2-1=(8-1)(8+1)=7*9=63$ composto

Lemma forma 4k+3 dei numeri primi di Mersenne (R. Turco)

Tutti i numeri primi di Mersenne sono di forma 4k+3.

Dimostrazione
Supponiamo per assurdo che i numeri primi di Mersenne siano di forma 4k+1.

$$Mp = 2^p - 1 \Rightarrow Mp + 1 = 2^p$$

$$se\ Mp = 4k+1 \Rightarrow 4k+1+1 = 2^p$$

allora:

$$2(2k+1) = 2^p \ \Rightarrow 2k+1 = 2^{p-1} \Rightarrow \textit{assurdo : un dispari uguale ad un pari !}$$

$$se\ Mp = 4k+3 \Rightarrow 4k+3+1 = 2^p$$

allora:

$$4(k+1) = 2^p \ \Rightarrow k+1 = 2^{p-2} \Rightarrow k = 2^{p-2} - 1$$ il che k è dispari e ciò è possibile

Corollario (R. Turco)

Se gcd(p, gM)=1 e gcd(Mp, gM)=1 $\Rightarrow$ gcd(p,Mp)=1

Ovvero il corollario dice che poiché già sapevamo che p e Mp sono primi che è possibile anche gcd(p, gM)=1.

Corollario (R. Turco)

Se Mp è di forma 4k+3, p e pn+1 sono entrambi della stessa forma: 4n+1 oppure 4n+3.

Teorema
"Se Mp è primo allora è primo anche p", ma non è detto il viceversa.

In poche parole che p sia primo è solo una condizione necessaria ma non sufficiente affinché Mp possa essere primo.

Test primalità Lucas-Lehmer – Numeri di Mersenne

Con il test di Lehmer si dimostra quando i numeri di Mersenne $Mp=2^p-1$ sono primi per ogni p primo.

Teorema Lucas-Lehmer

"Mp=2^p-1 è primo se e solo se s(p-2) mod Mp = 0. Dove s(p-2) è l'elemento p-2 della successione esponenziale:
s(0)=4
s(p)=s(p-1)^2-2"

In altri termini Mp è primo se divisibile per il termine p-2 della successione s(p).

Proviamo a calcolare alcuni termini della successione esponenziale:
s(0)=4
s(1)=s(0)^2-2=4^2-2=14
s(2)=s(1)^2-2=14^2-2=194
s(3)=s(2)^2-2=194^2-2=37634

Ora M(5) è primo?
M(5)=2^5-1=31
s(p-2) mod Mp = s(5-2) mod M(5) = 37634 mod 31 = 0. Sì, M(5) è primo.

Già se cerchiamo s(8) scopriamo che si tratta di un numero a 147 cifre o dell'ordine di 10^{147}, mentre s(18) avrà qualcosa come 150mila cifre.

Per questo motivo è stata proposta una modifica alla successione purché si preservino le condizioni per cui valga ancora la primalità:

s(0)=4
s(p)=(s(p-1)^2-2) mod Mp

Inoltre il test si arresta automaticamente quando s(p-2)=0 perché

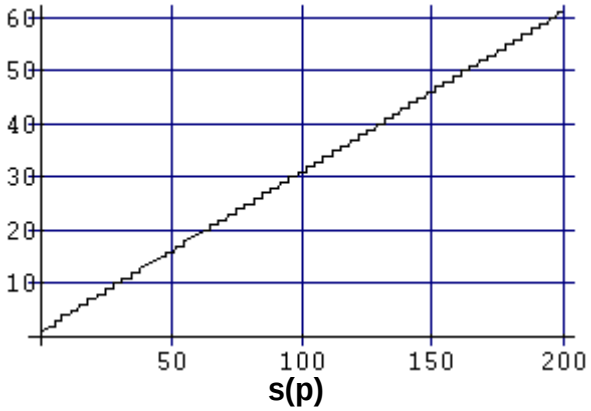
s(p-2)mod Mp=0 mod Mp = 0

Se calcoliamo i coefficienti di nuovo per vedere se M(5) è primo otterremo:
s(0)=4
s(1)=(s(0)^2-2) mod 31 = 14 mod 31
s(2)= (s(1)^2-2) mod 31= (14^2-2) mod 31 =194 mod 31=8
s(3)= (s(2)^2-2) mod 31 = (194^2-2) mod 31 = 37634 mod 31 = 0. M(5) è ancora primo.

Proprietà interessanti.
Se M(5)=31 è primo con p=5primo. Ora poiché 31 è primo anche M(31)=2147483647 sarà primo.

Infine la successione s(i), in termini di numeri di cifre non supera mai il numero di cifre di M(p), proprio per la definizione di modulo.

Se si fa un grafico dove in ordinata si riporta p e in ascissa s(p) si osserva un "andamento quasi lineare".



Per cui in tal modo si riescono a trattare numeri come M(50) con 18 cifre e superiori; mentre prima avveniva una esplosione esponenziale. Con un Pentium k6 è possibile testare M(200) in pochi secondi.

