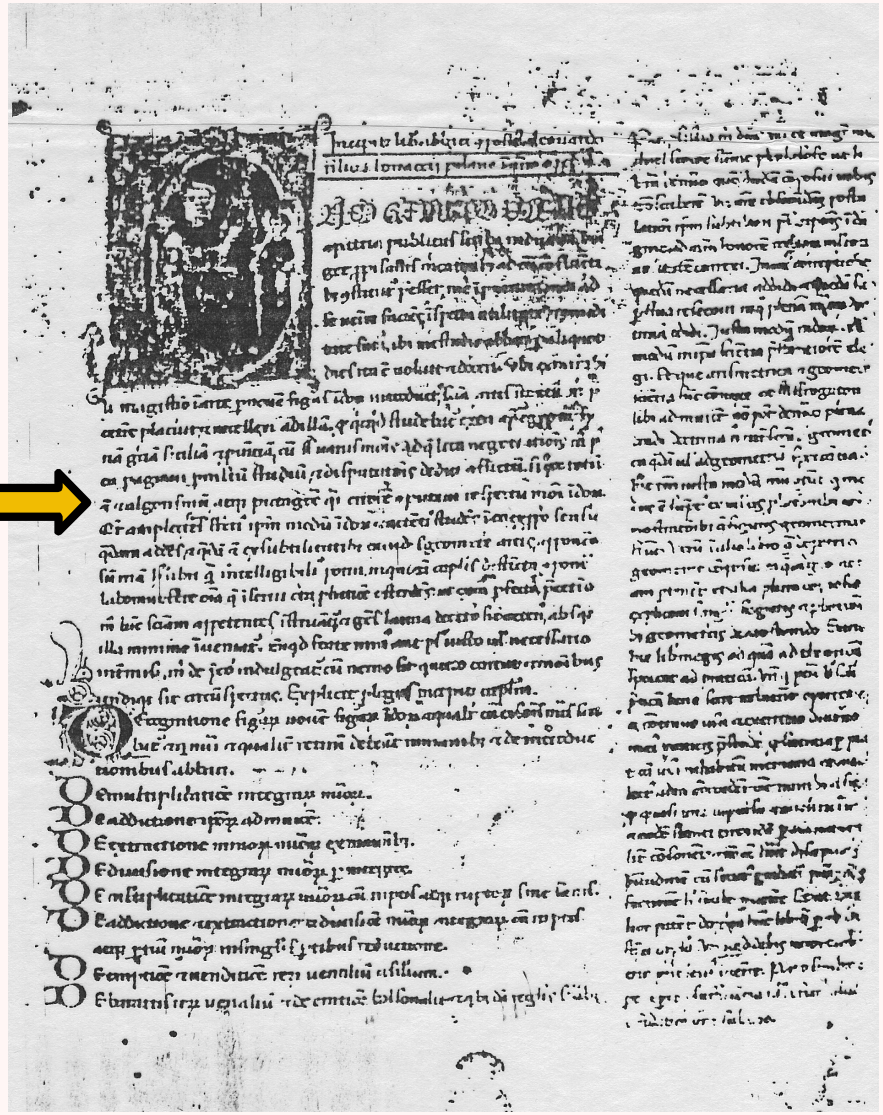
Quattro salti nel mondo degli algoritmi

Fabrizio Luccio

Camerino 2009



La crescita esponenziale

che poi si ritrova da per tutto

La rappresentazione

La complessità di calcolo

La casualità

La rappresentazione dell'informazione

Impiegando un alfabeto di k simboli il numero p di "parole" di lunghezza n cresce esponenzialmente con n :

$$p = k^n$$

Parole: aaa aab aac . . . . zzz

$k = 26,\ n = 3,\ p = 26^3 = 17.576$

per $k \geq 2$ si ha: $k^n > k^{n-1} + k^{n-2} + \ldots + k^1$

Il sapere contenuto in tutti i nostri documenti, scrittura, immagine o suono, si registra nei mezzi elettronici come una immensa sequenza di caratteri di un alfabeto binario (bit) rappresentati per convenzione con 0 e 1 .

