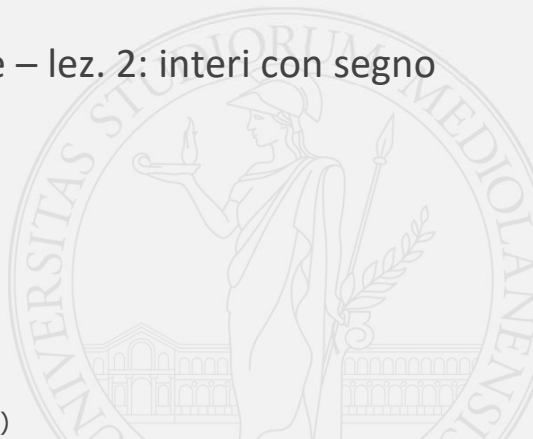




UNIVERSITÀ DEGLI STUDI DI MILANO

Rappresentazione dell'Informazione – lez. 2: interi con segno

Marco Tarini
(lucidi in collaborazione con prof.ssa Maria Damiani)



1



Sommario

Rappresentazione binaria dei numeri interi con segno:

- Rappresentazione in modulo e segno
 - Intervallo di rappresentazione
- Rappresentazione in complemento a 2 (C2)
 - Intervallo di rappresentazione
 - Regole pratiche
 - Somme e sottrazioni
 - Overflow



2

Rappresentazione dei numeri interi con segno

- **Interi con segno:** -2, -1, 0, +1, +2..
- Nella notazione che ci è più familiare, per indicare il segno si usa un ulteriore simbolo +/-
 $+ 345, \quad -27$
- Nel sistema binario, abbiamo a disposizione unicamente 2 simboli. Inoltre in un computer il numero di cifre che compongono un numero è prefissato. Il problema è dunque come rappresentare questa informazione col numero di bit a disposizione.
- Sia N il numero di bit. Consideriamo tre modalità di rappresentazione su N bit:
 - Modulo e segno
 - Rappresentazioni con offset (o «spiazzamento»)
 - Complemento a 2 (C2)

3

Rappresentazione in modulo e segno con N bit

Il bit più significativo indica il segno mentre gli altri bit rappresentano il numero in valore assoluto:

+ è rappresentato da 0

- è rappresentato da 1

Esempio: N=8 **0**000 0001 → +1
 1000 0001 → -1

4

Intervallo di rappresentazione

Si possono rappresentare i numeri nell'intervallo:

$$[-2^{N-1} + 1, +2^{N-1} - 1]$$

Es: $N=8 \rightarrow [-2^7 + 1, +2^7 - 1] \rightarrow [-127, +127]$

5

Svantaggi della rappresentazione in modulo e segno

- (1) Duplice rappresentazione dello 0
- (2) Le operazioni aritmetiche sono complesse

Esempio:

$N=3$

000	0
001	+1
010	+2
011	+3
100	0
101	-1
110	-2
111	-3

Due diverse
rappresentazioni
per lo 0

6

Rappresentazione con spiazzamento (o offset)

Idea: codifico X per rappresentare il valore $(X - \text{OFFSET})$
dove **OFFSET** (lo spiazzamento)
è una quantità scelta dalla codifica, una volta per tutte.

Per esempio, in base 10, con 2 cifre, con OFFSET **30**:

«40» rappresenta il valore $40 - 30 = 10$ (un numero positivo)

«30» rappresenta il valore $30 - 30 = 0$

«00» rappresenta il valore $0 - 30 = -30$ (un numero negativo)

Un vantaggio: un solo modo di rappresentare lo 0.

Uno svantaggio: lo zero si scrive in modo diverso da «tutte le cifre 0»

7

Rappresentazione con spiazzamento (o offset)

Domande:

1) in base B, con N cifre, e offset S,
qual è l'intervallo di rappresentabilità?
(cioè qual è il più grande e più piccolo numero esprimibile)?

2) Come si codificano in base 10, con 3 cifre, e spiazzamento 200, i
valori...

- 0
- +200
- -200
- 34

8

Rappresentazione con spiazzamento (o offset)

Domande:

1) in base B , con N cifre, e offset S ,
qual è l'intervallo di rappresentabilità? $[- S \dots B^N - 1 - S]$
(cioè qual è il più grande e più piccolo numero esprimibile)?

2) Come si codificano in base 10, con 3 cifre, e spiazzamento 200, i valori...