Il test di Lehmer ci svincola da fattorizzazioni, per cui è rapido. Se le potenze di 2 vengono fatte con lo shift (>>) si superano anche limiti sulle funzioni disponibili ad esempio in PARI/GP sul numero di cifre in gioco.

Lunghezza in cifre di $M(p)=2^p-1$
Possiamo valutare approssimativamente il numero di cifre di M(p) col log in base 10.

Ad esempio per p=86243 $\text{Log}_{10}(2^p-1)=25961.7$ ovvero circa 25962 cifre.

Teorema sui Numeri primi di Sophie Germain

"I numeri primi di **Sophie Germain** sono tali che:

$$S = 2p+1$$

con p e S contemporaneamente primi ed S è di forma 4n-1".

Dimostrazione
Ora se S=4n-1=2p+1 con p numero primo allora 4n=2p+2=2(p+1) e quindi 2n-1=p ovvero otteniamo un dispari che è un primo.

Teorema sui Numeri primi di Mersenne (R. Turco)

Possiamo dimostrare che:

"Se p = 4k-1 numero primo con k>0 e k≡0 mod 3 e tale che S=2p+1 ha S primo, allora Mp=2^p-1 non è primo".

Esempio
p=4*6-1=23 (k multiplo di 3)
S=2*23+1=47 primo
Mp=2^23-1 composto.

Teorema equivalente sui Numeri primi di Mersenne (R. Turco)

Lo stesso Teorema precedente si può riesprimere nel seguente modo:
"Se p = 4k-1 numero primo allora Mp è composto se sono contemporaneamente vere tre condizioni:

a. S=2p+1 è un numero primo
b. (p+1) mod 4 = 0
c. ((p+1)/4) mod 3 = 0**"**

Esempi:
p=11=4*3-1 k=3*1 Mp=2^11-1 non primo
p=23=4*6-1 k=3*2 Mp=2^23-1 non primo
p=47=4*12-1 k=3*4 Mp=2^47-1 non primo

Teorema sui numeri primi di Wagstaff ed i numeri primi di Mersenne (R. Turco)

"Se W =(4k+5)/3 è un intero dispari con k>1 e W>3, se W è un numero primo di Wagstaff con p primo e S=2p+1 non primo allora k = Mp è un numero primo di Mersenne**"**.

Dimostrazione
Un numero primo di Wagstaff è per definizione un numero primo con W = (2^p+1)/3 dove p è un numero primo.

Se p=4k-1 con k>0 e k multiplo di 3 allora W non è primo. Allora se p=4k-1, per avere W intero k non deve essere multiplo di 3 e tale che S=2p+1 non sia primo.

Ipotizziamo W numero primo. Se Mp=2^p-1 allora 3W=Mp+2.

Sappiamo che Mp=4k+3 per cui 3W=4k+5 da cui W = (4k+5)/3. Ora per essere W intero deve essere k>0. Con k=1 W=3 è primo ma k=1 non è un numero primo di Mersenne. Il successivo k, per avere W intero e numero primo, è k=7 che è un numero primo di Mersenne (per cui k>1 per avere i numeri primi di Mersenne e W>3). Per cui a questo punto per ogni numero primo W di Wagstaff abbiamo un numero primo di Mersenne Mp.

Questa tecnica non crea però tutti i numeri primi di Mersenne, difatti salta 3 e 5.

Il Teorema, di fatto, ha bisogno di 2 condizioni affinchè Mp sia primo:

- p numero primo e di un certo tipo
- W numero primo

Ovviamente i numeri primi sono sia di forma $4k\pm1$ e $4k\pm3$.

Teorema equivalente sui numeri primi di Wagstaff ed i numeri primi di Mersenne (R. Turco)

"Se p è un numero primo, che nel caso p=4k-1 abbia k>1 e non multiplo di 3 e tale che S=2p+1 abbia S non primo, se W = (2^p+1)/3 è un numero primo di Wagstaff, allora Mp è un numero primo di Mersenne**"**.

Dimostrazione Nuova Congettura di Mersenne

La congettura afferma che:

”Per ogni numero naturale dispari p, se almeno due delle seguenti affermazioni sono vere, allora lo sarà anche la terza:

- p = 2^k ± 1 o p = 4^k ± 3 per un qualche k naturale.
- 2^p − 1 è primo (Numero primo di Mersenne)
- (2^p + 1) / 3 è primo (Numero primo di Wagstaff).”

Se p è un numero dispari composto, allora, anche 2^p - 1 e (2^p+1)/ 3 lo sono. Questa è l'unica condizione necessaria per testare valori primi (test di primalità) che soddisfino la congettura.

Renaud Lifchitz ha dimostrato che la nuova congettura di Mersenne è vera fino a 12,441,900 testando sistematicamente tutti i numeri primi per cui è noto che vale almeno una delle condizioni (vedi http://www.primenumbers.net/rl/nmc/).

La “Nuova Congettura di Mersenne” è in linea con quanto visto prima.

