ESEMPIO

Un codice fiscale italiano per le persone fisiche è sempre costituito da 16 caratteri alfanumerici:

MNEGRG 65C23 E625 Y

I primi 6 caratteri sono normalmente estratti dal cognome e dal nome della persona, i successivi 5 ne rappresentano la data di nascita (le ultime due cifre dell'anno, il mese codificato mediante una singola lettera e il giorno) e il sesso (il giorno di nascita di una femmina è incrementato di 40), i successivi 4 codificano il comune italiano o lo stato estero di nascita e l'ultimo carattere... è in più! Esso viene calcolato mediante un algoritmo a partire dai primi 15 e può essere ricalcolato utilizzando lo stesso algoritmo ogni volta che un codice fiscale viene digitato o acquisito: nel caso che il risultato del calcolo sia lo stesso carattere che compare nel codice si può ragionevolmente assumere che il codice sia corretto, mentre se il risultato è un carattere diverso si ha la matematica certezza che esso è errato.

# 1 Entropia come misura della quantità di informazione

Il nome di Claude Shannon è legato alla sua definizione di «**entropia dell'informazione**»: questa grandezza esprime la quantità di informazione contenuta in una sequenza di simboli – per esempio una stringa di caratteri alfanumerici – e al tempo stesso determina il numero di bit necessari per codificare un singolo simbolo della sequenza.

Nel capitolo precedente è stato mostrato che disponendo di $b$ bit è possibile codificare $2^b$ simboli diversi mediante i numeri binari compresi tra 0 e $2^b-1$. Ma quanti bit servono per rappresentare $N$ simboli diversi, cioè i numeri compresi tra 0 ed $N-1$? Se con $b$ bit è possibile rappresentare al massimo il valore $2^b-1$, allora applicando la funzione inversa dell'esponenziale, il logaritmo, si ha che per codificare il numero $N-1$ sono necessari

$$\log_2 N \text{ bit.}$$

Per effettuare i calcoli con una normale calcolatrice che permette di calcolare il logaritmo in base 10 è possibile calcolare il logaritmo in base 2 applicando la seguente formula

$$\log_2 N = \log_{10} N : \log_{10} 2$$

dove $\log_{10} 2$ vale circa 0,301.

### Entropia termodinamica

Shannon ha ripreso il termine «entropia» dalla grandezza della fisica termodinamica che rappresenta l'impossibilità di trasformare completamente l'energia termica in lavoro meccanico e che è spesso interpretata come grado di disordine o casualità di un sistema termodinamico.
La formula per determinare l'entropia di un gas formulata dal fisico Ludwig Boltzmann nella seconda metà dell'Ottocento è strutturalmente simile a quella di Shannon per la quantità di informazione di una sequenza di simboli.

ESEMPIO

Nel caso $N = 1000$ si ha

$$\log_2 (1000) = \log_{10} (1000) : \log_{10} 2 = 3 : 0{,}301 = 9{,}967 \text{ bit}$$

Naturalmente non potendo usare una frazione di bit è necessario approssimare all'intero superiore per cui servono almeno 10 bit per rappresentare i numeri interi compresi tra 0 incluso e 1000 escluso.

La teoria di Shannon individua nella funzione logaritmo (la base 2 è una conseguenza del ricorso alla codifica binaria) un modo per esprimere la quantità di informazione. Per capire il motivo, è possibile analizzare un approccio «informativo» alla costruzione dei codici binari che rappresentano valori numerici (che a loro volta possono codificare simboli, per esempio di caratteri tipografici).

Disponendo i valori numerici compresi tra 0 e $2^b-1$, con $b$ massimo numero di bit utilizzabile, sequenzialmente e in ordine crescente, è possibile definire il seguente procedimento di costruzione del codice binario di un valore numerico:

1. dividere la sequenza a metà;
2. se il valore numerico cade nella metà sinistra scrivere il simbolo binario «0», altrimenti – se cade nella metà destra – il simbolo binario «1»;
3. ripetere il procedimento dal punto 1 limitatamente alla parte di sequenza in cui cade il valore numerico se essa è costituita da più di un elemento: scrivere il successivo simbolo binario a destra dei precedenti.

