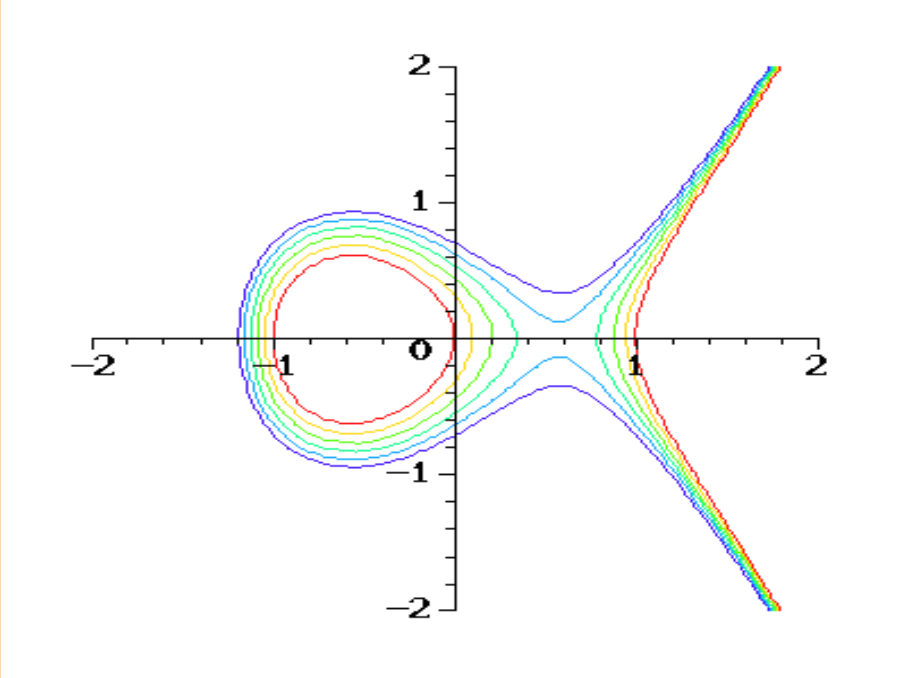


NUMERI E CRITTOGRAFIA

Carlo Toffalori (Camerino)

13 novembre 2014

ITIS Divini, San Severino Marche

Edgar Allan Poe, *Lo scarabeo d'oro*: la caccia al tesoro degli antichi pirati!

Istruzioni incomprensibili…

53‡‡†305))6*;**48**26)4‡.)4‡);806*;**48**†8Ч60))85;1‡
(;:‡*8†83(88)5*†;46(;88*96*?;8)*‡(;**48**5);5*†2 …

Il protagonista (William Legrand):

- 8 *come* e
- ;48 *come* the

e via dicendo.

Lo schema generale

- Due personaggi: **A** e **B** (due degli antichi pirati, oggi ***Alice*** e ***Bob***)
- Il "cattivo" (non necessariamente cattivo): **E** (William Legrand, oggi *Eve*, "the eavesdropper")

L'obiettivo di **A** e **B**: trasmettersi informazioni senza che **E** le capisca; le operazioni sono allora due

- Cifrare
- Decifrare

con una chiave opportuna. In questo consiste la **Crittografia.**

L'obiettivo di **E**: violare il sistema usato da **A** e **B**; questa è la **crittoanalisi**.

Un linguaggio “universale”: numeri invece che lettere

A	0	N	13
B	1	O	14
C	2	P	15
D	3	Q	16
E	4	R	17
F	5	S	18
G	6	T	19
H	7	U	20
I	8	V	21
J	9	W	22
K	10	X	23
L	11	Y	24
M	12	Z	25

Procedimento usuale di codifica e decodifica

Permutare numeri o lettere.