Dimostrazione della “Nuova Congettura di Mersenne” (R. Turco)
Servono almeno due condizioni come dice l'enunciato della congettura. Dal **Teorema equivalente sui numeri primi di Wagstaff ed i numeri primi di Mersenne** abbiamo visto che serve innanzitutto che i primi p di tipo 4k±1 siano con k>1 e con k non multiplo di 3 e S=2p+1 non primo. Numeri primi p che possono essere di forma 4k±1 e rispettare tale condizione sono quelli ad esempio 2^k±1.

Ad esempio:
2^2+1=4*1+1=5
2^4+1=4*4+1=17
2^8+1=4*64+1=257

Tuttavia non esistono solo i numeri primi di forma 4k±1, anzi per coprire tutto l'insieme P dei numeri primi occorre considerare anche i numeri primi di forma 4k±3; ma analogamente i primi 4^k±3 sono di tipo 4k±3; alcuni esempi sono:

4^1+3=4*1+3=7
4^2+3=4*4+3=19

4^3+3=67

Anche dai Teoremi precedenti era che se W = (2^p+1)/ 3 è un numero primo di Wagstaff allora Mp è un numero primo di Mersenne.

La parte sfruttabile di questa congettura in ambito FPMG (Free Project Mersenne Gap - vedi sito www.gruppoeratostene.com) è che se p è primo, si può valutare la primalità di Mp, attraverso un numero più piccolo di Mp, ovvero con i numeri primi di Wagstaff! E' circa un terzo minore. Tuttavia se si esamina la lunghezza del numero binario associato al numero di Mersenne e al numero di Wagstaff si osserva solo un risparmio di 1 bit!

Esempio in PARI/GP:

lenM=length(binary(2^127-1))

127

lenW= length(binary((2^127+1)/3))

126

La fattorizzazione come pre-screening

Se la primalità per i numeri di Mersenne è difficile da verificare, spesso si cambia tattica e si tenta di verificare se esiste un piccolo fattore che divide Mp. Spesso questo è usato proprio come Test di pre-screening. Il pre-screening, qui, è inteso come prova su qualche migliaio di cicli per scartare subito un numero composto; se il pre-screening non ci si riesce allora ci si impegna su un test di primalità di un numero molto grande che può richiedere un grande tempo di elaborazione, in dipendenza della grandezza del numero stesso in input.

Teorema del "small factor"

Siano p e q numeri primi. Se q divide Mp = 2^p-1, allora è q = +/-1 (mod 8) e q = 2kp + 1 per qualche intero k.

Esempio: Mp=2^11-1 q= 2*11+1=23 → Mp/q= 89 Mp è composto

Tecnica del Top bit

L'esponente di un numero di Mersenne è una sorta di DNA. Se si vuole verificare se esiste un numero che divide 2^p-1 esiste la tecnica del "Top bit" che è legata alla verifica se l'esponente come numero primo è legato ad un numero di Sophie Germain S=2p+1.

Ad esempio per 2^23-1 vogliamo vedere se S=2p+1=2* 23+1=47 numero primo divide 2^23-1. Convertiamo l'esponente p=23 in numero in binario e otteniamo 10111. seguiamo le operazioni che facciamo in tabella da sinistra a destra come esempio:

1. convertiamo in binario p esponente di 2^p-1
2. estraiamo il primo bit a sinistra di p (il top bit o il successivo se siamo in un secondo giro)
3. facciamo il quadrato del bit estratto (questo solo all'inizio) o del modulo precedente
4. Se il bit è 1 il risultato del quadrato lo moltiplichiamo per 2 (no in caso contrario)
5. eseguiamo il modulo S del risultato del punto 4
6. se sono finiti i bit e siamo rimasti con 1, come modulo, il procedimento è terminato e il numero 2^p-1 è composto perché divisibile per S
7. se non sono finiti i bit si riprende dal punto 2

Top bit	**Quadrato**	**x 2 opzionale** (condizionata al top bit se diverso da 0)	**Mod S (S=47)**
1 0111	1*1=1	1*2=2	2
0 111	2*2=4	Rimane 4 (il bit è 0)	4
1 11	4*4=16	16*2=32	32
1 1	32*32=1024	1024*2=2048	27
1	27*27=729	729*2=1458	1

Alla fine in basso a destra della tabella rimane 1; ciò vuol dire che 2^23 = 1 mod 47 o che 2^23-1 = 0 mod 47. In altri termini 47 è un fattore di 2^23-1 che è, quindi, composto.

Fattorizzazione P-1 di Pollard

Un'altra tecnica è l'utilizzo dell'algoritmo di John Pollard, ideato nel 1974. Si basa sul Piccolo Teorema di Fermat. E' noto difatti che a^(p-1) = 1 (mod p) o che a^(p-1) - 1 = 0 (mod p)

Se p divide a^(p-1)-1 allora abbiamo un fattore.

Se N è il numero da fattorizzare ed ha un fattore p che non sappiamo, possiamo tentare di usare un gruppo di valori a^k - 1 e vedere se hanno fattori in comune con N! Dato N, il metodo consiste, quindi:

2. scegliere un numero a: 1 < a < N

3. scegliere un numero k

4. se gcd(a, N) è diverso da 1, c'è un fattore, altrimenti...