Le immagini sono rappresentate come insiemi di pixel, percepiti dall'occhio umano come forme continue. Una sequenza di 24 bit rappresenta il colore di un pixel:

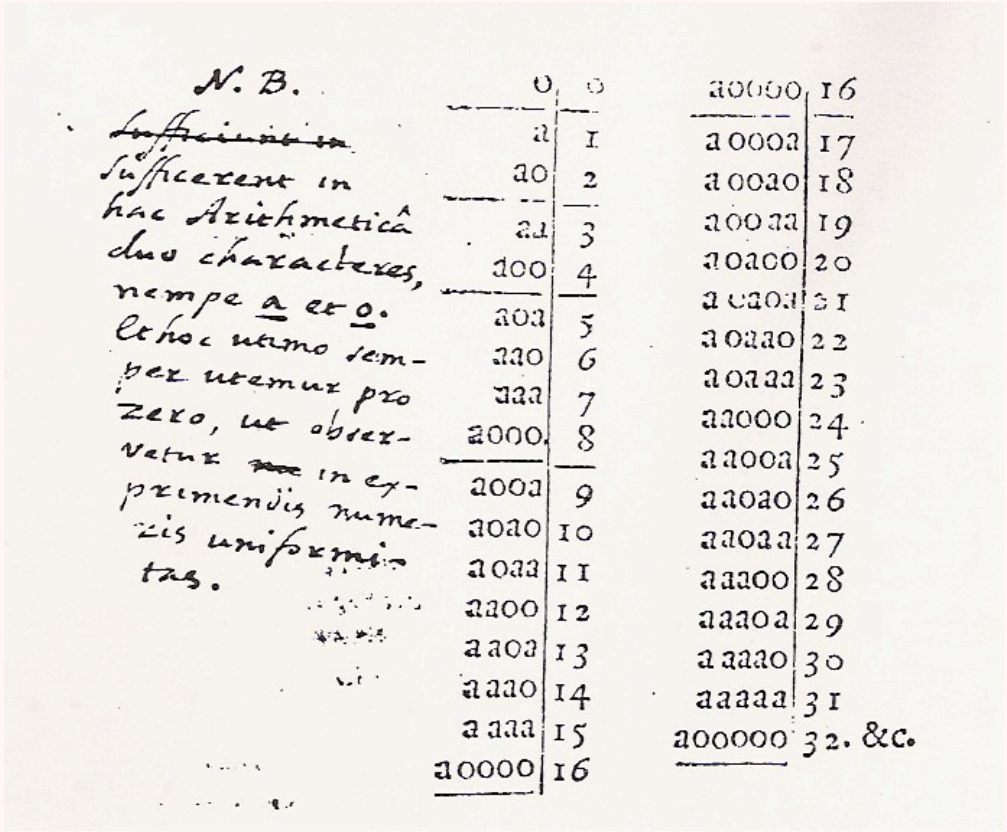
0000 0000 0000 0000 0100 0000

è il colore blu dello sfondo nella proiezione originale

Nel 1605 Francis Bacon dimostrò come le lettere dell' alfabeto potessero essere ridotte a sequenze di cifre binarie . . . . .

. . . . . e aggiunse come il metodo potesse essere utilizzato per rappresentare qualsiasi oggetto "*provided those objects be capable of a twofold difference only; as by Bells, by Trumpets, by Lights and Torches, by the report of Muskets, and any instruments of like nature*" .

La "scoperta" dei numeri binari



N. B.
~~Sufficiunt in~~
sufficerent in
hac Arithmeticâ
duo characteres,
nempe a et o.
Et hoc usimo sem-
per utemur pro
Zero, ut observ-
etur ~~non~~ in ex-
primendis nume-
ris uniformi-
tas.

