Prof. Mariacarla Staffa
a.a. 2022/2023

Laboratorio di Architettura Degli Elaboratori

Rappresentazione dei Numeri

Rappresentazione dell'informazione

- Informazione: notizia, dato o elemento che consente di avere conoscenza più o meno esatta di fatti, situazioni, modi di essere
  - Consente di ridurre l'incertezza

- Dato: elementi di informazione costituiti da simboli che debbono essere elaborati

Consideriamo un mazzo di carte

- E peschiamo una carta a caso senza guardarla
  - La carta potrebbe essere una qualsiasi delle 56 a disposizione
- Se vediamo che è una carta di cuori
  - Le possibilità si riducono a 13
- La carta è un asso di cuori

Rappresentazione dell'informazione

- In generale ogni rappresentazione è una funzione che associa ad ogni elemento una sequenza di simboli
- Per ogni rappresentazione, oggetti (numeri) distinti devono avere differenti rappresentazioni e la rappresentazione di ogni oggetto deve essere unica e non ambigua
  - pesca
- Le rappresentazioni usate sui calcolatori impiegano tutte sequenze finite di simboli, tali quindi da rappresentare insiemi finiti di naturali

The Digital Abstraction

Most physical variables are **continuous**

Voltage on a wire	Frequency of an oscillation	Position of a mass

Digital abstraction considers **discrete subset** of values

Alfabeti, parole, linguaggi

- Per alfabeto A intenderemo un insieme finito e distinguibile di segni che chiameremo a seconda del contesto cifre, lettere, caratteri, simboli etc.
  - Le nove cifre dell'alfabeto decimale A={'0',…,'9'}
  - Le ventisei lettere (minuscole) dell'alfabeto A={'a',…,'z'}
  - I quattro simboli delle carte francesi A={♥,♦,♣,♠}
- Una parola (o stringa) su A è una sequenza finita di simboli dell'alfabeto A
  - "18945" (A={'0',…,'9'})
  - "pkwocod" (A={'a',…,'z'})
  - "♥,♦,♣" (A={♥,♦,♣,♠})
- Con A* indicheremo tutte le possibili parole generabili a partire dall'alfabeto A

Alfabeti, parole, linguaggi

Un linguaggio L sull'alfabeto A è un qualsiasi sottoinsieme di A^*.

Parole italiane

- "pwfnfkr"
- "dog"
- "siengs"
- "casa"
- "door"
- "porta"

Parole inglesi

- "pwfnfkr"
- "dog"
- "siengs"
- "casa"
- "door"
- "porta"

Codifica, decodifica

- Le parole di un linguaggio non sono altro che sequenze di simboli che non hanno alcun senso finché non forniamo un modo per interpretarle
    - "♣ ♣♦♠" ?
- Associare gli elementi di un insieme D alle parole di un linguaggio L viene detta codifica o rappresentazione di D.
- Ad esempio sia D l'insieme dei concetti cane, gatto e aquila (notate bene che sto parlando dei concetti non delle parole "cane", "gatto" e "aquila"). Allora nella usuale codifica della lingua inglese

  Cane → "dog" Gatto → "cat" Aquila → "eagle"
- Formalmente una codifica è una funzione totale $f:D \rightarrow L$
    - Se f non è suriettiva allora diremo che è ridondante (1 a 1)
    - Se f non è iniettiva allora diremo che ambigua (più di 1 a 1)

Codifica, decodifica

- Sia D={cane, gatto, aquila} e
- L={“♣ ♣♦♦”, “♣ ♣ ♣”, “♦♦”}
- e sia f rappresentata graficamente

cane → “♣ ♣♣”
gatto → “♣ ♣♦♦”
aquila → “♣ ♣♣”

“♦♦”

- f è ambigua e ridondante

Codifica, decodifica

- Sia D={cane, gatto, aquila} e L={“♣ ♣♦♦”, “♣ ♣ ♣”, “♦♦”, “♦”} e sia f rappresentata graficamente

cane	“♣ ♣♦♦”
gatto	“♣ ♣♣”
aquila	“♦♦”
	“♦”