5. Si pone t = a^k mod N

6. Si pone d = gcd(t-1, N)

7. Usiamo un algoritmo di divisione per vedere se d è un fattore di N

   Se sì: abbiamo trovato un fattore

   Se no cambiamo a e/o k e torniamo allo step 4

Ovviamente dovremmo arrestarci prima o poi dopo aver deciso quanti cicli provare. Il metodo quindi è di pre-screening. Per i numeri di Mersenne specie i double Mersenne (MMp) il problema è che non si può passare direttamente in input all'algoritmo di Pollard un numero del tipo 2^(2^61-1)-1: non è rappresentabile da un normale computer.

Piccoli fattori con Q = +/- 1 mod 8 e Q primo

Un'altra tecnica è data dal **Teorema**:"Se p e q sono primi, se q divide Mp=2^p-1 allora q=+/- 1 mod 8 e q=2kp+1 per qualche k". E' equivalente a dire che "Se p è primo e per qualche k q=+/- 1 mod 8 e q=2kp+1, allora Mp è composto. Tale tecnica è sfruttabile per trovare un fattore del numero di Mersenne.

Il k in questo caso fa da acceleratore. Sono possibili anche relazioni del tipo:

q=2^kp±1 oppure q=4^kp±1 etc.

In appendice vari algoritmi di pre-screening in PARI/GP.

Progetti legati ai numeri di Mersenne

Vedi GIMPS project: http://www.mersenne.org

http://www.utm.edu/research/primes/mersenne.shtml
http://www.utm.edu/research/primes/prove/index.html

Numeri di Mersenne generalizzati
Se indichiamo i numeri di Mersenne classici con $M(p)=2^p-1$ con p primo, allora quelli generalizzati sono dati dalla formula:

$$M(p,k) = k*2^p \pm 1 \qquad (\textbf{\textit{Proth numbers}})$$

Il progetto che lavora su tali numeri è il ***progetto Proth***:

http://www.prothsearch.net/index.html

Un'altra generalizzazione è data da:

$$M(p,b) = b^p \pm 1 \qquad (\textbf{\textit{Cunningham numbers}})$$

Dove b=2,3,5,7,11 e non è multiplo di una base già usata come 4,6,8,9,10,12.

La base b=2 binaria è interessante perché riconducibile ai **numeri repunit** che hanno particolari proprietà, anche sulla primalità e ne può nascere un test di primalità.

Il ***progetto Cunningham*** è a:

http://www.cerias.purdue.edu/homes/ssw/cun
http://primes.utm.edu/top20/page.php?id=12

Interessante è anche il ***Teorema dei numeri composti di Waclaw Sierpinski*** ed i ***numeri di Sierpinski*** http://mathworld.wolfram.com/SierpinskisCompositeNumberTheorem.html

Il Teorema di Sierpinski afferma che: "Per tutti i numeri k di Sierpinski, tutti i numeri N = k · 2n + 1 sono composti per n >= 1".

In particolare il ***SoB Project*** ("Seventeen or Bust" http://www.seventeenorbust.com) *cerca di dimostrare che il numero k=78557 è l'ultimo numero di Sierpinski (trovato da John Selfridge nel 1962).*

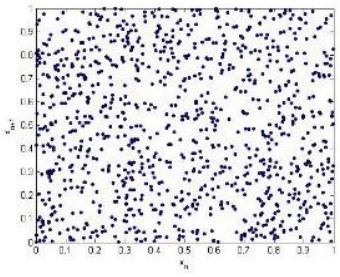
I generatori di numeri pseudo - casuali ed il Mersenne Twister

In molte applicazioni ad alte prestazioni sono importanti i simulatori di numeri casuali; ad esempio per le simulazioni di fisica e ingegneria: reattori nucleari, gasdinamica, traffico stradale, di problemi decisionali e finanziari (Borsa: prezzo di una opzione, ipotesi di Dow-Jones), in informatica (crittografia, VLSI, rendering etc), giochi di azzardo (metodo di Montecarlo), videogiochi etc.

Enrico Fermi negli anni '30 utilizzò ad esempio il Metodo di Montecarlo per la simulazione di eventi probabilistici per la diffusione casuale di neutroni.

Con degli algoritmi ad hoc si possono generare numeri pseudo - casuali (un computer deterministico che generi numeri non deterministici difatti è un assurdo che si cerca di superare con determinate tecniche: questo è il motivo che i numeri vengono detti pseudo - casuali).

Ancora oggi, in alcuni casi si usano le tecniche di Lehmer degli anni '40, creando dei generatori di numeri casuali distribuiti uniformemente LCG (Generatori Lineari Congruenziali). Una variante migliorata che comporta i numeri di Fibonacci è la LFG a cui appartiene il generatore Mersenne Twister. Nella figura successiva una distribuzione causale di punti associati a numeri primi.



Il metodo LCG, come tutti i generatori di numeri casuali, ha bisogno di un seme per inizializzare la sequenza di numeri secondo la regola $x_{n+1} = (a\, x_n + c) \bmod m$, $n \geq 0$.