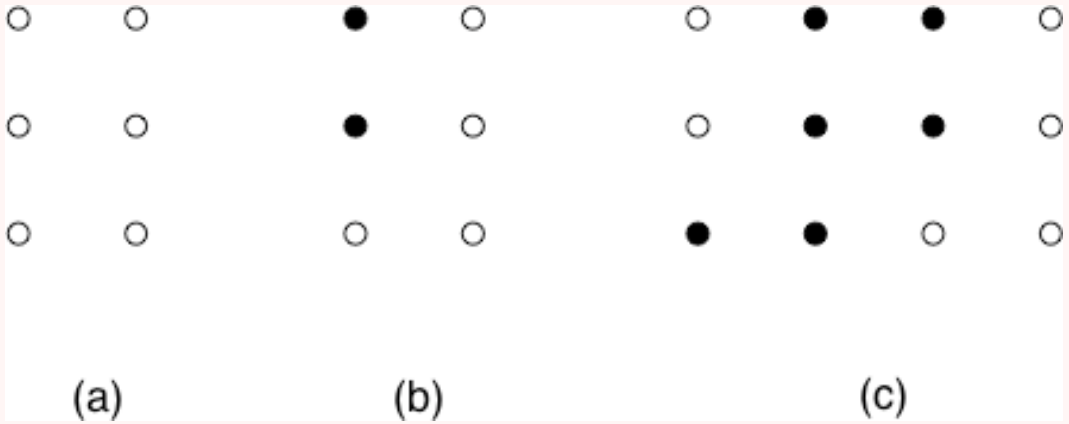
o	0
a	1
ao	2
aa	3
aoo	4
aoa	5
aao	6
aaa	7
aooo	8
aooa	9
aoao	10
aoaa	11
aaoo	12
aaoa	13
aaao	14
aaaa	15
aoooo	16

aoooo	16
aoooa	17
aooao	18
aooaa	19
aoaoo	20
aoaoa	21
aoaao	22
aoaaa	23
aaooo	24
aaooa	25
aaoao	26
aaoaa	27
aaaoo	28
aaaoa	29
aaaao	30
aaaaa	31
aooooo	32. &c.

Meditatio Proemialis.

Da: Johannes Caramuel's *Matesis Biceps Vetus et Nova*, 1670

Il Braille: un codice binario sorprendente

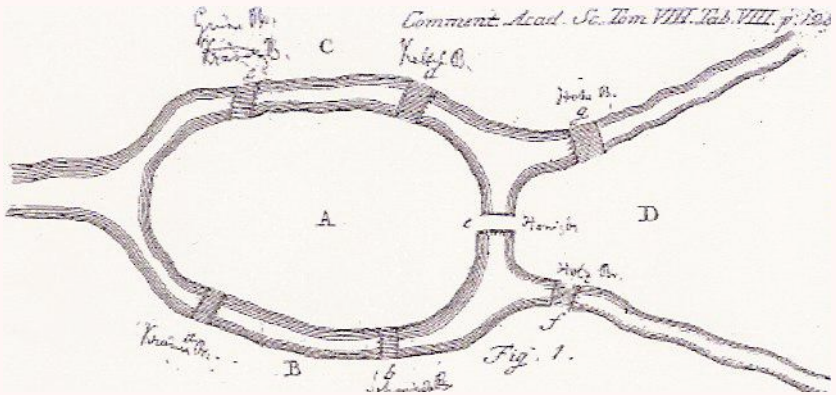


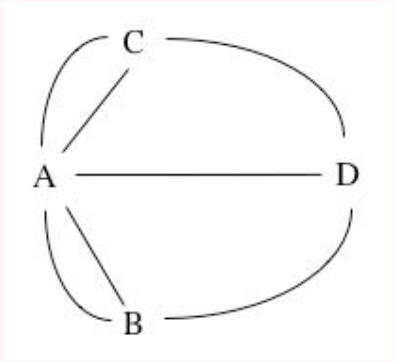
(a) Le sei posizioni dei punti
(b) La lettera B: i punti in rilievo sono in nero.
(c) il numero 2, composto da un "segno numeri" per il "cambio di ambiente", seguito dalla lettera B.