Esempio (Giulio Cesare)
Si sostituisce ogni lettera con quella che la segue di 3 passi:
$A \to D,\ B \to E,\ C \to F, \ldots, W \to Z, X \to A, Y \to B, Z \to C$

Matematicamente parlando…

$0 \rightarrow 3,\ 1 \rightarrow 4,\ 2 \rightarrow 5, \ldots, 22 \rightarrow 25,\ 23 \rightarrow 0,\ 24 \rightarrow 1,\ 25 \rightarrow 2$

Dunque

- si codifica tramite $x \rightarrow x+3$ (ma *attenzione a 23, 24 e 25*)
- si decodifica tramite $x \rightarrow x - 3$ (ma *attenzione a 0, 1 e 2*)

Le chiavi ± 3

Essenziale: chiave di codifica / decodifica

- In genere la stessa
- Comunque tra loro direttamente collegate e computazionalmente equivalenti.

Una strana aritmetica:

- arrivati a 26, si ritorna a 0

Come nelle ore del giorno (arrivati a 24 si torna a 0): *l'aritmetica dell'orologio*

- modulo 26,
- modulo 24,
- modulo *m* per ogni intero positivo *m*.

Procedimento usuale di crittoanalisi

Analisi di frequenza (Ishaq al-Kindi): si basa sul confronto tra

- le lettere più comuni nell'alfabeto (*e*, *the*, …)
- le lettere più frequenti nel messaggio (8, ;48, …)

Un altro esempio (Vigenère)

Il crittosistema di Cesare è troppo rigido e si viola facilmente con la crittoanalisi di frequenza.

Idea (Alberti, Vigenère): si fa dipendere la codifica

- non solo dal simbolo da sostituire,
- ma anche dalla sua posizione nel messaggio.

In dettaglio:

- fissiamo una stringa finita di numeri (come 2, 4, 7)
- un messaggio si codifica sommando il primo simbolo per 2, il secondo per 4, il terzo per 7, il quarto per 2, il quinto per 4, il sesto per 7, il settimo per 2, e via dicendo; si decodifica con le corrispondenti sottrazioni.

Messaggio in chiaro:	7	8	22	**24**	11	12	19	**24**
Chiave:	2	4	7	**2**	4	7	2	**4**
Messaggio cifrato:	9	12	3	**0**	15	19	21	**2**

Ancora vulnerabile grazie a tecniche di crittoanalisi di frequenza.

Ci sono cifrari sicuri ?

Esempio: Cifrario di Vernam, One-Time-Pad (taccuino monouso), 1917. Porta l'idea di Vigenère alle estreme conseguenze:

- la chiave è tanto lunga quanto il messaggio;

Inoltre

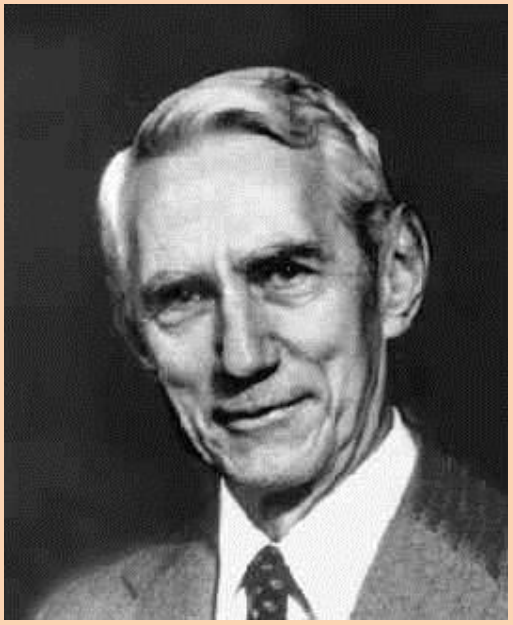
- l’informazione e la chiave sono sequenze di 0 ed 1
- la codifica e la decodifica avvengono sommando con la chiave modulo 2

Messaggio in chiaro	1	0	0	0	1	0	1	0	1	0	0	1	1	1	0
Chiave	1	1	0	1	1	1	1	1	0	1	1	0	0	1	0
Messaggio cifrato	0	1	0	1	0	1	0	1	1	1	1	1	1	0	0

Svantaggi:

- generazione della chiave (sequenza casuale di 0 ed 1)
- trasmissione della chiave

Una chiave per ogni messaggio (“taccuino monouso”): troppo costoso!
Eppure…



La crittografia secondo Shannon

Crittosistema

- un insieme di messaggi
- un insieme di chiavi
- per ogni chiave *k* in K, due procedure, l'una inversa dell'altra, e_k, d_k che producono i messaggi cifrati da quelli in chiaro e recuperano quelli in chiaro da quelli cifrati.