- f non è ambigua ma è ridondante

Codifica, decodifica

- Sia D={cane, gatto, aquila} e L={"♣ ♣♦♦", "♣ ♣ ♣"} e sia f rappresentata graficamente

cane
gatto
aquila

"♣ ♣♦♦"
"♣ ♣♣"

- f non è ridondante ma è ambigua

Codifica, decodifica

- Sia D={cane, gatto, aquila} e L={“♣ ♣♦♦”, “♣ ♣ ♣”, “♦♦”} e sia f rappresentata graficamente

cane → “♣♣♣”
gatto → “♣ ♣♦♦”
aquila → “♦♦”

- f non è ambigua e non è ridondante

Codifica, decodifica

- Sia ND la cardinalità di D e NL la cardinalità di L
  - Se $ND > NL$ qualsiasi codifica $f: D \rightarrow L$ è ambigua
  - Se $ND < NL$ qualsiasi codifica $f: D \rightarrow L$ è ridondante
- Inoltre è bene notare che
  - Non ambiguo significa che f è iniettiva (ad elementi distinti di D corrispondono elementi distinti di L)
  - Non ridondante significa che f è suriettiva (ogni elemento di L è la codifica di almeno un elemento di D)
  - Quindi se f è non ambigua e non ridondate allora f è iniettiva e suriettiva, quindi è una funzione biunivoca
- Una funzione $g: L \rightarrow D$ è invece detta una decodifica di D
- Se f è non ambigua (ovvero iniettiva) allora è invertibile. Quindi, induce naturalmente una decodifica $g=f^{-1}$

Proprietà delle codifiche

- Abbiamo già visto
  - ambigua/non ambigua
  - ridondante/non ridondante
- Altre proprietà
  - Economicità: numero di simboli utilizzati per unità di informazione
  - Semplicità nell'operazione di codifica e decodifica
  - Semplicità nell'eseguire operazioni sull'informazione codificata

Rappresentazione dei numeri naturali

- Occupiamoci ora delle possibili codifiche dei numeri naturali 0,1,2,…
- Un codice ovvio è dato dal sistema di numerazione decimale basato su A={'0',…,'9'}.
- Tale sistema è un sistema posizionale ovvero ad ogni cifra di una parola è assegnato un peso differente a seconda della posizione nella sequenza. Ad esempio dato "4456"
  - '6' rappresenta sei unità (cifra meno significativa)
  - '5' rappresenta cinque decine
  - '4' rappresenta quattro centinaia
  - '4' rappresenta quattro migliaia (cifra più significativa)

Il sistema numerico decimale è un sistema di tipo posizionale ovvero:

Le cifre che compongono un numero cambiano il loro valore secondo la posizione che occupano

7237 (settemiladuecentotrentasette) in base 10

$7x10^3$ +	$2x10^2$ +	$3x10^1$	+ $7x10^0$	
7x1000+	2x100 +	3x10	+ 7x1	
7000 +	200 +	30	+ 7	= **7237**

colonna del 1000
colonna del 100
colonna del 10
colonna dell'1

$$9742_{10} = 9 \times 10^3 + 7 \times 10^2 + 4 \times 10^1 + 2 \times 10^0$$

nove	sette	quattro	due
1000	100	10	1

Rappresentazione in base 10

- Decomponendo in potenze di 10, il numerale 1024 rappresenta il numero:

$$1 \times 10^3 + 0 \times 10^2 + 2 \times 10^1 + 4 \times 10^0$$

- Generalizzando, un numerale $c_{m-1}c_{m-2}\cdots c_0$ rappresenta

$$\sum_{i=0}^{m-1} c_i \cdot 10^i$$

- Il sistema decimale è quindi una codifica posizionale su base 10. Tuttavia non è l'unica, ad esempio i babilonesi utilizzavano un sistema di numerazione su base 60 (sessagesimale)

