



Università degli studi Roma Tre



La Rappresentazione dell'Informazione

A cura di A. Orlandini

Obiettivi

Completata questa unità saremo in grado di:

- Definire il significato di rappresentazione dei numeri
- Distinguere le varie tipologie di codifica
- Definire un sistema di numerazione
- Elencare vari sistemi di numerazione
- Descrivere le regole dei cambiamenti di base
- Descrivere l'aritmetica modulo k
- Descrivere la rappresentazione dei numeri relativi

Contenuti

- La Codifica dell'informazione
- I sistemi di numerazione
- I cambiamenti di base
- Rappresentazione modulo k
- Rappresentazione dei numeri con segno

Libri di testo di riferimento

- ❑ Batini, Carlucci Aiello, Lenzerini, Marchetti Spaccamela, Miola (1997). Fondamenti di Programmazione dei Calcolatori Elettronici. Ed. Franco Angeli.
- ❑ L. Cabibbo (2004). Fondamenti di Informatica: oggetti e Java. Mc Graw-Hill.



La Codifica dell'informazione

L'informazione all'interno del calcolatore

- La rappresentazione dell'informazione all'interno di un calcolatore è condizionata da due vincoli
 1. il calcolatore gestisce **solo sequenze di bit**
 - con i dispositivi elettronici è facile rappresentare le due cifre 0 e 1
 2. il calcolatore ha una **memoria limitata**
 - ne segue che la quantità di memoria dedicata alla rappresentazione di una singola informazione è anch'essa limitata
- Questi due vincoli sono largamente indipendenti tra loro
 - anche se il calcolatore avesse una memoria infinita dovremmo comunque studiare il modo di rappresentare i dati in binario
 - anche se il calcolatore potesse memorizzare direttamente informazioni generiche (per esempio numeri decimali) avremmo comunque il problema della limitatezza della sua memoria

Contenuti

- La rappresentazione binaria dell'informazione
 - la codifica tabellare
 - la rappresentazione di caratteri
 - la rappresentazione di valori booleani (vero e falso)
 - la conversione di base
 - il sistema di numerazione decimale
 - il sistema di numerazione binario

- Le rappresentazione dell'informazione con memoria limitata
 - i numeri naturali
0, 1, 2, 3, ...
 - i numeri interi relativi
..., -3, -2, -1, 0, +1, +2, +3, ...
 - i numeri razionali
0.0018, -2345.06, 12.0, ...

Codifica dei dati – il bit

- ❑ In un calcolatore i dati e le istruzioni sono codificati in forma binaria, ovvero come sequenze finite di cifre 0 e 1
 - il **bit** è il più piccolo dato in un calcolatore
 - un bit può avere valore 0 oppure 1
 - la parola “**bit**” è una forma contratta per **binary digit** (cifra binaria)
 - ciascun **bit** è memorizzato da una cella elementare di memoria, fisicamente realizzata come **dispositivo elettronico bistabile** (in cui sono chiaramente distinguibili due stati)
 - questi due stati vengono fatti corrispondere allo 0 e all'1
- ❑ Un bit è quindi un dato che può essere usato soltanto per rappresentare una informazione binaria
- ❑ Per rappresentare altre tipologie di informazioni sono necessarie **sequenze di bit**

Codifica dei dati – il byte

- ❑ Il byte è la più piccola sequenza di bit
 - un **byte** è una sequenza di 8 bit
- ❑ Le possibili combinazioni degli 8 bit in un byte sono $2^8 = 256$
 - un byte può essere utilizzato per rappresentare un valore tra 256 diversi possibili valori
 - ad esempio, un piccolo numero intero, un carattere in un alfabeto che contiene non più di 256 caratteri, ...
 - in generale, sono possibili diverse scelte sull'insieme dei valori possibili
 - ad esempio, con un byte si può rappresentare un numero naturale compreso nell'intervallo da 0 a 255, oppure un numero intero relativo nell'intervallo da –128 a 127
- ❑ Per rappresentare altre tipologie di informazioni sono necessarie sequenze di bit più grandi

Strategie per la codifica dei dati

- Nel codificare una informazione in un dato binario possiamo usare due diverse strategie
 - **codifica tabellare**
 - applicabile quando l'insieme dei valori che può assumere l'informazione è finito
 - si fa corrispondere ad ogni valore dell'informazione una diversa combinazione di 0 e di 1
 - **conversione di base**
 - applicabile quando l'informazione è un numero in base 10
 - si premette una fase di traduzione in cui il numero viene convertito in base 2

Codifica tabellare

- ❑ Quanti bit dobbiamo usare per codificare m valori diversi?
- ❑ Abbiamo bisogno di almeno m combinazioni diverse
 - i matematici non le chiamano “combinazioni”, ma “disposizioni con ripetizione”
- ❑ Con un bit abbiamo due sole combinazioni
 - 0 e 1
 - $2 = 2^1$
- ❑ Con due bit abbiamo quattro combinazioni diverse
 - 00, 01, 10 e 11
 - $4 = 2^2$
- ❑ Con tre bit abbiamo otto combinazioni diverse
 - 000, 001, 010, 011, 100, 101, 110 e 111
 - $8 = 2^3$
- ❑ In generale con n bit avremo 2^n combinazioni diverse

2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
(1)	(2)	(4)	(8)	(16)	(32)	(64)	(128)	(256)	(512)	(1024)

Rappresentazione di caratteri

- ❑ Sono disponibili diverse codifiche tabellari
- ❑ Le principali sono le seguenti
 - **ASCII**
 - sta per *American Standard Code for Information Interchange*
 - nella prima versione utilizzava 7 bit per codificare un carattere
 - nella versione estesa utilizza un byte per rappresentare un carattere
 - **UNICODE**
 - utilizza due byte per rappresentare un carattere
 - coincide con la codifica ASCII per i caratteri “di base” dell’alfabeto inglese
 - rappresenta 49.194 caratteri, negli alfabeti delle lingue principali (italiano, inglese, cirillico, arabo, ebraico, cinese, giapponese, ecc. ecc.)
 - è molto diffuso (formati HTML e XML, linguaggio Java, varie applicazioni, ecc.)