	T	**C**	**A**	**G**
T	TTT Phe (F)	TCT Ser (S)	TAT Tyr (Y)	TGT Cys (C)
	TTC "	TCC "	TAC "	TGC "
	TTA Leu (L)	TCA "	TAA **Ter**	TGA **Ter**
	TTG "	TCG "	TAG **Ter**	TGG Trp (W)
C	CTT Leu (L)	CCT Pro (P)	CAT His (H)	CGT Arg (R)
	CTC "	CCC "	CAC "	CGC "
	CTA "	CCA "	CAA Gln (Q)	CGA "
	CTG "	CCG "	CAG "	CGG "
A	ATT Ile (I)	ACT Thr (T)	AAT Asn (N)	AGT Ser (S)
	ATC "	ACC "	AAC "	AGC "
	ATA "	ACA "	AAA Lys (K)	AGA Arg (R)
	ATG Met (M)	ACG "	AAG "	AGG "
G	GTT Val (V)	GCT Ala (A)	GAT Asp (D)	GGT Gly (G)
	GTC "	GCC "	GAC "	GGC "
	GTA "	GCA "	GAA Glu (E)	GGA "
	GTG "	GCG "	GAG "	GGG "

La complessità di calcolo

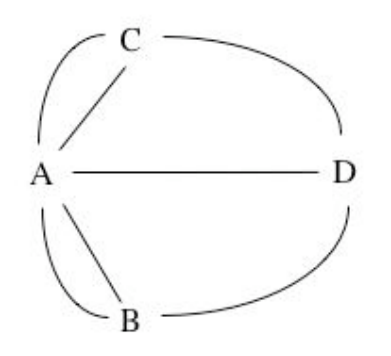
San Pietroburgo, 1735

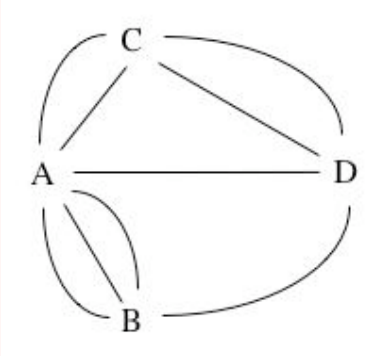




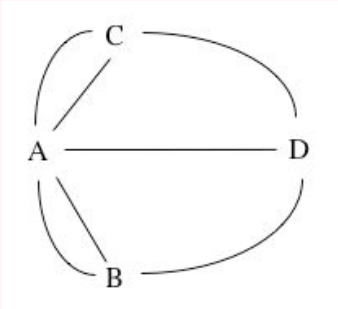
La condizione di Eulero:

Esiste un ciclo che traversa tutti gli archi (ponti) esattamente una volta se e solo se tutti i nodi (zone della città) hanno grado pari.

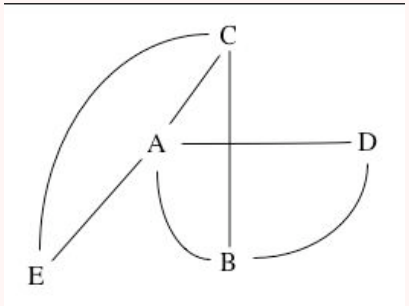




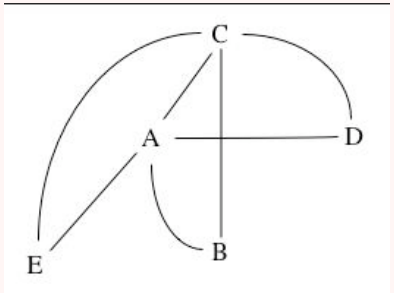
Il ciclo Hamiltoniano traversa tutti i nodi esattamente una volta



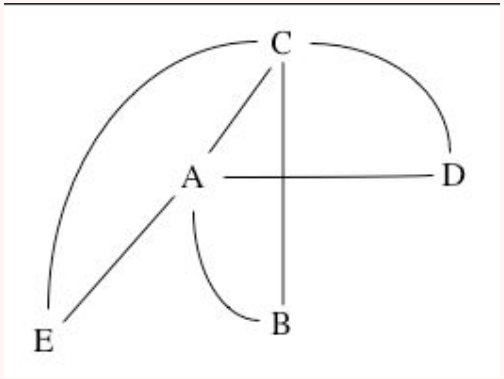
A B D C A



A D B C E A



? ?



A B C D E ?

A B C E D ?

. . . . . .

In linea di principio si devono fare n! = 120 prove

2	3	4	5	6	7	8	9	10
2	6	24	120	720	5040	40320	362880	> 3.5M

n! cresce come $(n/e)^n$

La complessità di calcolo

è il "tempo" (cioè il numero di operazioni elementari) necessario per risolvere un problema.