Number Systems

- Decimal numbers

1000's column
100's column
10's column
1's column

$$5374_{10} =$$

- Binary numbers

8's column
4's column
2's column
1's column

$$1101_2 =$$

Number Systems

- Decimal numbers

1000's column
100's column
10's column
1's column

$$5374_{10} = \underbrace{5 \times 10^3}_{\text{five thousands}} + \underbrace{3 \times 10^2}_{\text{three hundreds}} + \underbrace{7 \times 10^1}_{\text{seven tens}} + \underbrace{4 \times 10^0}_{\text{four ones}}$$

- Binary numbers

8's column
4's column
2's column
1's column

$$1101_2 = \underbrace{1 \times 2^3}_{\text{one eight}} + \underbrace{1 \times 2^2}_{\text{one four}} + \underbrace{0 \times 2^1}_{\text{no two}} + \underbrace{1 \times 2^0}_{\text{one one}} = 13_{10}$$

Powers of Two

- $2^0 =$
- $2^1 =$
- $2^2 =$
- $2^3 =$
- $2^4 =$
- $2^5 =$
- $2^6 =$
- $2^7 =$
- $2^8 =$
- $2^9 =$
- $2^{10} =$
- $2^{11} =$
- $2^{12} =$
- $2^{13} =$
- $2^{14} =$
- $2^{15} =$

Powers of Two

- $2^0 = 1$
- $2^1 = 2$
- $2^2 = 4$
- $2^3 = 8$
- $2^4 = 16$
- $2^5 = 32$
- $2^6 = 64$
- $2^7 = 128$
- $2^8 = 256$
- $2^9 = 512$
- $2^{10} = 1024$
- $2^{11} = 2048$
- $2^{12} = 4096$
- $2^{13} = 8192$
- $2^{14} = 16384$
- $2^{15} = 32768$

Rappresentazione

- Il sistema **decimale** utilizza **dieci** simboli per rappresentare un numero

0 1 2 3 4 5 6 7 8 9

- Il sistema **binario** utilizza **due** simboli

0 1

- Il sistema **ottale** utilizza **otto** simboli

0 1 2 3 4 5 6 7

- Il sistema **esadecimale** utilizza **sedici** simboli

0 1 2 3 4 5 6 7 8 9 A B C D E F

Rappresentazione in una base generica

- Ad ogni naturale b>1 corrisponde una codifica in base b.
- L'alfabeto A_b consiste in b simboli distinti che corrispondono ai numeri 0,1,…, b-1.

- Analogamente al sistema decimale, un numerale di cifre di A_b rappresenta il numero $s_{m-1} \cdots s_0$

$$\sum_{i=0}^{m-1} s_i \cdot b^i$$

$$587_{10} = 5 \times 10^2 + 8 \times 10^1 + 7 \times 10^0$$

Consideriamo ad esempio il sistema ottale (base 8).
A_8 consiste nelle cifre '0','1',...,'7'

$$13 \xrightarrow[8]{} 1 \cdot 8^1 + 3 \cdot 8^0 = 8 + 3 = 11$$

$$5201 \xrightarrow[8]{} 5{\cdot}8^3{+}2{\cdot}8^2{+}0{\cdot}8^1{+}1{\cdot}8^0 = 5{\cdot}8^3{+}2{\cdot}8^2{+}1{\cdot}8^0 = 2560{+}128{+}1 = 2689$$

Rappresentazione in una base generica

A questo punto potrebbero sorgere delle ambiguità. Se qualcuno vi dicesse: 11 è un numero pari. Pensereste che sia impazzito. Lui potrebbe ribattere: certo! infatti in base 5

$$11 \longrightarrow 1 \cdot 5^1 + 1 \cdot 5^0 = 5 + 1 = 6$$

E' chiaro che bisogna mettersi d'accordo su quale sistema di numerazione si adotta di volta in volta. Per questo useremo delle notazioni standard:

- **n** denota un (generico) numero naturale, a prescindere dalla sua rappresentazione
- $\mathbf{n_b}$ denota un naturale rappresentato in base b
- Una **sequenza di cifre decimali** rappresenta un particolare naturale espresso in base dieci. Ad esempio 236 denota il numero duecentotrentasei
- Una **sequenza di cifre seguite da un pedice b** rappresenta un numero espresso in base b.
- esempio:

$$1001_2 = 2^3 + 1 = 9$$

Basi maggiori di 10

- Per basi b < 10 possiamo chiaramente ri-usare le usuali cifre '0',…,'9'. Ad esempio, la codifica in base 6 utilizza le cifre '0',…,'5' mentre quella in base 3 le cifre '0', '1' e '2' e così via.
- E per basi b>10?
  - Si prendono in prestito le lettere dell'alfabeto
  - Per b=16 le cifre adottate sono '0',…,'9','a',…,'f', dove:

$$\begin{array}{ll} a_{16} = 10 & b_{16} = 11 \\ c_{16} = 12 & d_{16} = 13 \\ e_{16} = 14 & f_{16} = 15 \end{array}$$

Esempio:

$$b3c_{16} = 11 \cdot 16^2 + 3 \cdot 16^1 + 12 \cdot 16^0 = 2816 + 48 + 12 = 2876$$