Codifica ASCII e UNICODE

Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char
00000000	0	Null	00100000	32	Spc	01000000	64	@	01100000	96	`
00000001	1	Start of heading	00100001	33	!	01000001	65	A	01100001	97	a
00000010	2	Start of text	00100010	34	"	01000010	66	B	01100010	98	b
00000011	3	End of text	00100011	35	#	01000011	67	C	01100011	99	c
00000100	4	End of transmit	00100100	36	\$	01000100	68	D	01100100	100	d
00000101	5	Enquiry	00100101	37	%	01000101	69	E	01100101	101	e
00000110	6	Acknowledge	00100110	38	&	01000110	70	F	01100110	102	f
00000111	7	Audible bell	00100111	39	'	01000111	71	G	01100111	103	g
00001000	8	Backspace	00101000	40	(01001000	72	H	01101000	104	h
00001001	9	Horizontal tab	00101001	41)	01001001	73	I	01101001	105	i
00001010	10	Line feed	00101010	42	*	01001010	74	J	01101010	106	j
00001011	11	Vertical tab	00101011	43	+	01001011	75	K	01101011	107	k
00001100	12	Form Feed	00101100	44	,	01001100	76	L	01101100	108	l
00001101	13	Carriage return	00101101	45	-	01001101	77	M	01101101	109	m
00001110	14	Shift out	00101110	46	.	01001110	78	N	01101110	110	n
00001111	15	Shift in	00101111	47	/	01001111	79	O	01101111	111	o
00010000	16	Data link escape	00110000	48	0	01010000	80	P	01110000	112	p
00010001	17	Device control 1	00110001	49	1	01010001	81	Q	01110001	113	q
00010010	18	Device control 2	00110010	50	2	01010010	82	R	01110010	114	r
00010011	19	Device control 3	00110011	51	3	01010011	83	S	01110011	115	s
00010100	20	Device control 4	00110100	52	4	01010100	84	T	01110100	116	t
00010101	21	Neg. acknowledge	00110101	53	5	01010101	85	U	01110101	117	u
00010110	22	Synchronous idle	00110110	54	6	01010110	86	V	01110110	118	v
00010111	23	End trans. block	00110111	55	7	01010111	87	W	01110111	119	w
00011000	24	Cancel	00111000	56	8	01011000	88	X	01111000	120	x
00011001	25	End of medium	00111001	57	9	01011001	89	Y	01111001	121	y
00011010	26	Substitution	00111010	58	:	01011010	90	Z	01111010	122	z
00011011	27	Escape	00111011	59	;	01011011	91	[01111011	123	{
00011100	28	File separator	00111100	60	<	01011100	92	\	01111100	124	
00011101	29	Group separator	00111101	61	=	01011101	93]	01111101	125	}
00011110	30	Record Separator	00111110	62	>	01011110	94	^	01111110	126	~
00011111	31	Unit separator	00111111	63	?	01011111	95	_	01111111	127	Del

Codifica ASCII e UNICODE – estensione

Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char	Byte	Cod.	Char
10000000	128	Ç	10100000	160	á	11000000	192	+	11100000	224	Ó
10000001	129	ü	10100001	161	í	11000001	193	-	11100001	225	ß
10000010	130	é	10100010	162	ó	11000010	194	-	11100010	226	Ô
10000011	131	â	10100011	163	û	11000011	195	+	11100011	227	Ò
10000100	132	ä	10100100	164	ñ	11000100	196	-	11100100	228	ö
10000101	133	à	10100101	165	Ñ	11000101	197	+	11100101	229	Ö
10000110	134	á	10100110	166	ª	11000110	198	ä	11100110	230	µ
10000111	135	ç	10100111	167	•	11000111	199	Ä	11100111	231	þ
10001000	136	ê	10101000	168	¿	11001000	200	+	11101000	232	Ð
10001001	137	ë	10101001	169	®	11001001	201	+	11101001	233	Ú
10001010	138	e	10101010	170	¬	11001010	202	-	11101010	234	Û
10001011	139	i	10101011	171	½	11001011	203	-	11101011	235	Ü
10001100	140	î	10101100	172	¼	11001100	204	-	11101100	236	ý
10001101	141	ï	10101101	173	í	11001101	205	-	11101101	237	ÿ
10001110	142	Ä	10101110	174	«	11001110	206	+	11101110	238	—
10001111	143	Å	10101111	175	»	11001111	207	ó	11101111	239	·
10010000	144	É	10110000	176	-	11010000	208	ö	11110000	240	-
10010001	145	æ	10110001	177	-	11010001	209	Ð	11110001	241	±
10010010	146	Æ	10110010	178	-	11010010	210	Ê	11110010	242	-
10010011	147	ô	10110011	179	-	11010011	211	Ë	11110011	243	¾
10010100	148	õ	10110100	180	-	11010100	212	È	11110100	244	¶
10010101	149	ö	10110101	181	À	11010101	213	É	11110101	245	§
10010110	150	ù	10110110	182	Á	11010110	214	Î	11110110	246	+
10010111	151	ú	10110111	183	Â	11010111	215	Ï	11110111	247	-
10011000	152	ÿ	10111000	184	©	11011000	216	Ï	11111000	248	°
10011001	153	Û	10111001	185	-	11011001	217	+	11111001	249	°
10011010	154	Ü	10111010	186	-	11011010	218	+	11111010	250	°
10011011	155	σ	10111011	187	+	11011011	219	-	11111011	251	1
10011100	156	£	10111100	188	+	11011100	220	-	11111100	252	3
10011101	157	Ø	10111101	189	¢	11011101	221	-	11111101	253	2
10011110	158	×	10111110	190	¥	11011110	222	Ï	11111110	254	-
10011111	159	f	10111111	191	+	11011111	223	-	11111111	255	-

