

# Corso di Calcolatori Elettronici

## Informazione e sua rappresentazione: codifica

Roberto Canonico

Università degli Studi di Napoli Federico II

A.A. 2022-2023





- Fornire un'informazione significa effettuare la scelta di un elemento in un insieme assegnato
- Un'informazione è definita tramite tre caratteristiche fondamentali
  - 1 Valore
    - indica il particolare elemento assunto dall'informazione
  - 2 Tipo
    - indica l'insieme degli elementi all'interno del quale è stato scelto il valore
  - 3 Attributo
    - indica il significato associato all'informazione nel contesto in cui questa viene utilizzata

## Esempio

- Valore: 7,50
- Tipo: Insieme dei numeri reali
- Attributo: Soluzione dell'equazione di primo grado  $6x = 45$



Non confondere il valore dell'informazione (*elemento di un insieme*) con la sua rappresentazione

3	III	tre
padre	père	father
0,1	1/10	$1 * 10E(-1)$



- Tipi semplici
  - I valori sono entità atomiche

```
float f;
```

- Tipi strutturati
  - I valori sono aggregati di informazioni di tipi semplici

```
struct persona {  
    char name[SIZENAME];  
    char tfnumb[SIZETELE];  
    char addr[SIZEADDR];  
    struct persona *ptrnext;  
};
```



- Si è detto: *fornire un'informazione significa effettuare la scelta di un elemento in un insieme  $X$  (tipo dell'informazione)*
- Dato un insieme  $X$ , la sua *cardinalità*  $N = |X|$  è il numero di elementi
- Un'informazione il cui tipo ha cardinalità 2 è detta **bit** (*binary digit*)
- Maggiore è la cardinalità del tipo, maggiore è il numero di elementi tra cui scegliere
  - La *Teoria dell'Informazione* (Claude Shannon) introduce il concetto di *quantità di informazione* associata ad una scelta: essa dipende dalla probabilità con cui i singoli valori si possono presentare

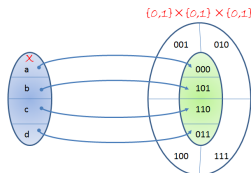
- Rappresentazione del valore  $x_i$  di una informazione di tipo

$$X = \{x_1, \dots, x_N\} \text{ (alfabeto sorgente)}$$

mediante una *parola codice* (codeword)  $c_i$ ,  
cioè una stringa di simboli presi da un insieme

$$A = \{a_1, \dots, a_k\} \text{ (alfabeto codice)}$$

- Se  $k = 2$  si parla di *codice binario*
- *Codice C*: insieme di tutte le parole codice associate agli elementi dell'alfabeto sorgente
- *Codifica*: funzione  $f : X \longrightarrow C$
- *Codifica non ambigua*: una stessa parola codice non può corrispondere a due distinti elementi dell'alfabeto sorgente
  - $f$  è iniettiva





- **Codice a lunghezza fissa**

- tutte le parole codice hanno lunghezza  $l$

$$X \longrightarrow C = A^l = A \times A \times \dots \times A \text{ (prodotto cartesiano di } A \text{ per se stesso, } l \text{ volte)}$$

con  $l$  tale che:  $|A^l| \geq |X|$  ovvero  $k^l \geq N$  ovvero  $l \geq \lceil \log_k(N) \rceil$

- **Codice a lunghezza variabile**

- ogni parola codice  $c_j$  ha una sua lunghezza specifica  $l_j$



Codice binario  
a lunghezza fissa ( $l = 3$ )  
Palette di 8 colori

Nero	000
Grigio	001
Blu	010
Marrone	011
Verde	100
Giallo	101
Rosso	110
Bianco	111

Codice decimale  
a lunghezza variabile  
Prefissi telefonici

Genova	010
Torino	011
Pinerolo (TO)	0121
Susa (TO)	0122
Imperia	0183
Milano	02
Roma	06
...	...





- Per codificare un tipo di cardinalità  $N$  mediante un alfabeto di  $k$  simboli è necessaria una stringa di lunghezza  $l$  tale che sia possibile far corrispondere a ciascun elemento  $x \in X$  una distinta tra le  $k^l$  *disposizioni con ripetizione dei  $k$  simboli di  $A$  sugli  $l$  posti* della stringa
- Per un codice a lunghezza fissa occorre che  $l$  sia tale che:

$$k^l \geq N \text{ ovvero } l \geq \lceil \log_k(N) \rceil$$

- Un codice si dice a *lunghezza minima* se

$$l = m = \lceil \log_k(N) \rceil$$

- I codici dell'esempio successivo (slide 11) sono a lunghezza minima
- Un codice si dice *ridondante* se

$$l > m = \lceil \log_k(N) \rceil$$

- I codici ridondanti si utilizzano per rilevare errori nella trasmissione o memorizzazione delle informazioni
- Una misura della ridondanza:

$$r = 1 - \left\lfloor \frac{(\log_k(N))}{l} \right\rfloor \in [0, 1]$$



- Un codice a lunghezza minima ha  $l$  tale che:

$$l = m = \lceil \log_k(N) \rceil$$

- Un codice a lunghezza minima si dice *completo* se

$$l = \log_k(N)$$

- Condizione necessaria è che  $N$  sia una potenza di  $k$
- Un codice a lunghezza minima si dice *incompleto* se

$$l > \log_k(N)$$

- I codici dell'esempio successivo sono incompleti
- Per un codice incompleto, alcune stringhe di simboli di lunghezza  $l$  **non** sono parole codice

# Codice a lunghezza fissa: esempio

- $X = \{\text{lunedì, martedì, mercoledì, giovedì, venerdì, sabato, domenica}\}$
- $A = \{a, b, c\}$
- $N = 7, k = 3, l = 2 \geq \lceil \log_3(7) \rceil$

## Codifica 1

*lunedì* → aa  
*martedì* → ab  
*mercoledì* → ac  
*giovedì* → ba  
*venerdì* → bb  
*sabato* → bc  
*domenica* → ca

## Codifica 2

*lunedì* → aa  
*martedì* → bb  
*mercoledì* → ba  
*giovedì* → bc  
*venerdì* → ac  
*sabato* → cb  
*domenica* → ca

- I codici sono entrambi a lunghezza minima ed incompleti



- La lunghezza  $l_i$  della parola codice  $c_i$  dipende da  $x_i$ :  $l_i = f(x_i)$
- Requisito di *univoca decodificabilità*: ciascuna parola codice non sia inclusa come sequenza iniziale (*prefisso*) di altre parole più lunghe
- L'uso di questi codici è giustificato quando gli elementi del tipo  $X$  (*alfabeto sorgente*) non hanno tutti la stessa probabilità di occorrenza
  - Parole codice più corte associate a elementi dell'alfabeto origine con maggiore probabilità di occorrenza

Value	Bit sequence	Value	Bit sequence
0	1	9	0001010
1	010	10	0001011
2	011	11	0001100
3	00100	12	0001101
4	00101	13	0001110
5	00110	14	0001111
6	00111	15	000010000
7	0001000	16	000010001
8	0001001	17	000010010



- Un *codice one-hot* è un codice ridondante per il quale tutte le parole codice hanno lunghezza  $l = N$ , e presentano un bit ad 1 ed i restanti  $N - 1$  bit a 0
- Si usa anche l'espressione *rappresentazione decodificata* perché la posizione dell'1 nella parola codice indica immediatamente il simbolo a cui corrisponde la parola codice

## Esempio

Cuori → 1000

Quadri → 0100

Fiori → 0010

Picche → 0001