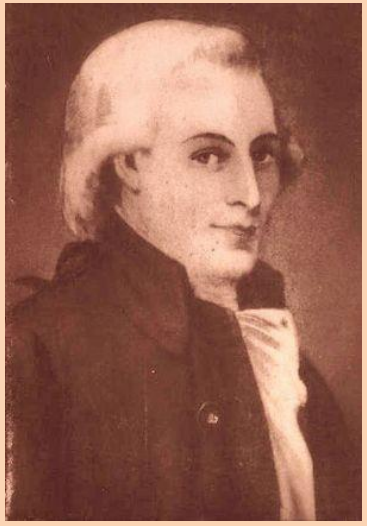
Il crittosistema è *perfetto* se la conoscenza del messaggio cifrato non ci dà nessun vantaggio per la conoscenza del messaggio in chiaro.

Un crittosistema perfetto richiede almeno *tante chiavi quanti messaggi*.

In altre parole: i cifrari alla Vernam sono "gli unici" perfetti.

Crittografia classica

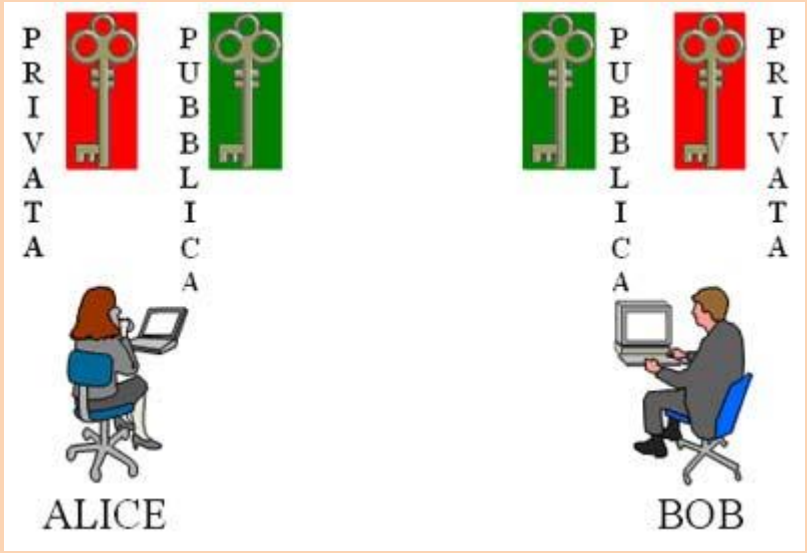
1. pochi utenti
2. una chiave di codifica/decodifica preventivamente concordata e scambiata
3. la codifica è computazionalmente equivalente alla decodifica (Casanova e M.me d'Urfé)



Crittografia moderna

- Esigenze: trasmissione in rete, voto telematico, ...
- Caratteristiche: un gran numero di utenti **A**, **B**, **E**, ...

Non più buoni o cattivi, tutti sono chiamati a partecipare.
Ma… *il segreto di tutti è il segreto di nessuno!*



Come rimediare

a) Non più una sola chiave per codifica e decodifica **ma**, per ogni utente A,
    - una chiave pubblica di codifica (disponibile a chiunque voglia scrivere ad A)
    - una chiave privata di decodifica (conosciuta **solo** da A)

b) Decodificare deve essere enormemente più difficile che codificare (salvo che per A)

Un esempio pratico: il *doppio lucchetto* (Diffie-Hellman 1976)

- B invia un messaggio ad A in una scatola chiusa con la sua chiave (il primo lucchetto)
- A chiude ulteriormente con la sua chiave (i secondo lucchetto) e rispedisce a B
- B apre il suo lucchetto e rispedisce ad A
- A apre il suo lucchetto e la scatola

In teoria: le funzioni a senso unico

- facili da calcolare
- invertibili
- con inversa difficile da calcolare

L'idea

- f codifica
- f^{-1} decodifica

Osservazione

La nozione di funzione a senso unico è

- non rigorosa (che significa: facile/difficile da calcolare?)
- da controllare periodicamente (quello che è difficile oggi può diventare facile domani)

Problema: Dove cercare funzioni a senso unico?