Rappresentazione di valori booleani

- ❑ Si usa una codifica tabellare
- ❑ Per rappresentare un valore booleano sarebbe sufficiente un solo bit
- ❑ Tipicamente, però, i sistemi informatici usano uno o più byte, in cui il valore zero rappresenta *false* e tutti gli altri valori rappresentano *true*
 - per esempio, su alcune piattaforme un booleano è rappresentato con 4 byte



I Sistemi di Numerazione

Conversione di base

- E' nostra consuetudine rappresentare i numeri naturali tramite il **sistema di numerazione posizionale**
 - i simboli che rappresentano il numero hanno un'interpretazione dipendente dalla loro posizione
- In particolare è largamente diffuso il sistema di numerazione in **base dieci (detto anche decimale)**
 - la numerazione in base dieci fa uso di dieci simboli
0, 1, 2, 3, 4, 5, 6, 7, 8 e 9
 - un numero è denotato da una sequenza di cifre
 - il valore di una cifra dipende dalla cifra stessa e dalla sua posizione nella sequenza
 - per esempio la sequenza di cifre **563** corrisponde al numero
 $5 \times 100 + 6 \times 10 + 3 = 563$

Sistema di numerazione in base dieci

$$10^2=100$$

×

$$10^1=10$$

×

$$10^0=1$$

×

la sequenza di cifre →

5

+

6

+

3

rappresenta il numero →

$$5 \times 100$$

$$+ 6 \times 10$$

$$+ 3 \times 1$$

$$500$$

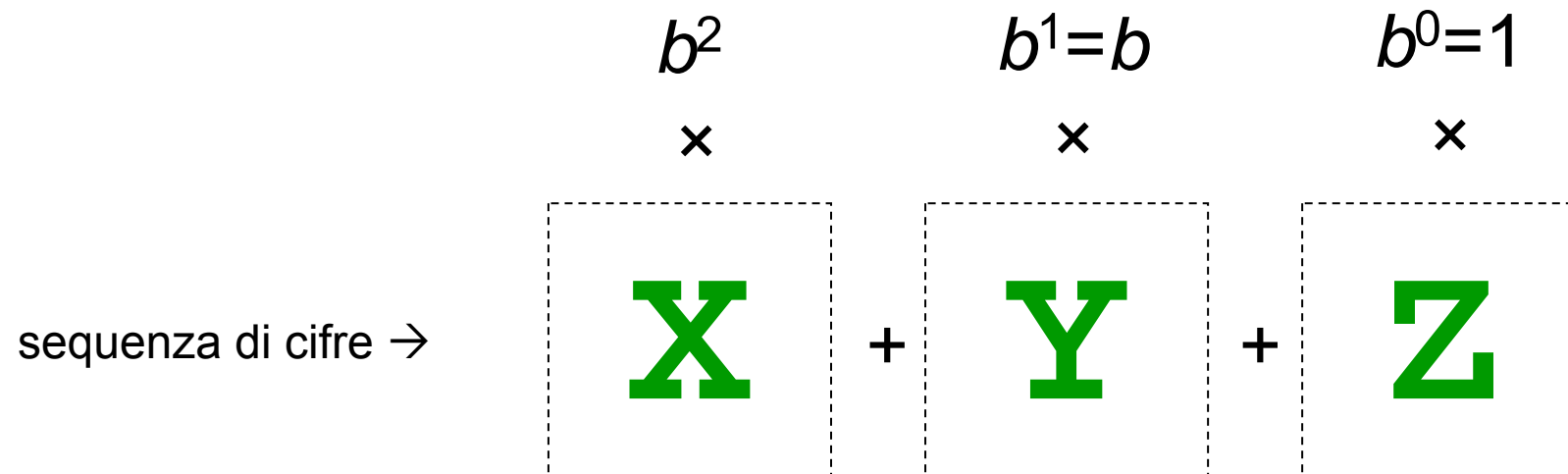
$$+ 60$$

$$+ 3$$

$$563$$

Sistema di numerazione in base qualsiasi

- Il sistema di numerazione posizionale può essere generalizzato a qualsiasi base b , disponendo di b simboli diversi



rappresenta il numero → $\mathbf{x} \times b^2 + \mathbf{y} \times b + \mathbf{z} \times 1$

le cifre **X**, **Y** e **Z** sono prese dall'insieme dei b simboli a disposizione

Basi di uso frequente

- ❑ **Sistema di numerazione in base dieci (o decimale)**
 - utilizza i simboli 0,1,2,3,4,5,6,7,8 e 9
 - adottato per gli ovvi rapporti con la rappresentazione tramite flessione delle dita (indigitazione)
- ❑ **Sistema di numerazione in base due (o binario)**
 - utilizza i simboli 0 e 1
 - adottato dai calcolatori elettronici per la facilità di rappresentare mediante grandezze elettromagnetiche due stati, corrispondenti ai due simboli
- ❑ **Sistema di numerazione in base sedici (o esadecimale)**
 - utilizza i simboli 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E ed F
 - adottato per la sua compattezza e per la facilità di conversione con la rappresentazione binaria
- ❑ **Sistema di numerazione in base uno (o unario)**
 - utilizza solo il simbolo I
 - non è veramente un sistema posizionale ($1^0 = 1^1 = 1^2 = 1^3 = \dots = 1$)
 - adottato talvolta perché il numero può essere incrementato facilmente con l'aggiunta di una cifra (esempio: III+I = IIII)

Sistema di numerazione in base due

- Il sistema di numerazione in base due (detto anche *binario*) fa uso dei due soli simboli 0 e 1