Il problema della scelta dei migliori valori per a, c ed m per ottenere che la successione abbia il massimo periodo è quindi il punto cruciale del metodo. Una delle scelte più popolari è $m=2^{31}-1$, $a=75$, $c=0$. Questo garantisce un periodo di oltre due miliardi di numeri casuali.

Il fatto che m sia un numero primo molto particolare (un numero primo di Mersenne, ossia esprimibile come 2^n-1 con n intero positivo primo), è fondamentale per ottenere il massimo periodo.

I generatori LCG però non sono adatti a diversi test statistici, ma nella sua semplicità evidenzia un' aspetto importante nella generazione di numeri pseudo - casuali, ossia la necessità di avere a disposizione moltissimi numeri casuali ed in modo rapido.

Successivamente, sono stati proposti i "generatori di Fibonacci ritardati" o LFG, per la similarità con la sequenza di Fibonacci $x_{n+1}=x_n + x_{n-1}$, ideati negli anni '50 da G.J. Mitchell e D.P. Moore, e che arrivavano a produrre oltre 38 milioni di miliardi di miliardi di numeri pseudo - casuali!

Nonostante questo, anche i generatori LFG risultano insoddisfacenti in alcune applicazioni e il loro comportamento dipende molto dall'inizializzazione del metodo.

Ad oggi il miglior generatore disponibile è il Mersenne Twister, o MT in breve, una variante dei metodi LFG presentata nel 1997 da M. Matsumoto and T. Nishimura. Attualmente l'algoritmo MT, implementato in molti linguaggi di programmazione, garantisce un periodo colossale, pari al numero di Mersenne 219937-1, da cui il nome del metodo. Come termine di paragone basti pensare che al numero di atomi, espresso in fattoriale, che compongono il nostro universo:1080!

Numeri di Fermat

I numeri di Fermat sono dati dalla formula:

$$F(n)= 2^{2^{\wedge}n}+1$$

E' facilmente calcolabile che sono primi da n=0,...,4 poi diventano rari e a valori di n molto grandi. Anche la loro dimostrazione sull'infinità è in "cerca di autore".

$f(0) = 2^{2^{\wedge}0} + 1 = 2^1 + 1 = 3$	prime
$f(1) = 2^{2^{\wedge}1} + 1 = 2^2 + 1 = 5$	prime
$f(2) = 2^{2^{\wedge}2} + 1 = 2^4 + 1 = 17$	prime
$f(3) = 2^{2^{\wedge}3} + 1 = 2^8 + 1 = 257$	prime
$f(4) = 2^{2^{\wedge}4} + 1 = 2^{16} + 1 = 65{,}537$	prime
$f(5) = 2^{2^{\wedge}5} + 1 = 2^{32} + 1 = 4{,}294{,}967{,}297$	NOT prime!
$f(6) = 2^{2^{\wedge}6} + 1 = 2^{64} + 1 = 18{,}446{,}744{,}073{,}709{,}551{,}617$	NOT prime!
$f(7) = 2^{2^{\wedge}7} + 1 = 2^{128} + 1 =$	NOT prime!

Come può essere nata l'idea? Non lo sappiamo con certezza ma possiamo ipotizzare, da quello visto precedentemente sulla scomposizione delle equazioni $z^n-1=0$ e $z^n+1=0$, che se i numeri di Mersenne producevano primi (esclusi i composti) con 2^n-1, allora andava indagato se si poteva produrre numeri primi anche con la formula 2^n+1.

Teorema dei primi di Fermat

Condizione necessaria ma non sufficiente affinché 2^n+1, con n=2^m, sia un numero primo è che n sia un numero pari.

Dimostrazione

Da una ispezione dei numeri di Fermat di sopra è evidente che è maggiore la probabilità che la formula dà un primo se l'esponente n di 2^n+1 è pari, ovvero solo nei casi $n=2^m$ si possono produrre dei numeri primi. Difatti se $n=k*2^m$ allora k è dispari ed il numero è un composto (perché k contiene tutta la scomposizione dei primi dispari di n); difatti:

$$2^n+1=2^{k*2m}+1=(2^{2m})^k+1=(2^{2m}+1)(\text{...altro fattore...})$$ quindi composto

Numeri di Fermat e figure con riga e compasso

Questa dimostrazione dei numeri di Fermat è importante anche per la costruzione di figure con riga e compasso e risale a *Gauss*.

Lemma Figure con riga e compasso

Se p è numero primo, i p-agoni costruibili con riga e compasso sono quelli per cui $p-1=2^n$ o in altri termini $p = 2^n+1$ ed n è una potenza del 2.

Con i numeri di Fermat si è visto proprio che p è un numero primo se n=2^m.

Gauss in questo modo dimostrò che sono costruibili p-agoni con p=3, p=5, p=17... ma anche con prodotti di primi di Fermat (p=3*5=15) e i raddoppiamenti di essi (p=6, 10, 30, ...).

Lemma dei p-agoni

Un p-agono è costruibile se scomponendo n in fattori primi, allora gli unici fattori primi di esso sono il 2 o i numeri primi di Fermat (presi anche più volte).