Una parentesi computazionale

Il ruolo del *calcolatore* nella vita di oggi

Illustri precursori
Leibniz, *Dissertatio de arte combinatoria*, 1666: "*Calculemus!*" (Calcoliamo!)

"*Io chiamo calcolo qualunque notazione che rappresenti il ragionamento, quand'anche non avesse alcun rapporto con i numeri*"

Modelli matematici anche per situazioni non matematiche
"*dai ragionamenti complicati ai calcoli semplici, dai vocaboli di significato vago ed incerto a caratteri determinati*"

Servono

- "*lingua characteristica*": un linguaggio scientifico universale
- "*calculus ratiocinator*": un calcolo della ragione

(si anticipano Intelligenza artificiale e Deduzione automatica)

Dubbio: "Tutto" si può calcolare?

Motivazioni: difficoltà a risolvere certi problemi matematici (ricordare i ***Teoremi di Incompletezza di Gödel***)

Una soluzione positiva: basta produrre un algoritmo che funziona

Una soluzione negativa: bisogna escludere ogni possibile procedimento

Da chiarire preliminarmente: che cosa è un algoritmo?

Meglio: quali sono i problemi che hanno un algoritmo di soluzione (calcolo)?

Turing, Church, Kleene, Godel, ...(1936)

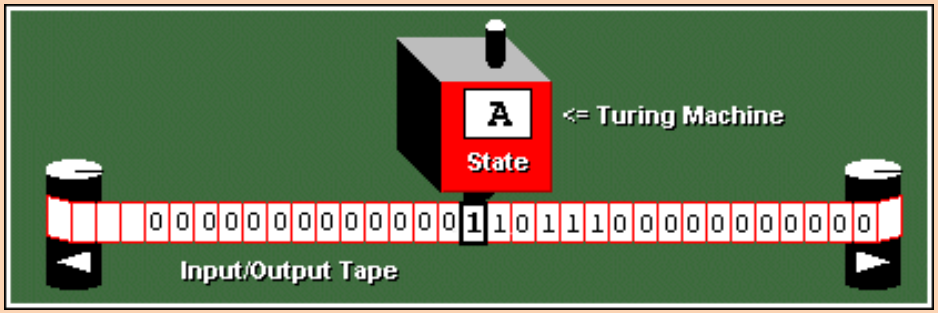
Il concetto chiave: la ***Macchina di Turing***

- Informalmente: una vecchia macchina da scrivere
- In germe: il primo prototipo di calcolatore moderno (10 anni prima dell'ENIAC di John Von Neumann)

La Tesi di Church-Turing: “Un problema si può calcolare se e solo se c’è una macchina di Turing che lo calcola”

Argomenti a favore

- L’assenza di controesempi
- L’equivalenza con altri approcci alla computabilità (il lambda-calcolo di Church, la ricorsività di Godel-Kleene, ...)
- L’ambizione di simulare meccanicamente il pensiero umano (il modello dell’*Impiegato diligente*)



Per molti versi valida ancora oggi

Sulla sua base: problemi (matematici e non solo) che **NON si possono risolvere**

Un dubbio sui problemi che si possono risolvere: il costo di una computazione - sempre sostenibile?

***Un esempio dalla Crittografia*:**

- la macchina Enigma,
- Turing a Bletchley Park: la necessità di una crittoanalisi ***rapida***!