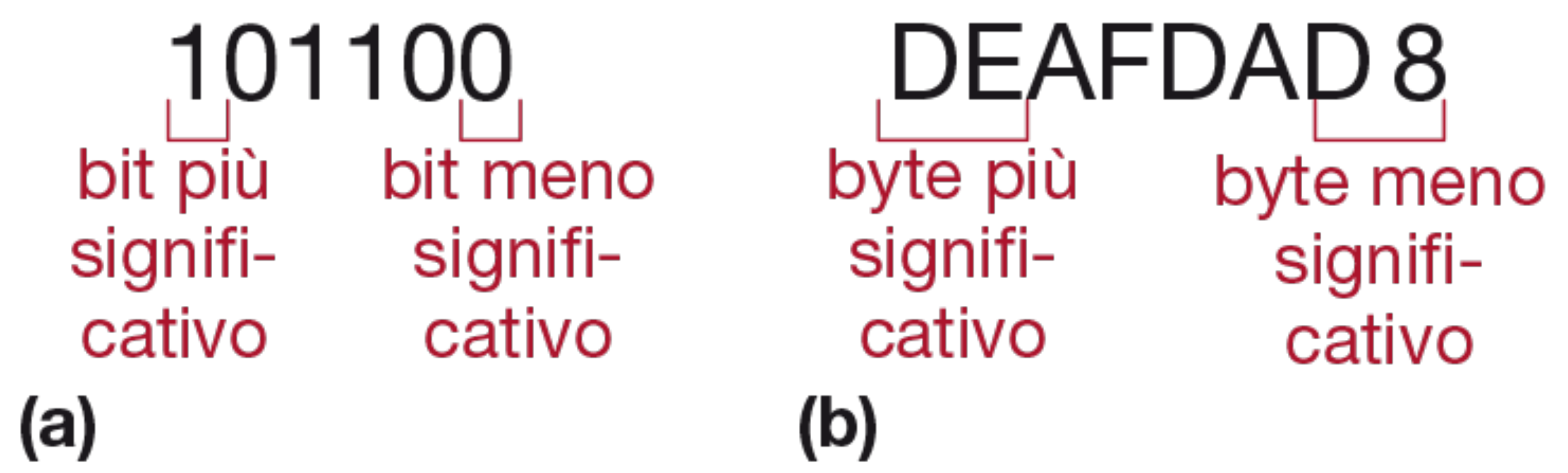
colonna del 256

colonna del 16

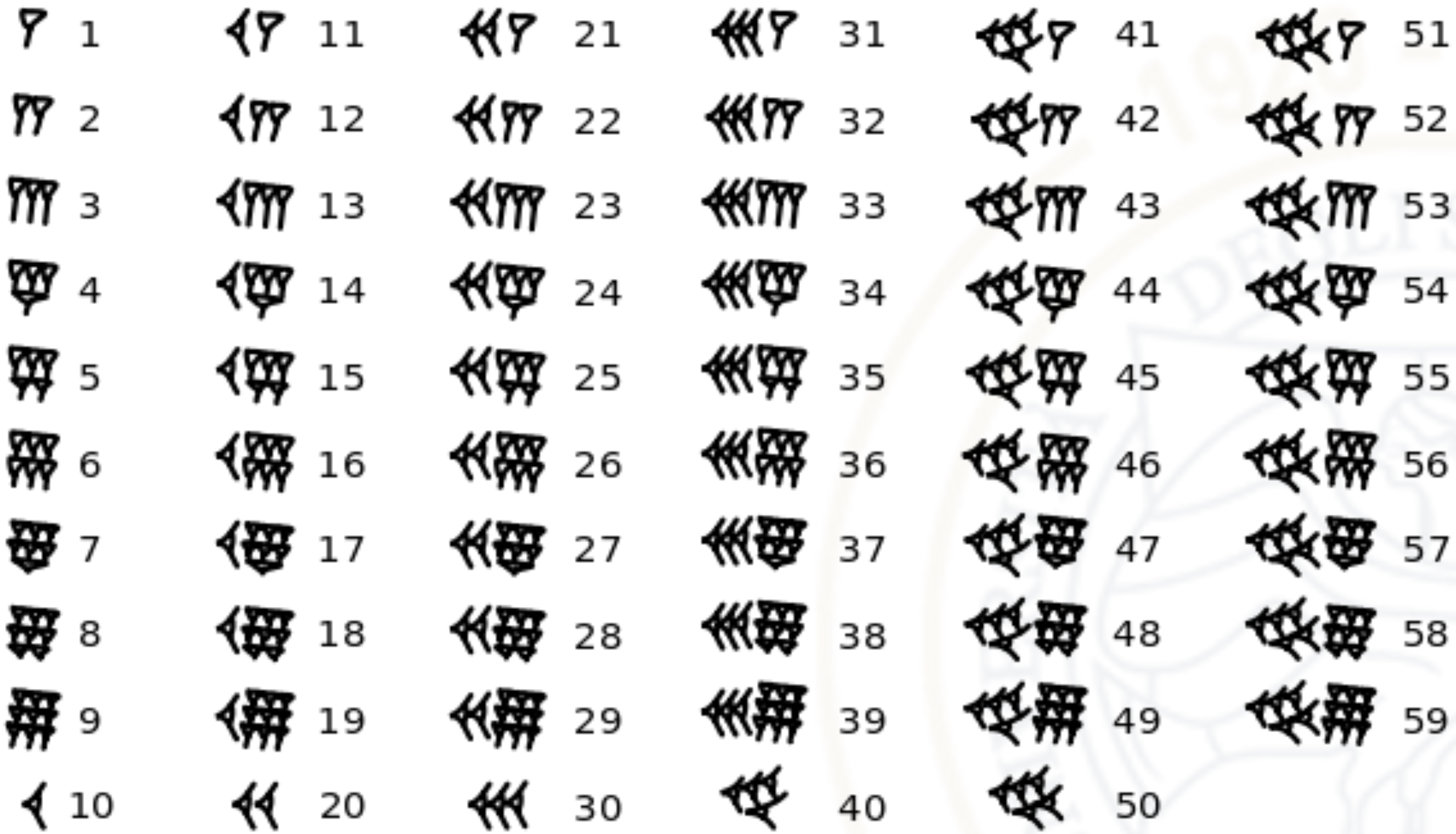
colonna dell'1

$$2ED_{16} = 2 \times 16^2 + E \times 16^1 + D \times 16^0 = 749_{10}$$

due	quattordici	tredici
256	16	1


101100
bit più significativo
bit meno significativo
(a)
DEAFDAD8
byte più significativo
byte meno significativo
(b)