Sono complessi i problemi che richiedono tempo esponenziale

Complessità polinomiale e esponenziale

Un algoritmo di complessità n^s risolve un problema P su n dati in tempo t su un computer A. Lo stesso algoritmo, su un computer B k volte "più veloce" di A, risolve tempo t lo stesso problema su m dati e si ha:

$$c\, n^s = t \qquad c\, m^s = k\, t \qquad \text{dunque} \qquad m = k^{1/s}\, n$$

Ripetiamo il ragionamento con un algoritmo di complessità 2^n. Abbiamo:

$$c\, 2^n = t \qquad c\, 2^m = k\, t \qquad \text{dunque} \qquad m = n + \log_2 k$$

Le classi di complessità P = NP ?

P è la classe dei problemi che si risolvono in tempo polinomiale.

NP è la classe dei problemi che si verificano in tempo polinomiale (ma si sanno risolvere solo in tempo esponenziale).

NPC è la sottoclasse di NP tale che qualsiasi problema in NP si riduce a NPC in tempo polinomiale.

NPH è la classe dei problemi (non in NP) che si riducono a NPC in tempo polinomiale.

Un esempio molto interessante

Le equazioni diofantee

$ax + by + c = 0$ è in P

linea retta: risolubile in tempo polinomiale nella lunghezza di a, b, c

$ax^2 + by + c = 0$ è in NPC

parabola: risolubile in tempo polinomiale nel valore di a, b, c

$E = 0$ è in NPH

equazione di grado arbitrario e qualunque numero di variabili:

in effetti $E = 0$ non è risolubile ! algoritmicamente

La casualità

Il 28 Dicembre 1985, dopo più di tre anni, uscì il “ritardato” 34 sulla ruota di Napoli. L'annuncio fu diffuso alla televisione mentre la gente gioiva nelle strade (faccia ‘ngialluta ha fatto la grazia). Nel periodo in cui il 34 era rimasto nascosto le scommesse sul ritardato erano aumentate a dismisura.

Esu

dio del caso

Pierre-Simon Laplace

Essai philosophique sur les probabilités (1814)

Les combinaisons régulières n'arrivent plus rarement, que parce qu'elles sont moins nombreuses. Si nous recherchons une cause, là où nous apercevons de la symétrie, ce n'est pas que nous regardions un événement symétrique comme moins possible que les autres, mais cet événement devant être l'effet d'une cause régulière, ou celui du hasard, la première de ces suppositions est plus probable que la seconde.

L'esperimento di Laplace

C C C C C C C C C C C C C C C C C C C C
G E N E R A - 2 0 - C

C P P C C C P C P C P C C P P P C C P C
G E N E R A " C P P C . . . . C "

Una sequenza è casuale se non ammette alcun algoritmo di generazione (o regola per descriverlo) più corto della sequenza stessa

La definizione formale

h è una sequenza binaria arbitraria;

S_0, S_1, S_2,.... è una famiglia (infinita e numerabile) di sistemi di calcolo, per esempio Macchine di Turing;

p è un "programma" per S_i che produce

$$h\text{: } S_i(p) = h;$$

$$K_i(h) = \min\{|p|\text{: } S_i(p) = h\}$$

è la **complessità algoritmica** di h secondo S_i

S_u è un *sistema universale* che simula il funzionamento degli altri. Un suo programma è

$$<i,p>: \quad S_u(<i,p>) = S_i(p) = h$$

$$|<i,p>| = |p| + \lceil \log_2 i \rceil \quad \Rightarrow \quad K_u(h) \leq K_i(h) + c_i$$

$K(h) = K_u(h)$ è la **complessità algoritmica** di h, cioè il contenuto informativo della sequenza

Una sequenza h è **casuale** se

$$K(h) \geq |h| - \lceil \log_2 |h| \rceil$$

La *complessità algoritmica* di una sequenza, ora interpretata come suo contenuto d'informazione, è *la lunghezza del più corto algoritmo che la genera*.