Una conferma devastante (Fischer-Rabin, 1974): ci sono problemi matematici che si possono risolvere ***solo*** in tempi almeno esponenziali (nei casi peggiori).

Nota Tempi esponenziali sono proibitivamente lunghi: $n \rightarrow 2^n$ cresce “troppo” rapidamente

L’invenzione degli scacchi. La ricompensa sui 64 quadri della scacchiera:

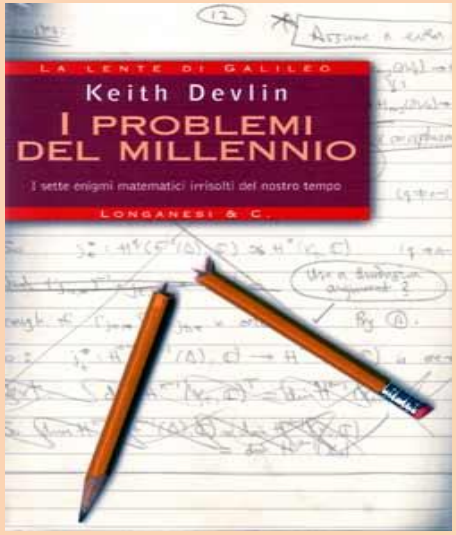
- 1 chicco di grano sul primo quadro
- ad ogni nuovo quadro si raddoppia il numero di chicchi

In totale: $1 + 2 + 2^2 + \ldots + 2^{63} = 2^{64} - 1$ chicchi di grano: una quantità sterminata!

Una nuova prospettiva: problemi risolubili a costo accessibile.
La teoria della ***Complessità Computazionale***.

Quale parametro per misurare il "costo" di una computazione?
Il più ragionevole (ma non l'unico): il ***tempo***

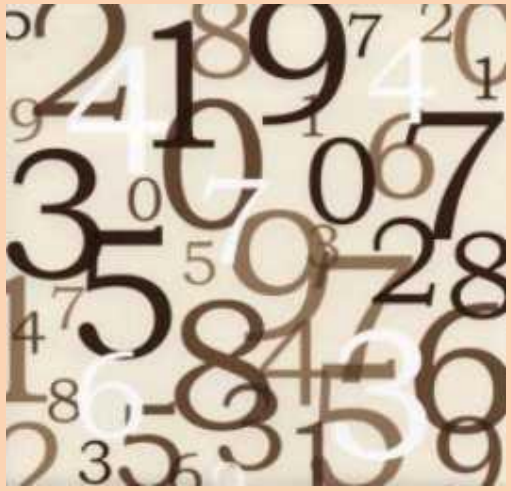
Ma che significa "tempo rapido"?
Un problema del millennio: P = NP?



Dalle speculazioni teoriche ad esempi concreti: dove trovare funzioni a senso unico?

I numeri naturali 0, 1, 2, 3, 4, 5, ...

- apparentemente banali
- nascondono i più grossi misteri della matematica



2 citazioni (libere)

- L. Kronecker: *“i soli creati da Dio”*
- A. Weil: *“dimostrano l’esistenza di Dio e anche quella del diavolo”*

I misteri più grandi sui naturali riguardano la divisione

Un naturale $N > 1$ è

- *primo* se gli unici divisori di N sono 1 e N
- *composto* altrimenti

Teorema fondamentale dell'Aritmetica. Ogni naturale $N > 1$ si decompone in uno ed un solo modo (a meno dell'ordine dei fattori) come prodotto di numeri primi.

$$15 = 3\cdot 5, \quad 18 = 2\cdot 3^2, \quad 24 = 2^3\cdot 3, \quad 32 = 2^5 , \ \ldots$$

Due problemi

Un input comune: $N \in \mathbf{N}$, $N > 1$

Due output collegati ma diversi:

- (PRIMI) N è primo o composto?
- (FATTORIZZAZIONE) la decomposizione di N nei suoi fattori primi ($\Rightarrow$ per N composto, d divisore di N, $1<d<N$)

Un algoritmo elementare (conosciuto già dagli antichi Greci): si esplora $1 < d < N$ e si divide N per d

- se la divisione è esatta per qualche d, si ha che N è composto (e si trova un suo divisore $d \neq 1, N$),
- se la divisione non è esatta per nessun d, N è primo.