- 0 «200»
- +200 «400»
- -200 «000»
- 34 «234»

9

Rappresentazione in complemento a 2 (C2, o CP2)

- La rappresentazione in complemento a 2 è quella comunemente utilizzata negli elaboratori. Permette di ovviare ai problemi posti dalle altre rappresentazioni e fornisce una serie di vantaggi
- L'idea base è che il MSB (Most Significant Bit, la cifra binaria più a sinistra) vale il opposto del valore solito: cioè, su N cifre, vale MENO 2^{N-1}
- Esempio con 6 bit:

$$\begin{array}{cccccc} \textcolor{red}{1} & 0 & 1 & 1 & 1 & 0 \\ \textcolor{red}{\uparrow} & \textcolor{blue}{\uparrow} & \textcolor{blue}{\uparrow} & \textcolor{blue}{\uparrow} & \textcolor{blue}{\uparrow} & \textcolor{blue}{\uparrow} \\ -32 & 16 & 8 & 4 & 2 & 1 \end{array} \text{C2} \quad = \quad -\textcolor{red}{32} + 8 + 4 + 2$$
$$\quad \quad \quad = \quad -18$$

10

Rappresentazione in complemento a 2 (C2, o CP2)

Esempio con 4 bit:

Numeri ≥ 0	000 0	Unica rappresentazione dello 0
	001 +1	
	010 +2	
	011 +3	
Numeri < 0	111 -1	
	110 -2	
	101 -3	
	100 -4	

11

Rappresentazione in complemento a 2 (C2, o CP2)

Osservazione (capire perché):

-2^{n-1} $+2^{n-1}$ meno 1
1 1 1 ... 1 1 1 _{C2}

Domande:

- Come determino il segno di un numero in CP2?
- Come codifico il numero minore possibile (quindi, negativo) in CP2 a N bit? Quanto vale?
- Come codifico il numero maggiore possibile (quindi, positivo) in CP2 a N bit? Quanto vale?
- Mettendo tutti i bit ad 1, cosa codifico in CP2? (non dipende dal numero di bit!)

12

Rappresentazione in complemento a 2 (C2, o CP2)

Risposte: (capire come mai!)

- *Come determino il segno di un numero in CP2?*
Dal primo bit (come con modulo e segno!): 1 => negativo, 0 => positivo
- *Come codifico il numero minore possibile (quindi, negativo) in CP2 a N bit?*
100...00
Quanto vale? -2^{n-1}
- *Come codifico il numero maggiore possibile (quindi, positivo) in CP2 a N bit?*
011...11
Quanto vale? $+2^{n-1} - 1$
- *Mettendo tutti i bit ad 1, cosa codifico in CP2?*
Sempre il valore -1

13

Intervallo di rappresentazione in C2

- Intervallo di rappresentazione: $[-2^{N-1}, +2^{N-1} - 1]$
- La rappresentazione dello 0 è unica, quindi l'intervallo di rappresentazione comprende un numero in più rispetto alla rappresentazione in modulo e segno

Per N=3 $[-4, +3]$

Per N=4 $[-2^{4-1}, +2^{4-1} - 1] \rightarrow [-8, +7]$

Per N=8 $[-2^{8-1}, +2^{8-1} - 1] \rightarrow [-128, +127]$

14

Algoritmo per invertire il segno di un numero in CP2
(invertire il segno = calcolare l'opposto)

Passo 1: flip di ogni bit
Passo 2: aggiungo uno al risultato

ESEMPI (con 5 bit):

11010 $-16+8+2 = -6$	$= (\text{FLIP}) =>$	00101	$= (+1) =>$	00110 $4+2 = +6$
11111 -1	$= (\text{FLIP}) =>$	00000	$= (+1) =>$	00001 $+1$
01011 $8+2+1 = 11$	$= (\text{FLIP}) =>$	10100	$= (+1) =>$	10101 $-16+4+1 = -11$
00000 0	$= (\text{FLIP}) =>$	11111	$= (+1) =>$	00000 0

No overflow!

15

Algoritmo per invertire il segno di un numero in CP2:
una variante equivalente (forse più facile?)

Un solo passo: scandendo il numero da destra a sinistra, flip di ogni bit a partire dal primo 1 *escluso*

Esempi (su 5 bit):

11010 $-16+8+2 = -6$	$=>$	00110 $4+2 = +6$
		
01011 $8+2+1 = +11$	$=>$	10101 $-16+4+1 = -11$
		

16

Rappresentare un numero negativo in CP2

Rappresentare il numero -35 in complemento a 2 su 8 bit

Modo 1)

Metodo 1:

- Il numero è negativo, quindi il primo bit è **1**, e vale $-2^7 = -128$
- Gli altri sette bit devono valere in tutto... $128 - 35 = 93 = 1011101_2$
- Risposta: **11011101**₂

Metodo 2:

- Rappresento 35 in base 2: 0010 0011
- Calcolo l'opposto: **1101 1101**

N - 1

17

Decodificare un numero da CP2

Quale intero x è rappresentato in CP a 2 dalla sequenza binaria: 1100 1010 ?