> La *casualità* è definita come proprietà intrinseca di una sequenza *indipendentemente dalla sorgente che l'ha generata*. Una sequenza che risponde a una semplice regola è *non casuale* anche se è stata generata perfettamente a caso.

Questa definizione di casualià *prescinde dall'esistenza*, filosoficamente discutibile, di sorgenti che generano elementi a caso.

Dunque secondo la teoria di Kolmogorov le sequenze casuali contengono la massima informazione. Siamo pronti ad accettare tutto questo?

Notiamo ora che, per il fenomeno della crescita esponenziale, le sequenze lunghe *n* sono più numerose di quelle lunghe meno di *n* indipendentemente dall'alfabeto impiegato . . . .

Dunque *le sequenze casuali esistono*

ma stabilire se un'arbitraria sequenza è casuale *è un problema indecidibile*.

Siano σ_0 , σ_1, σ_2 ,.... le sequenze di caratteri di un alfabeto arbitrario, ordinate per lunghezza crescente.

Poniamo che esista un *programma R(i)* che decida se σ_i è una sequenza casuale.
Utilizzando il paradosso di Berry sugli interi (1908) dimostriamo che questo genera una contraddizione.

Consideriamo il nuovo programma:

programma *A*
  for $i = 1$ **to** ∞
    if $\{R(i) =$ **true** $\cap \ |\sigma_i| > |R| + |A|\}$
      print < i > **and stop**.

Il programma *A* individua (*genera*) una sequenza σ_i casuale e più lunga del programma che l'ha generata: una contraddizione.

Dobbiamo concludere che ***R*** **non esiste.**

La compressione dei dati

Jorge Luis Borges
Del rigor en la ciencia (1946)

En aquel Imperio, el Arte de la Cartografía logró tal Perfección que el Mapa de una sola Provincia ocupaba toda una Ciudad, y el Mapa del Imperio, toda una Provincia. Con el tiempo, estos Mapas Desmesurados no satisficieron y los Colegios de Cartógrafos levantaron un Mapa del Imperio, que tenía el Tamaño del Imperio y coincidía puntualmente con él.

Fissato un sistema di riferimento, la massima compressione di una sequenza *S* si ottiene sostituendo *S* con l'algoritmo più corto che lo genera. Quindi le sequenze casuali non possono essere compresse.

Nel DNA le sequenze "non codificanti" hanno molte ripetizioni e possono essere rappresentate in forma compressa. Invece le sezioni codificanti (geni) non sono comprimibili con i software standard, e sembrano quindi casuali.