	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	(1024)	(512)	(256)	(128)	(64)	(32)	(16)	(8)	(4)	(2)	(1)
	x	x	x	x	x	x	x	x	x	x	x
...	P	Q	R	S	T	U	V	W	X	Y	Z

Operazioni con i numeri binari

□ Esempi di somma di interi

$$\begin{array}{r} 0 \ 1 \ 0 \ 0 \ 1 \ 1 \ 1 \ 0 \quad \text{riporti} \\ 0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ + \\ 0 \ 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ = \\ \hline 1 \ 0 \ 1 \ 1 \ 0 \ 1 \ 0 \ 1 \end{array}$$

$$\begin{array}{r} 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ 1 \quad \text{riporti} \\ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 1 \ 1 \ + \\ 1 \ 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ = \\ \hline 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0 \end{array}$$

Operazioni con i numeri binari

□ Esempio di sottrazione di interi

- lo sappiamo fare solo se il primo numero (minuendo) è maggiore del secondo numero (sottraendo)

$$\begin{array}{r} 0 \ 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \quad \text{prestiti} \\ 1 \ 0 \ 1 \ 0 \ 1 \ 1 \ 1 \ 0 \ - \\ 0 \ 1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1 \ = \\ \hline 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 1 \end{array}$$

Operazioni con i numeri binari

□ Esempi di moltiplicazione di interi

$$\begin{array}{r} 10110 \times \\ 101 = \\ \hline 10110 \\ 00000 \\ 10110 \\ \hline 1101110 \end{array}$$

$$\begin{array}{r} 10001 \times \\ 1100 = \\ \hline 00000 \\ 00000 \\ 10001 \\ \hline 10001 \\ \hline 11001100 \end{array}$$

Operazioni con i numeri binari

- ❑ Moltiplicare per 2^k equivale a traslare a sinistra di k posizioni

$$\begin{array}{r} 1\ 0\ 1\ 1\ 0\ \times \\ 1\ 0\ = \\ \hline 0\ 0\ 0\ 0\ 0 \\ 1\ 0\ 1\ 1\ 0 \\ \hline 1\ 0\ 1\ 1\ 0\ 0 \end{array}$$

$$\begin{array}{r} 1\ 0\ 1\ 1\ 0\ \times \\ 1\ 0\ 0\ = \\ \hline 0\ 0\ 0\ 0\ 0 \\ 0\ 0\ 0\ 0\ 0 \\ 1\ 0\ 1\ 1\ 0 \\ \hline 1\ 0\ 1\ 1\ 0\ 0\ 0 \end{array}$$

Operazioni con i numeri binari

□ Esempi di divisione tra interi

$$\begin{array}{r}
 \overline{1101110} \quad \overline{1100} \quad \overline{1110} \quad \overline{1100} \\
 1100 \\
 \hline
 1110 \\
 1100 \\
 \hline
 10
 \end{array}$$

riprova:

$$\begin{array}{r}
 1100 \times \\
 1001 = \\
 \hline
 1100 \\
 0000 \\
 0000 \\
 \hline
 1100 \\
 1101100 + \\
 10 \\
 \hline
 1101110
 \end{array}$$

Operazioni con i numeri binari

- Dividere per 2^k equivale a traslare a destra di k posizioni

Diagram illustrating binary division:

$$\begin{array}{r} 1101110 \quad | \quad 100 \\ \underline{100} \\ 101 \\ \underline{100} \\ 111 \\ \underline{100} \\ 110 \\ \underline{100} \\ 10 \end{array}$$

ne segue che, se X è un numero binario:

- l'ultima cifra di $X/2$ è la penultima cifra di X
- l'ultima cifra di $X/4$ è la terzultima cifra di X
- l'ultima cifra di $X/8$ è la quartultima cifra di X
- ecc

Esercizi sulle operazioni in binario

□ Calcolare i risultati delle seguenti operazioni

- $0100101 + 0101 =$

- $100001010 + 11111111 =$

- $101010010 - 1111000 =$

- $100001 - 11 =$

- $10010001 \times 111 =$

- $10101101 \times 1010 =$

- $100010101 \times 100000 =$

- $1010010 : 1000 =$ (con resto)

- $1110010 : 1101 =$ (con resto)

- $100101 : 110 =$ (con resto)



Cambiamenti di base

Conversione di un decimale in un binario

- Conversione del numero $X_{(10)}$ (decimale) in $X_{(2)}$ (binario)
 - supponiamo che $X_{(2)}$ sia composto dalle cifre .. $x_4 x_3 x_2 x_1 x_0$
 - è immediato determinare se $X_{(10)}$ è pari o dispari
 - se $X_{(10)}$ è dispari anche $X_{(2)}$ è dispari
 - ne segue che $x_0 = 1$
 - se $X_{(10)}$ è pari anche $X_{(2)}$ è pari
 - ne segue che $x_0 = 0$
 - immaginiamo di dividere $X_{(10)}$ e $X_{(2)}$ per due, scartando il resto (che è proprio x_0)
 - se $X_{(10)}/2$ è dispari anche $X_{(2)}/2$ è dispari
 - l'ultima cifra di $X_{(2)}/2$ è 1 \Rightarrow la penultima cifra di $X_{(2)}$ è 1 $\Rightarrow x_1 = 1$
 - se $X_{(10)}/2$ è pari anche $X_{(2)}/2$ è pari
 - l'ultima cifra di $X_{(2)}/2$ è 0 \Rightarrow la penultima cifra di $X_{(2)}$ è 0 $\Rightarrow x_1 = 0$
 - iteriamo il procedimento finché non otteniamo tutte le cifre di $X_{(2)}$

Conversione di un decimale in un binario

□ Esempio: $X_{(10)} = 75$; $X_{(2)} = ?$

- $75 : 2 = 37$ con resto 1 ($x_0 = 1$)
- $37 : 2 = 18$ con resto 1 ($x_1 = 1$)
- $18 : 2 = 9$ con resto 0 ($x_2 = 0$)
- $9 : 2 = 4$ con resto 1 ($x_3 = 1$)
- $4 : 2 = 2$ con resto 0 ($x_4 = 0$)
- $2 : 2 = 1$ con resto 0 ($x_5 = 0$)
- $1 : 2 = 0$ con resto 1 ($x_6 = 1$)