Basi maggiori di 10



Lunghezza di n rispetto ad una base

- Chiamiamo lunghezza di n rispetto a b il numero di cifre che occorrono per rappresentare n in base b

  - la lunghezza di 101 rispetto a 2 è 7: $1100101_2=101$
  - la lunghezza di 101 rispetto a 10 è 3: $101_{10} = 101$
  - la lunghezza di 101 rispetto a 16 è 2: $65_{16}=101$

- La lunghezza di un numerale decresce al crescere della base di codifica

Valore massimo rappresentabile

Riferendoci alla base 10

- Con 1 cifra rappresentiamo i numeri da 0 a 9
- Con 2 cifre i numeri da 0 a 99
- Con 3 cifre i numeri da 0 a 999
- Con m cifre i numero da 0 a 10^m-1

Quindi se indichiamo con v_{max} il maggior numero rappresentabile con m cifre in base 10 abbiamo $$v_{max} = 10^m - 1$$

Valore massimo rappresentabile

- E per una base diversa da 10 quale è il massimo valore rappresentabile con m cifre?
- Consideriamo, ad esempio, il caso b=2 e m=4, il massimo valore rappresentabile corrisponde al numerale 1111

$$1111_2 = 2^3 + 2^2 + 2^1 + 2^0 = 15 = 2^4 - 1$$

- Per una generico b e m il maggior numero rappresentabile si ottiene concatenando m volte la cifra "più alta" (b-1). Quindi:

$$v_{max} = (b-1)b^{m-1} + (b-1)b^{m-2} + \ldots + (b-1)b^1 + (b-1)b^0 =$$

$$= (b \cdot b^{m-1} - b^{m-1}) + (b \cdot b^{m-2} - b^{m-2}) + \ldots + (b \cdot b^1 - b^1) + (b \cdot b^0 - b^0) =$$

$$= b^m - b^{m-1} + b^{m-1} - b^{m-2} + \ldots + b^2 - b + b - 1 = b^m - 1$$

Cifre necessarie per rappresentare n

$$b^m - 1 \geq n$$
$$b^m \geq n + 1$$
$$m \geq \log_b(n+1)$$

- Nella slide precedente avevamo il numero di cifre m e volevamo sapere quale è il *massimo numero rappresentabile* in una certa base b.
- Consideriamo il problema inverso: abbiamo un valore n e ci chiediamo quante *cifre m occorrono per rappresentarlo*.
- Chiaramente il massimo numero rappresentabile con m cifre dovrà essere maggiore o uguale a n

$$m \geq \log_b(n+1)$$

- In particolare cerchiamo il più piccolo m tale che

$$m = \lceil \log_b(n+1) \rceil$$

Binary Values and Range

- ***N*-digit decimal number**
  - How many values? **10^N**
  - Range? **$[0, 10^N - 1]$**
  - Example: 3-digit decimal number:
    - **$10^3 = 1000$ possible values**
    - **Range: [0, 999]**

- ***N*-bit binary number**
  - How many values? **2^N**
  - Range: **$[0, 2^N - 1]$**
  - Example: 3-digit binary number:
    - **$2^3 = 8$ possible values**
    - **Range: $[0, 7] = [000_2$ to $111_2]$**

Digressione: base unaria

- Ricordate che quando abbiamo introdotto i sistemi di numerazione abbiamo assunto la base $b \geq 2$.
- Perché non è possibile considerare una numerazione unaria? Certo! Ma ci sono delle controindicazioni…
    - La base unaria consta di una solo cifra I (detta in gergo matematico *mazzarella*
    - Una mazzarella rappresenta 0, due mazzarelle 1, …, n+1 mazzarelle rappresentano n
    - $IIIIII_1=5$
    - Notate che a prescindere dalla posizione ogni mazzarella vale 1, quindi questo sistema non è posizionale
    - Non potrebbe essere altrimenti visto che 1 elevato ad una qualsiasi potenza è sempre uguale a 1!