Algoritmi randomizzati

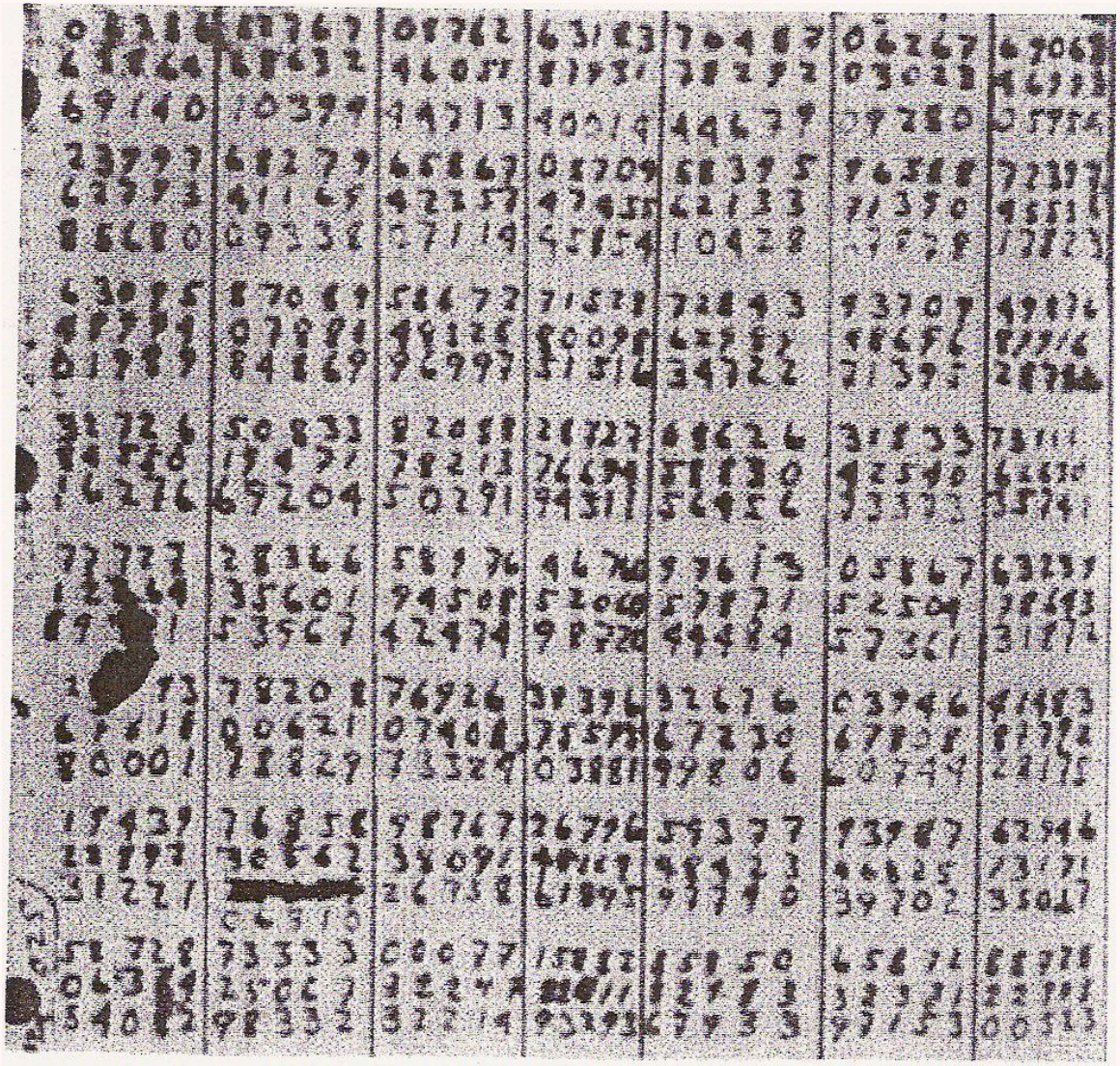
Algoritmi Las Vegas : determinano soluzioni *certamente* corrette in tempo *probabilmente* breve.

> Esempio: da un insieme contenente N elementi di due specie diverse A e B in ugual numero, estrarre un elemento della specie A.

Algoritmi “Monte Carlo” (la grande innovazione degli anni ’70): determinano soluzioni *probabilmente* corrette in tempo *certamente* breve.

> Il primo vero esempio è un algoritmo per determinare se un numero N è primo. Ma vediamo qualcosa di molto più semplice . . . .

Algoritmi crittografici



Un algoritmo crittografico non può essere mantenuto segreto a lungo . . . .

quindi le comunicazioni devono essere basate su un'informazione addizionale, la chiave, che sia mantenuta segreta e possa essere modificata facilmente.

L'algoritmo deve essere pubblico,
solo la chiave deve essere segreta

Il cifrario One-Time Pad (1917) impiega un codice binario per messaggi e chiavi, eseguendo tra essi l'operazione di OR-ESCLUSIVO

```
M:  0 1 0 0 0 1 0 1 1 0 1 0 . . .
K:  1 1 0 1 0 0 0 1 0 1 1 0 . . .
C:  1 0 0 1 0 1 0 0 1 1 0 0 . . .
```

Se si impiega una chiave lunga come il messaggio, casuale e non riutilizzabile, il cifrario One-Time Pad diviene matematicamente inattaccabile!

Intuitivamente se il messaggio M è lungo *n*

vi sono 2^n chiavi possibili, e ciascuna genera un diverso crittogramma come trasformazione casuale di M

One-Time Pad fu usato nella Hot Line
per le comunicazioni tra la Casa Bianca
e il Cremlino a partire dal 1967

Un piccolo libro per cominciare

F. Luccio, L. Pagli.
Algoritmi, divinità e gente comune.
Edizioni ETS, Pisa 1999.

http://www.edizioniets.com/