Possibili scorciatoie:

- basta controllare $1 < d <$ radice quadrata di N
  (se $N = d \cdot q$ con $d, q > 1$, allora d o q è minore o uguale della radice quadrata di N)
- se d non funziona, neppure 2d, 3d, ... funzionano

Gauss, *Disquisitiones Aritmeticae*, 1801 (articolo 329)

"Il problema di separare i primi dai composti e di decomporre i secondi nei loro fattori primi è conosciuto essere uno dei più importanti ed utili in Matematica. La dignità stessa della scienza sembra richiedere di esplorare ogni possibile mezzo per la soluzione di un problema così elegante e famoso."

Perché questa esigenza due millenni dopo i Greci?

Gauss: *“Le tecniche conosciute in precedenza richiederebbero una fatica intollerabile anche per i più instancabili* ***calcolatori****.”*

Algoritmo elementare: per controllare N un numero di divisioni fino alla radice quadrata di N, esponenziale rispetto alla lunghezza di N

Possibile separare PRIMI da FATTORIZZAZIONE!

Teorema di Wilson. Per N naturale > 1, N è primo se e solo se (N – 1)! + 1 è divisibile per N.

Da ricordare: $(N - 1)! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (N - 1)$

- Controlla PRIMI, non FATTORIZZAZIONE
- Richiede il calcolo di (N –1)! dunque N-2 moltiplicazioni (un numero esponenziale rispetto alla lunghezza di N)

Uno dei tanti misteri sui primi: la Congettura di Goldbach

Apostolos Doxiadis, *Zio Petros e la congettura di Goldbach*

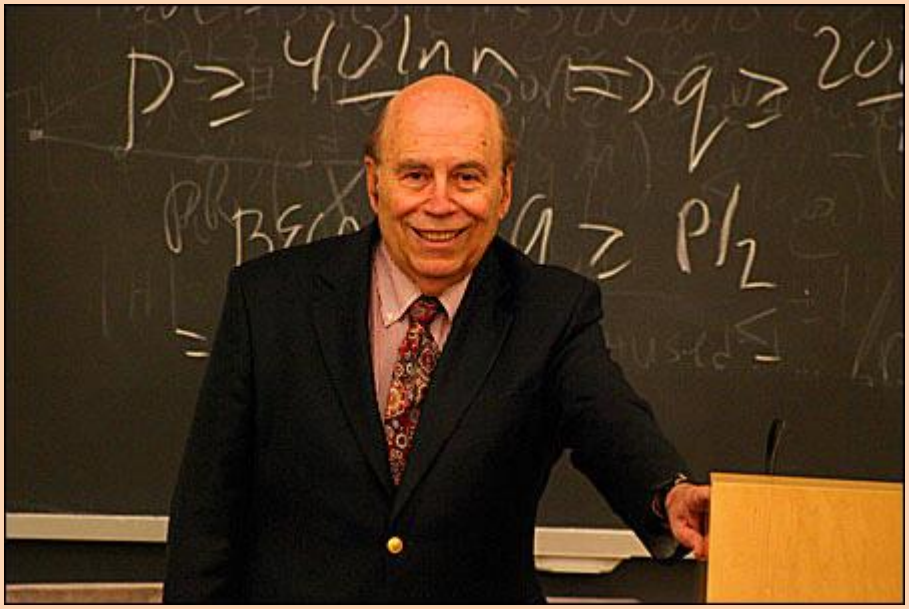
C. Goldbach 1742, lettera a L. Euler: "*Ogni $N \geq 6$ è la somma di al più 3 primi (non necessariamente distinti)*"

L. Euler: "*Basta provare che ogni numero pari $N \geq 4$ è la somma di 2 primi*" (perché?)

4 = 2 +2, 6 = 3 +3, 8 = 3 +5, 10 = 5 +5 = 3 + 7

Riconoscere i PRIMI

1978: un algoritmo efficiente ma fallibile (*Miller-Rabin*)