□ $X_{(2)} = x_6 x_5 x_4 x_3 x_2 x_1 x_0 = 1001011$

Conversione da un binario ad un decimale

□ Conversione di un numero $X_{(2)}$ (binario) in $X_{(10)}$ (decimale)

- supponiamo che $X_{(2)}$ sia composto dalle cifre .. $x_4 x_3 x_2 x_1 x_0$
- applichiamo direttamente la formula:
$$X_{(10)} = x_0 \cdot 2^0 + x_1 \cdot 2^1 + x_2 \cdot 2^2 + x_3 \cdot 2^3 \dots$$

□ Esempio: $X_{(2)} = 10011$; $X_{(10)} = ?$

- $$X_{(10)} = 1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 0 \cdot 2^3 + 1 \cdot 2^4$$
$$= 1 + 2 + 16 = 19$$

□ Altro esempio: $X_{(2)} = 11010$; $X_{(10)} = ?$

- $$X_{(10)} = 0 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \cdot 2^3 + 1 \cdot 2^4$$
$$= 2 + 8 + 16 = 26$$

Metodi di conversione veloci

□ Ordine di grandezza di un numero binario

- si può ottenere ricordando che $2^{10} = 1024 \approx 1000 = 10^3$
 - per esempio $1101001010001010110010_{(2)}$
 - può essere così suddiviso 11 0100101000 1010110010
 - il suo ordine di grandezza è dunque $3 \times 1000 \times 1000 = 3\,000\,000_{(10)}$
 - il suo valore effettivo è $3\,449\,522_{(10)}$

□ Numeri ottenibili per sottrazione o somma

- alcuni numeri possono essere ottenuti per sottrazione o somma di numeri notevoli il cui equivalente decimale è noto o facile da calcolare
- esempio
 - per i numeri decimali $999_{(10)} = 1000_{(10)} - 1_{(10)}$
 - analogamente, per i numeri binari $1111_{(2)} = 10000_{(2)} - 1_{(2)}$
 - dunque $1111_{(2)} = 2^4 - 1 = 16 - 1 = 15_{(10)}$



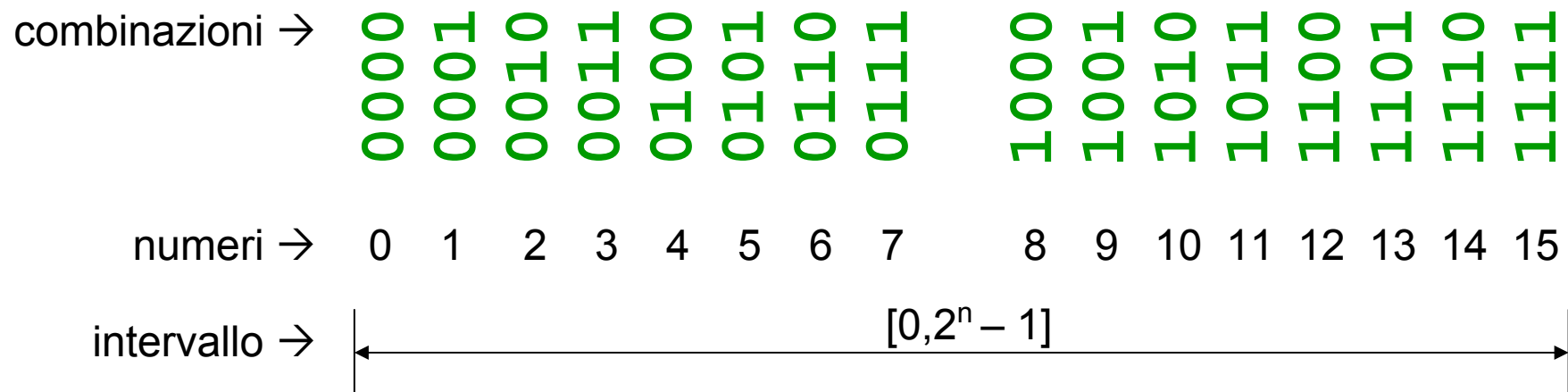
*Rappresentazione con memoria
limitata*

Rappresentazione con memoria limitata

- Quando la memoria è limitata e l'insieme dei valori da rappresentare è infinito è possibile solo una rappresentazione parziale dell'informazione
 - cioè: non tutti i valori possono essere rappresentati
- Nel seguito supporremo di avere a disposizione n bit con i quali vogliamo rappresentare i seguenti dati
 - un numero naturale (0, 1, 2, ...)
 - un numero relativo (... , -3, -2, -1, 0, +1, +2, +3, ...)
 - un numero razionale (0.0018, 2345.06, 12.0, ...)