Link interessanti sui numeri di Fermat sono:
http://www.fermatsearch.org/
http://www.fermatsearch.org/history/cosgrave record.htm/

Test di Pepin

Un classico caso di applicazione del teorema di Pocklington e del teorema conseguente è il test di Pepin, usato con i numeri di Fermat.

Sia F(n) l'n-esimo numero di Fermat ($F(n)= 2^{2^n}+1$) con n>1, F(n) è primo se e solo se $3^{(Fn-1)/2} = -1 \bmod F(n)$.

Dimostrazione

Se $3^{(Fn-1)/2} = -1 \bmod Fn$ allora Fn è primo dal teorema conseguenza di quello di Pocklington con a=3. Se Fn è primo allora $3^{(Fn-1)/2} = (3|Fn) \bmod Fn$ e (3|Fn) = -1.

Il simbolo (3|Fn) è la simbologia di Jacobi.

Teorema di Proth

"Sia $n=h2^k+1$ con $2^k>h$, se vi è un intero a tale che $a^{(n-1)/2} = -1 \bmod n$ allora n è primo"

Congettura di Catalan

Data una sequenza:
C0=2, C1 = 2^{C0}-1, C2=2^{C1}-1, C3=2^{C2}-1, C4=2^{C3}-1, allora C4 è primo".

Si può dimostrare che la sequenza dà sempre primi?
http://www.utm.edu/research/primes/mersenne.shtm

Test di Solovay-Strassen

E' un test probabilistico per individuare i composti.

In pseudo-codice è:

```
Solovay-Strassen(n,s)
  For i=1,s do
     If( Test_SS(n) ) return composto
  Return primo

Test_SS(n)
  A:= random (2, n-2);
  if( gcd(a,n) > 1) return true;
    if( (a/n) ≠ a^((n-1)/2) mod n) return true
  else return false
```

Sopra (a/n) è il *simbolo di Jacobi*. Se n è primo allora il *simbolo di Jacobi e di Legendre* coincidono.

Test APR

Fu introdotto nel 1983 da Adleman (la A degli autori del RSA), Pomerance e Rumley. E' il più veloce test tra quelli non polinomiali. E' molto complesso ed anche la sua implementazione è molto onerosa.

Si basa su tre elementi fondamentali:

- Teoria dei campi ciclotomici
- I numeri primoriali
- Il Teorema del Resto Cinese (CRT)

Un *campo ciclotomico* è una estensione dei numeri razionali generata da una radice dell'unità, ovvero generata da un numero che elevato ad n dà 1.

I *numeri primoriali* sono indicati con n# (un po' come i fattoriali indicati con n!) e rappresentano il prodotto di tutti i numeri primi minori o uguali a n.

Ad esempio 7# = 2*3*5*7=210

Il *CRT* dice che:
"Se $n_1,n_2,...,n_k$ sono primi tra loro allora il sistema delle congruenze modulo:
$x = a_1 \bmod n_1$
$x = a_2 \bmod n_2$
.....
$x = a_k \bmod n_k$

ha una soluzione unica modulo $n=n_1n_2...n_k$".

Primalità AKS

AKS deriva dai nomi degli autori indiani Agrawal, Krawal e Saxena. Nato nell'agosto del 2002 e discende da una generalizzazione del Piccolo Teorema di Fermat.

E' di grosso interesse tra i ricercatori perché è il primo algoritmo di **test di primalità deterministico con complessità polinomiale e senza appoggiarsi alla GRH**: esso dimostra innanzitutto che **la classe dei problemi di primalità sta in P**. Non ha applicazione nella crittografia a causa del suo alto grado del polinomio in gioco (12 o 18 con congetture da verificare che lo porterebbero a 6).

AKS estende banalmente il PTF visto prima: "un intero è primo se e solo se vale, per un positivo a coprimo a p: $(x + a)^p = x^p + a \bmod p$"

La dimostrazione di correttezza sfrutta sia elementi della Teoria dei numeri che dei campi finiti.

Teorema sottostante all'AKS

"Sia p un intero dispari ed r un numero primo.

Se:
(1) p non è divisibile per nessuno dei primi fino ad r;
(2) L'ordine di p mod r è almeno $\log^2 p/\log^2 2$;
(3) per ogni $a < r$ si ha $(x + a)^p = x^p + a$ nell'anello $Z[x]/ p$,

allora p è primo"

Algoritmo AKS

L'**algoritmo** funziona nel seguente modo:

- Sia p l'intero dispari da testare. Si verifica innanzitutto che p non sia una potenza di un qualche numero primo.
- Si determina il più piccolo numero primo r che non divida né p né alcuno dei numeri pi-1, per ogni $i < \log^2 p/\log^2 2$. Se viene trovato un r divisore di p ovviamente l'algoritmo termina.
- Per ogni intero $a < r-1$ si verifica che valga $(x + a)^p = x^p + a$ nell'anello $Z[x]/ p$. Se ciò non accade l'algoritmo termina.
- Se l'algoritmo non è terminato precedentemente, p è primo.