L'idea: sacrificare la precisione per aumentare la velocità $\Rightarrow$ si ammettono risposte sbagliate o silenzi, purché la probabilità di errore sia bassa

E. Borel: un evento che ha probabilità < 10^{-50} non accadrà mai, e se anche accade non sarà mai rilevato.

Algoritmi probabilistici veloci

- Montecarlo: risposte probabilmente vere in tempi certamente rapidi (si ricorre a testimoni che possono mentire)
- Las Vegas: risposte certamente vere in tempi probabilmente rapidi (si ricorre a testimoni che possono tacere).

L'algoritmo di Miller-Rabin

- *probabilità di errore* dopo il primo tentativo: è al più 1/4 (perché 3 testimoni a su 4 sono onesti),
- *tempi di lavoro*: un polinomio di grado 5 rispetto alla lunghezza di N.

Ma dopo 100 applicazioni con esito concorde

- *probabilità di errore* si riduce sotto $4^{-100} < 10^{-50}$
- *tempo di lavoro* ancora un polinomio di grado 5 rispetto alla lunghezza di N.

2002: un algoritmo efficiente e infallibile (*AKS Agrawal-Kayal-Saxena*)

Tempi di lavoro (nella ”implementazione” di Lenstra-Pomerance, 2005): approssimativamente un polinomio di grado 6 nella lunghezza dell’input.

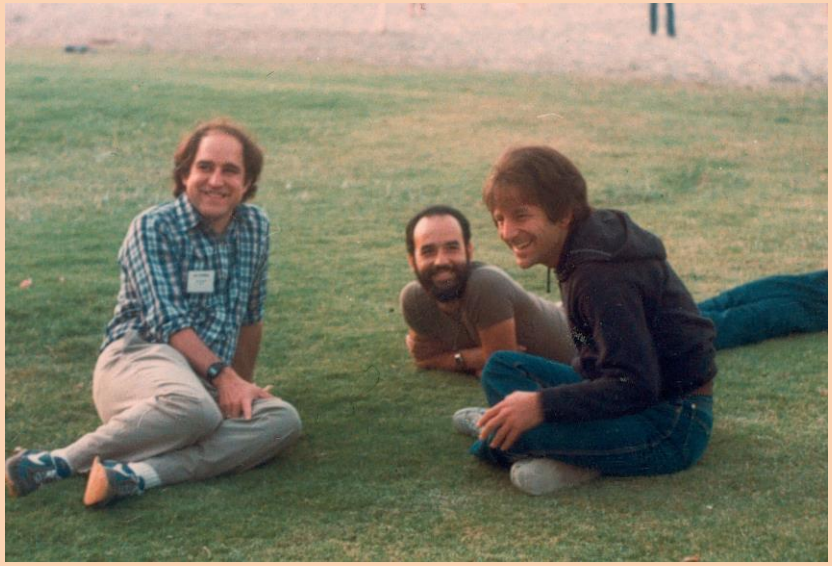
FATTORIZZAZIONE

Vari metodi (curve ellittiche, frazioni continue, ...), nessuno polinomiale nella lunghezza di N: ma attenzione ai procedimenti quantistici (P. Shor, 1994) !

L’algoritmo del letamaio (Lenstra jr.): *“Ammettiamo di avere due numeri primi $p \neq q$ e il loro prodotto $N = p \cdot q$ e di perdere p, q in un letamaio, così che ci rimane solo N. Deve essere sentito come una sconfitta della scienza il dover ammettere che l’algoritmo più rapido per recuperare p e q è quello di cercare nel letamaio”*

Ritorno alla crittografia

Il criptosistema RSA (*Rivest-Shamir-Adleman*, 1978)



La funzione a senso unico: la *moltiplicazione*

Per p ≠ q primi grandi

- facile calcolare $N = p \cdot q$,
- lento e difficile recuperare p, q dal prodotto N .