**ESEMPIO**

Per $b = 3$ i valori numerici sono quelli compresi tra 0 e 7:

0 1 2 3 4 5 6 7

Nel caso del valore numerico 2 la prima iterazione del procedimento

0 1 2 3 | 4 5 6 7

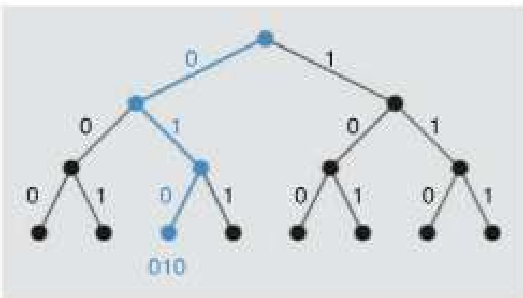
genera il simbolo binario «0»; la seconda iterazione del procedimento

0 1 | 2 3

genera il simbolo binario «1»; la terza e ultima iterazione del procedimento

2 | 3

genera il simbolo binario «0»: il codice è quindi «010» che corrisponde effettivamente al formato binario del valore numerico 2.
Un modo alternativo di illustrare questo procedimento è mediante un diagramma ad «albero binario»:




**OSSERVAZIONE** La tecnica illustrata dall'esempio precedente è nota come «ricerca binaria» ed è alla base dei giochi in cui si possono porre domande aventi risposte limitate a «sì» o «no». Un classico esempio è quello di indovinare un numero ignoto compreso tra 0 e un valore massimo $M$ potendo porre soltanto la domanda «è più grande di...?»: in questo caso è sempre possibile indovinare al massimo con $\log_2 M$ domande, approssimando il valore del logaritmo all'intero superiore. Per esempio per $M = 1000$ si inizierà chiedendo se il numero è più grande di 500: in caso di risposta affermativa si continuerà chiedendo se è più grande di 750, in caso di risposta negativa se è più grande di 250... e così via dividendo sempre a metà l'intervallo dei valori che rimane da esplorare; il numero di scelte che si dovranno effettuare non sarà superiore a $\log_2 1000 = 9{,}967$ che approssimato all'intero superiore è pari a 10.

Per Shannon l'entropia $H$ di una sequenza $S$ di $N$ simboli $s_1, s_2, \ldots s_N$ in cui l'ordine non ha alcuna importanza è espressa dalla seguente formula

$$H(S) = -\sum_{i=1}^{N} f_i \times \log_2 f_i = -(f_1 \times \log_2 f_1 + \ldots + f_N \times \log_2 f_N)$$

dove $f_i$ è la frequenza con cui il simbolo $s_i$ compare nella sequenza, cioè il numero di volte in cui compare diviso il numero di simboli da cui è composta la sequenza stessa.
L'entropia esprime il numero minimo di bit per simbolo – cioè il numero minimo di scelte binarie – necessario per codificare la sequenza: per questa grandezza è stato proposto come unità di misura lo *shannon*.

**ESEMPIO**

Data la sequenza di 8 simboli

$$S = \{\#, @, \#, @, *, @, §, @\}$$

Il simbolo «#» compare 2 volte e ha quindi frequenza 2 : 8 = 0,250; il simbolo «@» compare 4 volte e ha quindi frequenza 4 : 8 = 0,5, i simboli «§» e «*» compaiono ciascuno 1 volta e hanno quindi frequenza 1 : 8 = 0,125.
L'entropia della sequenza è espressa dalla formula di Shannon:

$$-(0{,}250 \times \log_2 0{,}250 + 0{,}500 \times \log_2 0{,}500 + 0{,}125 \times \log_2 0{,}125 + + 0{,}125 \times \log_2 0{,}125) =$$

$$= -(0{,}250 \times -2{,}000 + 0{,}500 \times -1{,}000 + 0{,}125 \times -3{,}000 + 0{,}125 \times -3{,}000) =$$

$$= -(-0{,}500 - 0{,}500 - 0{,}375 - 0{,}375) = -(-1{,}750) = 1{,}750$$

Il risultato stabilisce che la codifica binaria della sequenza non può impiegare meno di 1,75 bit per simbolo; una banale codifica che impiega 2 bit per simbolo potrebbe essere la seguente:

| | |
|---|---|
| # | 00 |
| @ | 01 |
| * | 10 |
| § | 11 |

La codifica dell'intera sequenza *S* composta da 8 simboli richiede 2 × 8 = 16 bit:

00 01 00 01 10 01 11 01

### Codifica *Morse*

Lo schema di codifica dei caratteri alfanumerici, ideato da Samuel Morse e usato per le comunicazioni telegrafiche a partire dalla prima metà dell'Ottocento, era basata su 5 simboli: linea (suono lungo), punto (suono breve), intervallo breve, intervallo medio e intervallo lungo. La codifica di *Morse* non era a lunghezza fissa: la lettera «A» per esempio era codificata da un punto e una linea, mentre la lettera «Z» era codificata da due linee seguite da due punti.

Uno schema di codifica in cui tutti i simboli sono rappresentati con lo stesso numero di bit è definito a **lunghezza fissa** (**FLC**, *Fixed Length Code*). La codifica binaria di tipo FLC di una sequenza $S$ di $N$ simboli ha una lunghezza complessiva minima uguale all'entropia della sequenza $H(S)$ approssimata all'intero superiore moltiplicata per il numero $N$ di simboli.

**OSSERVAZIONE** Data una sequenza $S$ formata da un unico simbolo, per esempio

$$S = \{@, @, @\}$$

la frequenza del simbolo @ è ovviamente 3 : 3 = 1 e l'entropia espressa dalla formula di Shannon è:

$$-(1 \times \log_2 1) = -(1 \times 0) = 0$$

Il contenuto informativo di questa sequenza è nullo! L'informazione infatti è sempre legata alle differenze dei simboli (di forma, di posizione, ...): una sequenza di simboli priva di differenze non esprime nessuna informazione.

L'entropia di una sequenza di simboli – e quindi la quantità di informazione che essa contiene – è invece massima quando ciascuno dei simboli è presente con la stessa frequenza. Per esempio, nel caso di sequenze composte solo da 2 simboli («0» e «1»), la variazione dell'entropia in funzione della loro distribuzione è la seguente:

$$S = \{1, 1, 1, 1, 1, 1, 1, 1\} \rightarrow H(S) = 0$$

$$S = \{0, 1, 1, 1, 1, 1, 1, 1\} \rightarrow H(S) =$$
$$= -(0{,}125 \times \log_2 0{,}125 + 0{,}875 \times \log_2 0{,}875) = 0{,}544$$

$$S = \{0, 0, 1, 1, 1, 1, 1, 1\} \rightarrow H(S) =$$
$$= -(0{,}250 \times \log_2 0{,}250 + 0{,}750 \times \log_2 0{,}750) = 0{,}811$$

$$S = \{0, 0, 0, 0, 1, 1, 1, 1\} \rightarrow H(S) =$$
$$= -(0{,}500 \times \log_2 0{,}500 + 0{,}500 \times \log_2 0{,}500) = 1{,}000$$

$$S = \{0, 0, 0, 0, 0, 0, 1, 1\} \rightarrow H(S) =$$
$$= -(0{,}750 \times \log_2 0{,}750 + 0{,}250 \times \log_2 0{,}250) = 0{,}811$$

$$S = \{0, 0, 0, 0, 0, 0, 0, 1\} \rightarrow H(S) =$$
$$= -(0{,}875 \times \log_2 0{,}875 + 0{,}125 \times \log_2 0{,}125) = 0{,}544$$

$$S = \{0, 0, 0, 0, 0, 0, 0, 0\} \rightarrow H(S) = 0$$

Il grafico – che riporta in ascissa il numero di simboli «1» presenti nella sequenza – mostra che l'entropia è massima quando i simboli della sequenza hanno la stessa frequenza:


L'architettura di un computer non consente di scegliere la lunghezza in bit della codifica di una sequenza di simboli più adatta: i registri della CPU e le locazioni della memoria RAM hanno infatti una lunghezza predefinita (tipicamente 32 o 64 bit) e possono essere utilizzati in modo efficiente solo con codifiche che siano multiple del byte (8 bit). Per questo motivo molti codici utilizzati in pratica hanno codifiche dei simboli multiple di 8 bit: il codice ASCII per i caratteri tipografici per esempio, ma anche i 24 bit della codifica RGB delle immagini digitali.