Rappresentazione binaria di numeri naturali

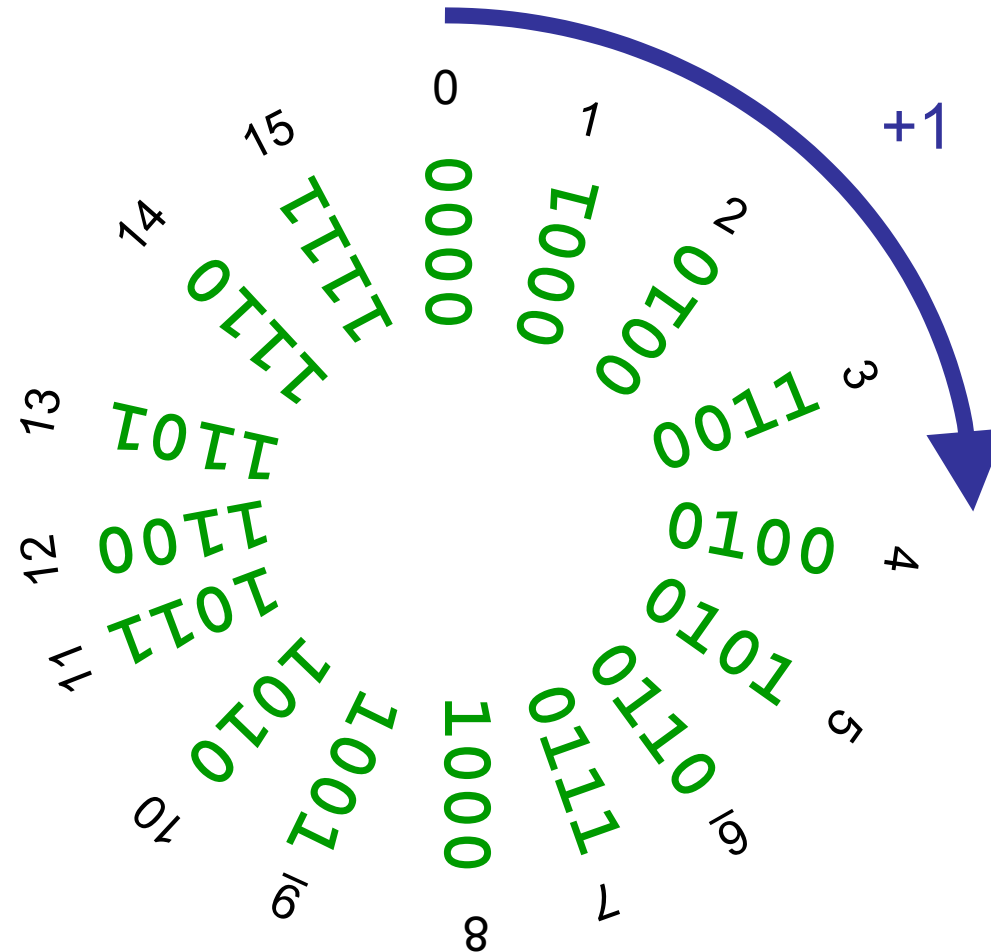
- Un numero naturale (0, 1, 2, ...) può essere rappresentato dalla sua notazione in binario
- Quanti numeri possiamo rappresentare con n cifre in base due?
 - abbiamo 2^n combinazioni diverse che possono essere utilizzate per rappresentare i numeri da zero a $2^n - 1$
 - esempio con 4 cifre (n=4)



Note sulla rappresentazione dei naturali

- Avendo un numero limitato di bit per la rappresentazione dei numeri naturali non è possibile rappresentare numeri arbitrariamente grandi
 - il numero più grande che possiamo rappresentare con n bit è $2^n - 1$
- Che numero otteniamo se sommiamo 1 a $2^n - 1$?

Circolarità della rappresentazione





*Rappresentazione dei numeri con
segno*

Rappresentazione di interi relativi

- Nella rappresentazione dei numeri relativi occorre destinare alcune combinazioni di bit alla rappresentazione dei numeri negativi
 - Un sistema intuitivo è quello di rappresentare esplicitamente segno e modulo
 - questa rappresentazione è infatti chiamata “modulo-segno”
 - Il sistema più utilizzato è però quello di usare il complemento alla base
 - in questo caso il complemento a due

Rappresentazione modulo-segno

Il primo bit della sequenza rappresenta il segno

- 0 = numero positivo, 1 = numero negativo
- esempio con sequenze di quattro bit ($n=4$)

combinazioni →	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
segno →	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
numeri →	0	1	2	3	4	5	6	7	?	-1	-2	-3	-4	-5	-6	-7
intervalli →	-----[0, $2^{n-1} - 1$]-----								-----[-1, $-(2^{n-1} - 1)$]-----							

- ## Questa rappresentazione è di facile lettura, ma è difficile eseguire operazioni aritmetiche, perché il segno va trattato in maniera difforme dagli altri bit

Operazione di complementazione

□ Data una sequenza di k cifre rappresentante il numero N in base b , si definisce (algebra modulo k):

- **complemento alla base di N** il valore

$$C_b = b^k - N$$

- **complemento diminuito di N** il valore

$$C_d = C_b - 1$$

- segue, quindi, che

$$C_b = C_d + 1$$

Operazione di complementazione

- ❑ Nel sistema binario, C_b prende il nome di **complemento a due**, mentre C_d **complemento a uno**
- ❑ Osservazione: nel sistema binario, il complemento a uno di un numero si ottiene semplicemente sostituendo nella sequenza di bit gli 0 con 1 e gli 1 con 0
- ❑ N.B. valgono le seguenti proprietà

$$C_b(C_b(N)) = N$$

$$C_d(C_d(N)) = N$$

Operazione di complementazione

□ Esempi:

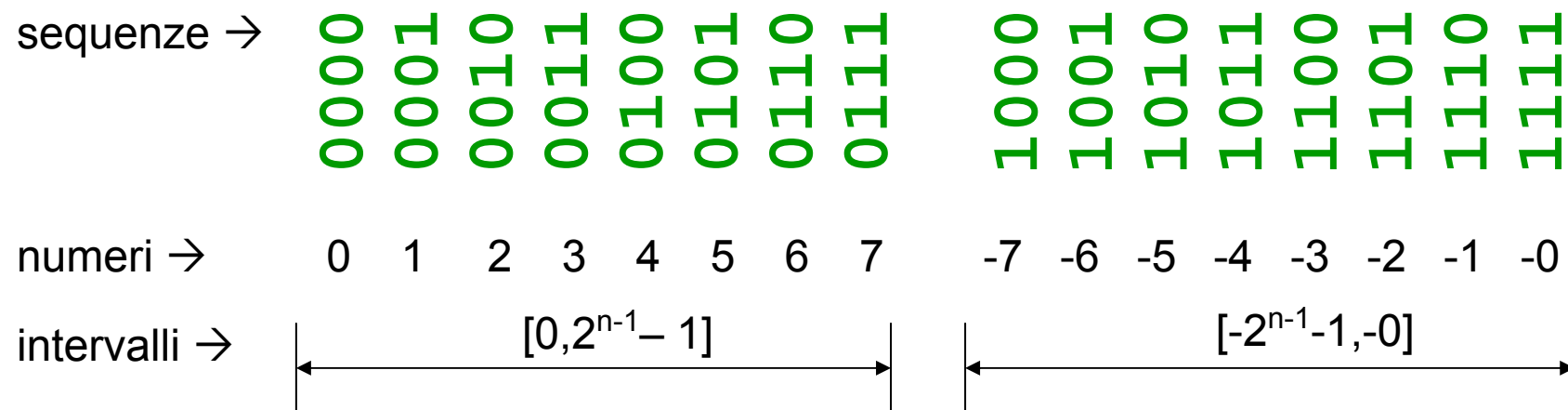
- $N = 001_{(2)}$
 - $C_d = 110_{(2)}$
 - $C_b = 110_{(2)} + 1 = 111_{(2)}$
- $N = 100_{(2)}$
 - $C_d = 011_{(2)}$
 - $C_b = 011_{(2)} + 1 = 100_{(2)}$
- $N = 000_{(2)}$
 - $C_d = 111_{(2)}$
 - $C_b = 111_{(2)} + 1 = 000_{(2)}$

Complemento a uno

- Data una sequenza che rappresenta un numero assoluto, si ottiene la sequenza che rappresenta il corrispondente numero:
 - **positivo**: usando la rappresentazione binaria tradizionale
 - **negativo**: complementando ad uno l'intera rappresentazione del corrispondente numero positivo

Complemento ad uno

- Gli intervalli rappresentati sono simmetrici ma con doppia rappresentazione dello zero



Esempi di complemento ad uno

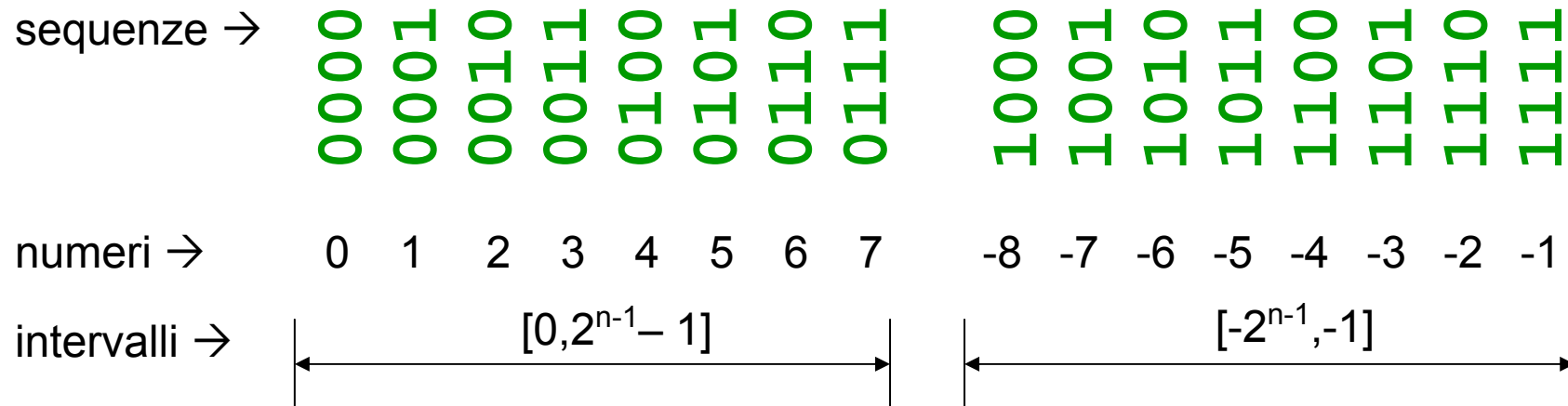
Assoluto	Positivo	Negativo
$0101_{(2)}$	$0101_{(2)}$	$1010_{(2)}$
$01011_{(2)}$	$01011_{(2)}$	$10100_{(2)}$
$0000_{(2)}$	$0000_{(2)}$	$1111_{(2)}$

Rappresentazione in complemento a due

- Nella rappresentazione in **complemento a due** i numeri negativi si ottengono dai corrispondenti numeri positivi complementando ogni cifra e sommando uno

- esempio

- $5 = 0101$
- $-5 = 1010 + 1 = 1011$



- La prima cifra continua a rappresentare il segno

Rappresentazione in complemento a due

- Metodo più veloce per ottenere la rappresentazione di un numero negativo da quella del corrispondente positivo:
 - partendo da destra si lasciano invariate tutte le cifre fino al primo uno (incluso) e poi si invertono tutti i seguenti
 - esempio
 - $N = 10011010110110000$
 - $10011010100110000 \leftarrow$ invariato fino al primo uno (incluso)
 - $01100101011010000 \leftarrow$ invertite tutte le cifre seguenti
 - $-N = 01100101011010000$

- L'operazione di cambiamento di segno è eseguibile tramite il complemento a due
 - esempio
 - $5 = 0101$
 - $-5 = 1011$ (ottenuto complementando a due il precedente)
 - $5 = 0101$ (ottenuto complementando a due il precedente)