Comunque la trattazione dei dettagli e la dimostrazione è molto lunga e non verrà qui mostrata. Alcuni buoni riferimenti sull'AKS sono [1] e [2].

Test Goldwasser-Kilian

Fu ideato nel 1986. Per il test di primalità sfrutta le curve ellittiche (vedi [3]) in un campo finito.

Si consideri una curva ellittica En(A, B): $Y^2 = X^3 + AX + B$ definita su Z/nZ, con discriminante $\Delta = 4A^3 + 27B^2 \neq 0$ e n non divisibile per 2 o 3.

Sussiste il seguente **Teorema**:

"Sia En(A, B) una curva ellittica definita su Z/nZ, n primo con 2, 3 e 4A3+27B2, se esistono:
• un punto M che appartiene a En(A, B), M ≠ I;
• un intero q, primo e strettamente maggiore di n1/ 2+1+2n1/ 4, tale che qM=I;

allora n è primo".

Vediamo l'algoritmo da applicare:

- Sia p il numero da testare.
- Si seleziona una curva ellittica su Z/pZ, con parametri A e B casuali e discriminante non nullo.
- Sia poi Np l'ordine del gruppo ellittico Ep(A, B) trovato. Finché non si trova un gruppo con ordine di tipo 2q e con q probabile primo va ripetuta questa fase.
- Si verificano le condizioni del teorema sopra esposto, supponendo primo il q ricavato precedentemente. Se queste sussistono allora la primalità di p è certificata a condizione che q sia primo.
- Si ripete ricorsivamente l'algoritmo ponendo p=q, finché non si è certi che l'ultimo q trovato sia primo.

Considerazioni sugli algoritmi

Alcuni algoritmi di primalità notoriamente sono classificati a ragione tra i più veloci. In Appendice si fa menzione a due test probabilistici: Piccolo Teorema di Fermat e Miller-Rabin (comparabili in termini di velocità) e a tre deterministici: TDT, Lucas-Lehmer e AKS (l'AKS migliore in termine di velocità).

In Appendice vengono mostrati però solo il Piccolo Teorema di Fermat, TDT e Lucas-Lehmer, questo perché per AKS e Miller-Rabin servirebbe una trattazione a parte e abbastanza lunga.

Per chi è per natura un "softwar-ista smanettone a caccia di record" e dispone solo di hardware semplice come un PC, allora algoritmi come quello del Piccolo Teorema di Fermat, di Rabin-Miller, Lehmer e AKS sono il "coltellino svizzero" dei "recordmen"([4]).

Alcuni trucchi o scorciatoie (vedi numeri di Wagstaff oppure regole come "il modulo di una potenza è la potenza del modulo" e altre ancora) sono ideate a fronte di teoremi e congetture valide, anche se non dimostrate (vedi Nuova congettura di Mersenne), della Teoria dei Numeri; altri trucchi sono ideati in base alla tecnologia e il linguaggio a disposizione (shift bit a bit, utilizzo dei numeri primi se disponibili, etc) che possono consentire di superare i limiti di elaborazione normale e di migliorare la velocità anche di alcune funzionalità.

In altri casi, come il Test di Lucas-Lehmer, non danno problemi tecnici ma a prezzo di un maggiore tempo di elaborazione.

[4] Come il nostro amico e stimato professore Di Maria Giovanni (vedi www.gruppoeratostene.com).

Tuttavia essi consentono di ottenere la soddisfazione di arrivare a qualche numero di quelli titanici e dimostrare che è primo, ma soprattutto di aver scritto il software con le proprie mani e comprendere le varie problematiche in gioco.

Riferimenti

[1] Test di Primalità Probabilistici e Deterministici - Tesi di Laurea di Giancarlo D'Urso - 2006 - 2007 Università degli studi di Catania
[2] An implementation of the AKS Primality Test - Robert G. Salembier and Paul Southeringhton member IEEE
[3] Congettura di Birch e Swinnerton-Dyer - Curve ellittiche - Fattorizzazione discreta - Crittografia - R. Turco, M. Colonnese