### Frequenze e probabilità

Partendo da una sequenza di simboli è possibile calcolarne l'entropia e quindi, almeno in teoria, stabilirne una codifica ottima. Quando le sequenze di simboli che si devono codificare non sono note a priori è possibile stimare il valore dell'entropia sostituendo il concetto di frequenza con quello di «probabilità». Per esempio, la probabilità con cui uno specifico carattere compare in un testo abbastanza lungo è una caratteristica specifica della lingua: la lettera «A» ha una probabilità di comparsa in Inglese di 0,0817 (8,17%), in Francese di 0,0764 (7,64%), in Tedesco di 0,0651 (6,51%), in Spagnolo di 0,1253 (12,53%) e in Italiano di 0,1174 (11,74%).

Una codifica non ottimale è definita **ridondante**: con questo termine si indica che il numero di bit per simbolo della codifica è superiore alla misura dell'entropia della sequenza da codificare.

**OSSERVAZIONE** In pratica una codifica FLC è sempre ridondante per il fatto che il numero di bit per simbolo è un numero intero, nel migliore dei casi si tratta il numero intero immediatamente superiore al valore dell'entropia stessa.

# 2 Ridondanza dell'informazione e rilevazione di errori

Alcune codifiche introducono deliberatamente una limitata ridondanza allo scopo di rilevare errori di generazione, memorizzazione o trasmissione dell'informazione. La valutazione della ridondanza di una codifica allo scopo di rilevare o correggere errori si basa sulla cosiddetta **distanza minima di Hamming**, così chiamata dal nome del matematico americano Richard Hamming che la propose negli anni Cinquanta del secolo scorso in un importante studio sugli schemi di codifica per la correzione degli errori.

La **distanza di Hamming** tra due codici binari aventi la stessa lunghezza è il numero di bit che differiscono tra i due codici.

**ESEMPIO**

La distanza di Hamming tra il codice ASCII del carattere «A» e il carattere «Z»

01000001
01011010

è 4 perché le relative codifiche binarie differiscono in 4 diverse posizioni.

La **distanza di Hamming minima** di uno schema di codifica è la minima distanza di Hamming tra due codici validi dello schema.

**ESEMPIO**

La distanza di Hamming minima del codice ASCII è 1 in quanto ogni configurazione binaria a 8 bit codifica un simbolo: esistono quindi codici validi con distanza di Hamming uguale a 1.

**OSSERVAZIONE** Uno schema di codifica con distanza di Hamming minima uguale a 1 non consente la rilevazione di errori, in quanto esistono almeno due configurazioni binarie corrispondenti a un codice valido che differiscono di un solo bit.

In generale la rilevazione degli errori presenti in un codice si ottiene aggiungendo alcuni bit di controllo ridondanti rispetto allo schema di codifica. Il computer che genera o trasmette il codice calcola i bit di controllo a partire dai bit del codice in base a un determinato algoritmo e il computer che elabora o riceve il codice ricalcola i bit di controllo utilizzando lo stesso algoritmo: se i bit di controllo calcolati sono gli stessi di quelli generati o ricevuti si assume la correttezza del codice, se sono diversi si ha la certezza che il codice è errato.

### Codifiche per la correzione degli errori

L'introduzione di una limitata ridondanza negli schemi di codifica permette di rilevare eventuali errori di memorizzazione o trasmissione dell'informazione. Esistono anche schemi di codifica che, a fronte di una ancora maggiore ridondanza dell'informazione, permettono di individuare e quindi di correggere gli errori di memorizzazione o trasmissione. I *QR-code* per esempio sono codificati con lo schema di Reed-Solomon e restano interpretabili anche se il 25% della superficie risulta illeggibile.

Lo stesso schema di codifica ridondante viene utilizzato per i CD audio e i dischi *Blu-ray* per i video: in questo modo la presenza di eventuali graffi sulla superficie non inficia l'ascolto e la visione perché l'informazione viene integralmente ricostruita anche a fronte di errori.