Esempi di somme in complemento a due

□ $2 + 3 = 5$

$$\begin{array}{r} 2 + \\ 3 = \\ \hline 5 \end{array}$$

0 0 1 0 riporti

$$\begin{array}{r} 0010 + \\ 0011 = \\ \hline 0101 \end{array}$$

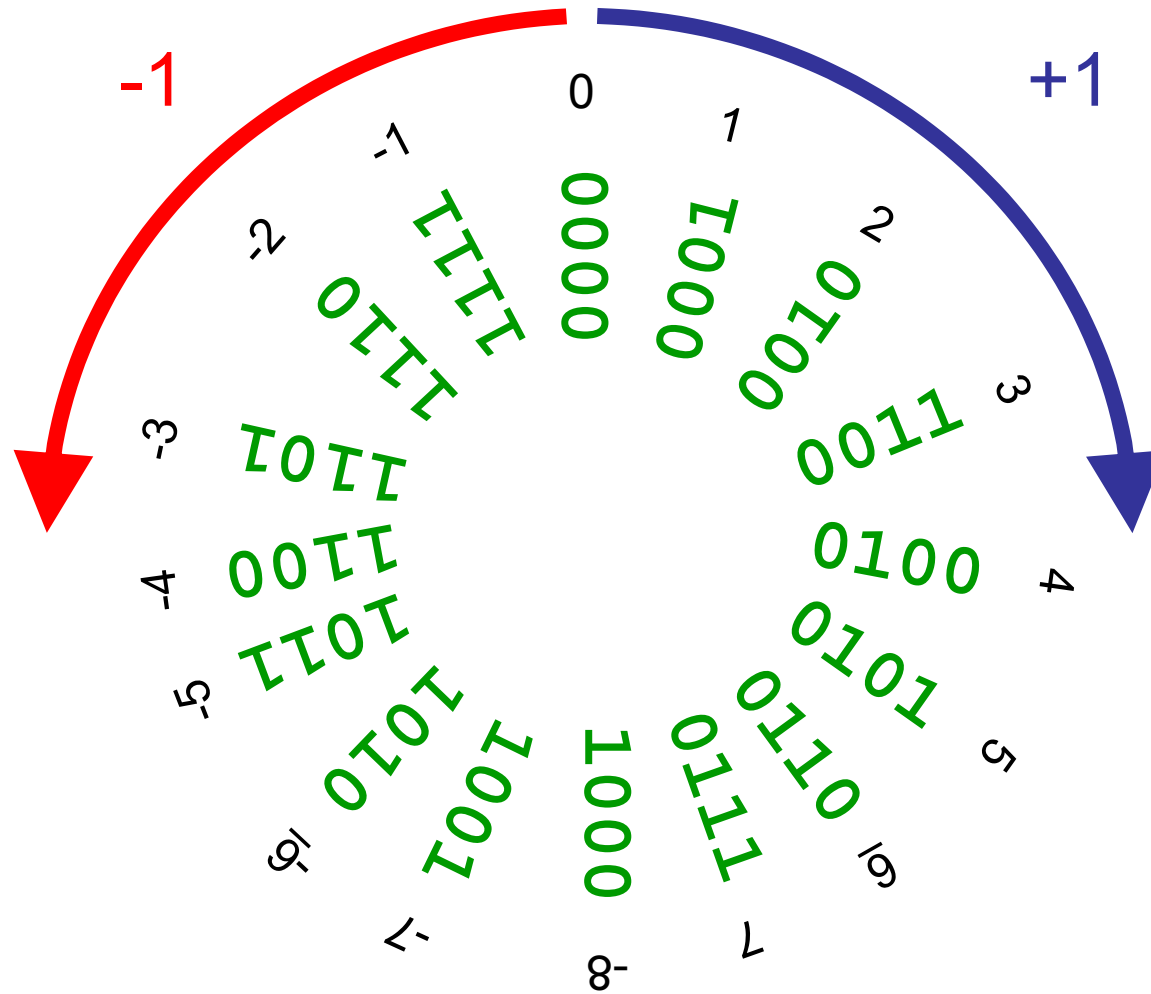
□ $(-2) + (-3) = -5$

$$\begin{array}{r} -2 + \\ -3 = \\ \hline -5 \end{array}$$

1 1 0 0 riporti

$$\begin{array}{r} 1110 + \\ 1101 = \\ \hline 1011 \end{array}$$

Circolarità della rappresentazione



Aritmetica con il complemento a due

- la somma di due numeri relativi di n cifre si esegue come nel caso dei numeri naturali
 - l'eventuale riporto sulla cifra $n+1$ viene ignorato
 - se il risultato eccede $2^{n-1} - 1$ si ottiene un numero negativo che non corrisponde al risultato desiderato
 - se il risultato è minore di -2^{n-1} si ottiene un numero positivo che non corrisponde al risultato desiderato
- la sottrazione tra due numeri relativi di n cifre si esegue complementando a due il secondo numero ed eseguendo la somma:
 - $n - m = n + C_2(m)$

Rappresentazione di numeri razionali

- ❑ La difficoltà della rappresentazione dei numeri razionali deriva dal fatto che un intervallo arbitrariamente piccolo ne contiene infiniti
- ❑ E' dunque possibile rappresentarne solo un sottoinsieme
- ❑ Una delle rappresentazioni più utilizzate è quella in virgola mobile, anche detta floating point

Rappresentazione in virgola mobile

- Un qualunque numero razionale può essere riscritto in modo da evidenziare una mantissa ed un esponente
 - esempi
 - $235.023 = .235023 \times 10^3$ (.235023 è la mantissa, 3 è l'esponente)
 - $-0.0000235 = -.235 \times 10^{-4}$ (-.235 è la mantissa, -4 è l'esponente)
- Ciò è vero anche per i numeri in base due
 - esempio
 - $1001001.0101 = .10010010101 \times 2^{-7}$
- La rappresentazione in virgola mobile consiste proprio nel rappresentare mantissa ed esponente del numero razionale espresso in base due
 - un bit viene generalmente usato per il segno di mantissa
 - una porzione dei bit viene dedicata alla rappresentazione della mantissa
 - i rimanenti bit vengono utilizzati per rappresentare l'esponente in complemento a due