1100 **1**010

Modo 1: $-2^7 + 2^6 + 2^3 + 2^1 = -128 + 64 + 8 + 2 = -54$

Modo 1)

Modo 2: Il numero è negativo (dato che comincia con 1).

Per trovare il suo valore assoluto, inverte il segno, ottenendo **0011 0110**, che vale

$$2^5 + 2^4 + 2^2 + 2^1 = +32 + 16 + 4 + 2 = 54$$

0011 0110

Quindi, la risposta è **-54**

(nota: di caso in caso, un dei due modi può essere più veloce)

18

Operazione di addizione/sottrazione in C2

Valgono le seguenti proprietà:

- 1) La somma di 2 numeri x e y in complemento a due è uguale al complemento a due della somma

$$(x+y)_c = x_c + y_c$$

- 2) La operazione di sottrazione tra due numeri interi può essere effettuata come addizione

$$x_c - y_c = x_c + (-y)_c$$

Queste proprietà rendono più semplici i circuiti che realizzano le operazioni aritmetiche

19

Esempio

Calcolare: $11 - 13$ con $N=5$

$$11 - 13 = 11 + (-13)$$

In complemento a 2 :

$$+11_{10} \rightarrow 01011_2$$

$$-13_{10} \rightarrow 10011_2$$

→ posso fare somma e differenza con lo stesso procedimento, quindi con lo stesso **circuito**!

11 ← riporti

$$\begin{array}{r} 01011 + \\ 10011 = \\ \hline \end{array}$$

$$11110 \rightarrow -2_{10}$$

20

Overflow nella addizione in C2

Si ricorda che l'addizione genera overflow quando la somma non ricade nell'intervallo di rappresentazione, e quindi non è rappresentabile.

Da tenere presente: non vale la stessa regola vista per i numeri interi senza segno. In particolare il bit di riporto dalla cifra più significativa non è rilevante

Regola pratica per determinare se si è verificato overflow:

- La somma fra un numero positivo e un numero negativo non genera overflow.

L'eventuale riporto nella somma viene ignorato

- Si verifica invece overflow quando gli addendi sono entrambi positivi o negativi e il segno della somma è **diverso** da quello degli addendi

21

Esempio

N=4 Effettuare la somma +7-2

+ 7 -2 $\rightarrow$ 0111 + 1110 = +5

$$\begin{array}{r} 0111_2 + \\ 1110_2 \\ \hline 1\ 0101_2 \end{array}$$

Poichè gli operandi sono di segno diverso, non si può verificare overflow. La cifra più significativa che risulta dal riporto viene ignorata. Il risultato è dunque: 0101

22

Nei linguaggi di programmazione ad alto livello: TIPI BASE L'esempio del go

Senza segno («unsigned»)					
Tipo base	Sinonimo	Bits	Bytes	Val min	Val max
uint8	byte	8	1	0	255
uint16		16	2	0	64 mila
uint32	uint uintptr (di solito)	32	4	0	4 milioni
uint64		64	8	0	16 miliardi

Circa!
Qual è la cifra esatta,
in base 16?

Con segno					
Tipo base	Sinonimo	Bits	Bytes	Val min	Val max
int8		8	1	-128	+127
int16		16	2	-32 mila	+32 mila
int32	int (di solito)	32	4	-2 milioni	+2 milioni
int64		64	8	-8 miliardi	+8 miliardi

Circa!
Qual è la cifra esatta,
in base 16?

23

Esercizi di riepilogo

Rappresentare con numero di cifre $N = 8$ (un byte)

- I numeri: -14, -20, +31 in C2 e modulo e segno
- Intervallo di rappresentazione per rappresentazione in C2 e modulo e segno
- Elencare i numeri rappresentabili in C2 compresi nell'intervallo [-24, -20]
- il più grande numero in valore assoluto rappresentabile in complemento a 2 e in modulo e segno
- Determinare i numeri in C2 rappresentati dalla sequenza:
1111 1111
1010 1010
- Trovare due numeri in C2 che sommati generano un OVERFLOW
- Converti in C2 i valori 12 e -30, e, usando le due rappresentazioni, calcola la rappresentazione di 12-30

24