Appendice software

I sorgenti in PARI/GP sono compilabili anche con gp2c.

```
/*
/*
* Trial Division Test (TDT)
*
* It is used as Prescreening
* on 30000 primes
*
*/
TDT(q, c=30000)= local(status=1,i=0); {

 status = 1;

 print("  ");
 print("Prescreening TDT ...");
 print("  ");

 if( lift(Mod(q,2)) == 0 & q != 2,
        return(0);
 );


 forprime(i=3,c,
    if( lift(Mod(q,i)) == 0 & q != i,
       status = 0;
       break;
    );
 );

 return(status);

}

/*
*  Mersenne Primality
*
*/

Mers(q,alg=1) = local(status=0); {

 /* alg=1 is default */

 read("CONFIG.TXT");
 read("MILLER.TXT");
 read("AKS.TXT");


 print(" ");
 print("p \t\t= ", q);

 /* M=2^p-1;   This has overflow problems on power then we use shift */

 M = 1;
```

```
 M = shift(M, q) - 1;

 Wrid = (M + 2) / 3;
 /* We use New Conjecture of Mersenne */
 /* if p is prime and (2^p + 1)/3 is prime then Mp is prime */

 if( alg == 1, print("Miller-Rabin's test ..."); status = MillerRabin(Wrid););
 if( alg == 2, status = LFT(Wrid););   /* Little Fermat's Theorem */
 if( alg == 3, print(" "); print("test AKS ..."); status = AKS(Wrid););
 if( alg == 4, status = MersLehmer(M););

 if( status == 1,
     write(myfile, " p=", q , " ", "Mp=2^p-1=", M, " Wagstaff=", Wrid);
     digit=ceil(logb(M,10));
     write1(myfile, " digit = ", digit;);
     print("Mp=2^p-1 \t= ", M, " Prime!");
     print(" ");
     print("Wagstaff's prime = ", Wrid);
     print(" ");
     print("digit = \t", digit;);
 );

 if( status == 0,
     print("Not Prime!");
     M=0;
 );

 return(M);

}

/*
* Lehmer's test for Mersenne's numbers
*
* MersLehmer(2^46663-1) time= 3mn 19,422 ms
*
*
*/

MersLehmer(Mp) = {

 print("Lehmer's test ...");

 pmin2=log2(Mp+1) - 2;

 s=4;   /* s(0)=4  and  s(i) mod Mp = [s(i-1)^2-2]mod Mp
           Mp is prime iff s(p-2)mod Mp = 0 */

 for(i=1,pmin2,
      appo = lift(Mod(s, Mp)^2);
      s = appo - 2 ;
     );


 if( lift(Mod(s,Mp)) == 0, return(1););

 return(0);

}

/*
*
*  Little Fermat's Theorem
*
*/

LFT(n,nbasi=100,logprint=0) = local(M=0, i=0, ret=1); {

 if( n <=1, error("You must insert a positive number > 1 ..."));

 print(" ");
 print("Little Fermat's Theorem ...");
 print("Test on a number of base : ", nbasi);

 ret=1;
```

```
 /* We use prime number, a stronger condition than gcd(base, n) == 1
    but this excludes Carmichael numbers (a-PR) and a-SPR */

 forprime(i=1,nbasi,

   if( i < n,

      /* We use Mod(i,n)^(n-1) instead of Mod(i^(n-1),n) for the speed */
       if( lift(Mod(i, n)^(n-1)) != 1,

         ret=0;


         if( logprint == 1,
            print("Not prime for base : ", i);
         );

         break;

       );

    );

  );

 return(ret);

}
```

Small factor

```
/*
* TestMM
* p is the exponent in MMp
* for example if 2^(2^7-1)-1 then p=2^7-1
*
*/


TestMM(p) = local(status=1, r=1, A=0, q=0);{
printp("Test TMM \t…");

 A = 100000000;

 for(k=1,A,
    q = 2*k*p+1;
    if(ispseudoprime(q,1),

       r=Mod(2,q)^(p%eulerphi(q))-1;

       if(r==0,
          print(r," -> small factor = ",q);
          printp("Not Prime!");
          status=0;
          k=A;
        );
     );
 );

 if( status == 1, printp("small factor there isn't in intervall 3 .. ", A););

 return(status);

}
```

Top bit

```
TM(p) = local(status=1, i=1, len=0, S=0);{

 printp("Test TM \t…");
 S=2*p+1;
 len = length(binary(p));
 B = Vecsmall(binary(p));

 q = B[i]*B[i];
```

```
 while( i<=len & status ==1,

        if( B[i] != 0,
            q = q*2;
        );

       r = q%S;
        q = r*r;

        if( i == len & r == 1,
            status = 0;
            printp("Not Prime!");
        );
        i++;
 );
 return(status);
}
```

Pollard

```
/*
*
* P-1 Factorization (Pollard)
* N numero da fattorizzare
* B numero della base dei fattori (i primi B primi)
* s=1 è opzionale ed indica il numero di base a scelta casuale
*
*/

 M_P(n,B,s=1)= local(m, a, p, j, d, g, l, i, lgn, status=1);{

    if(n <= 1 | !bittest(n,0) | s<= 0, error("Input non valido"); return);
    l= floor(lgn/log(2));
    for(i=1,s,
      until(a, a=random()%n);    /* genero a finche' a non divide n */
      /* print("base scelta numero ", i, ": a = ",a); */
      g=gcd(a,n);
       if(g > 1, print(g," e` un fattore di ",n); status=0; return(status););
       p = 3;
       j = 1;
       d = 1;
       lgn = log(n);
      while(d == 1 && j <= B,
              a = lift(Mod(a,n)^(p^floor(lgn/log(p))));
             d = gcd(a-1, n);
              if(d > 1 && d < n , print(d," e` un fattore di ",n); status=0; return(status););
               j = j++;
             p = nextprime(p++);
           );
       m = 1;
             while(d == 1 && m <= l,
                                      a = lift(Mod(a,n)^2);
                                                d = gcd(a-1,n);
                                                if(d > 1 && d < n, print(d," e` un fattore di ",n); status=0; return(status););
                                                m = m++;
                                       );
              );
    return(status);
}
```